https://doi.org/10.20965/jaciii.1998.p0088

Paper:

Improved Fuzzy and Neural Network Algorithms for Word Frequency Prediction in Document Filtering[1]

Péter Baranyi[*], László T. Kóczy[**], and Tamás D. Gedeon[***]

[*]Computer and Automation Institute Hungarian Academy of Sciences and
Dept. of Automation Technical University of Budapest
Budafoki u. 8, Budapest, H-1111, Hungary
E-mail: baranyi@elektro.get.bme.hu
[**]Dept. of Telecommunication and Telematics, Technical University of Budapest
Sztoczek u.2, Budapest, H-1111, Hungary
E-mail: koczy@ttt.bme.hu
[***]Dept. of Information Engineering, School of Computer Science and Engineering,
The University of New South Wales
Sydney 2052 Australia
tom@cse.unsw.edu.au

[Received October 10, 1997; accepted January 25, 1998]

With very large document collections or high-volume document streams of , finding relevant documents is a major information filtering problem. An aid to information retrieval systems produces a word frequency measure estimated from important parts of the document using neural network approaches. In this paper, a fuzzy logic technique and, as its simplified case, a neural network algorithm are proposed for this task. The comparison of these two and an alternative neural network algorithm are discussed.

Keywords: Information retrieval, Neural network, Fuzzy logic, Parameter set reduction

1. Introduction

An information system allows users to efficiently retrieve documents relevant to current interest from a very large collection.[7] The information retrieval system must determine relevant documents by matching key words or phrases of user interest specified in the query, fragments of natural language texts, and words and phrases used in documents of the collection. To be effective, however, a matching search must consider the whole topic determined by the queried words and their synonyms, or relevant documents are lost. If, e.g., only synonyms are used in the text of certain very relevant documents, these documents may be completely ignored. To alleviate this problem, much research has gone into creating and maintaining information filters,[10-14] specifying categories, building and synonym lists.[8] In these systems, the problem of automatic indexing is inevitable.[1,2,5,6] The aim of indexing text items is to (implicitly) summarize their content.[9] Once important keywords are found and their occurrence frequencies are known or estimated, it is possible to build up cooccurrence maps and hierarchical cooccurrence relations.[2,3,16]

Proposed automatic indexing algorithms are based on a combination of significant measures such as the frequency-keyword approach, title-keyword, location-method and cue-method.[2,3] One idea for determining the most appropriate way of combining indexing parameters is based on neural network approaches that can learn a composition function of significance measures. We train a neural network to estimate the word frequency measure component of a retrieval index from a relatively small proportion of document text, e.g., the first and last 20%, so, all of these measures can be calculated locally and do not require the global document collection information required in real frequency-keyword calculations. This may provide considerable gains in the future in highly parallel implementation.[15] Consequently, the efficiency of information retrieval depends significantly on the effectiveness of the frequency-keyword measure, hence, the applied algorithms.[17]

In this paper, a fuzzy logic algorithm (FLA) is proposed with examples for the estimation of the word frequency measure component in comparison to neural network algorithms. It will be shown that this algorithm also can be introduced as a neural network, where neurons can include different piece-wise linear transfer functions. A simple neural network (SNN) as a special case of the proposed FLA will also be discussed with examples. These algorithms are optimized considering the main difficulties of these applications, i.e., calculation time complexity.

The advantage of applying neural networks or fuzzy logic technique lies in their ability to imitate and implement the actions of expert operator(s) without the need of accurate mathematical models. The drawback, however, is that there is no standardized framework regarding the design, optimality, reducibility, and definition of the concept. These algorithms, either generated by expert operators or by some learning or identification schemes, may contain redundant, weakly contributing, or unnecessary components. To achieve a good approximation, some approaches may overestimate the number of hidden neural network units and layers or the number of fuzzy rules, thereby resulting in a

1 Supported by Australian Research Council Gran No. A49600961

© Fuji Technology Press Ltd. Creative Commons CC BY-ND: This is an Open Access article distributed under the terms of the Creative Commons Attribution-NoDerivatives 4.0 International License (http://creativecommons.org/licenses/by-nd/4.0/).

large neural network or fuzzy rule base causing problems in computational time complexity. Applying learning algorithms to a large number of neurons or having a large number of rules in a fuzzy rule base requires considerable calculation time even in small applications

As mentioned above, the problem is that the collection of documents, from which the selected ones must be retrieved may be extremely large. Thus, two important aims must be taken into consideration to generate a neural network or fuzzy logic representation. One is to achieve a sufficient estimation of the frequency of considered words, including their cooccurrence relations. The other is to use simplified functions in neurons and to reduce the number of layers and neurons or fuzzy rules as much as possible to reduce the required computational effort.

We first describe a standard neural network approach for estimating word frequency.[1,2,5,6] The algorithm proposed reduces computational time complexity. The SNN has fewer neurons and simplified functions in units rather than the usual sigmoid function requiring an exponential operation. The advantages and disadvantages of the FLA and SNN are discussed below.

Using the proposed methods, the learning time is considerably reduced and the obtained results are significantly improved.

2. Description of Standard Neural Network Approach

Take the last improved network from Ref.2) **(Fig.1)**. Words and training parameters are selected as follows: n_w = 75 words are used to index a collection of 350 documents. The chosen neural network has three inputs ($n_p = 3$) for each of the 75 words for the title-keywords, location-, and cue-frequencies. Thus, the total number of input is $I = n_w n_p = 225$. The inputs value is $x_i \in [0,1]$, where $i = 1..I$. The output values $y_k \in [0,1] (k = 1, K, n_w)$ are the estimated frequency-keyword measures for each word. The number of output neurons is $O = 75$. The number of hidden neurons is $H = 4$. The transfer function included in the neurons is:

$$f(a) = \frac{1}{1+e^{-a}}$$

The essence of the experiment is to use these measures as training inputs to the network, and then to attempt the prediction of frequency-keyword measures.

The network input vector is $\underline{\mathbf{x}} = [x_i] (i = 1..s..I$ where $s =$

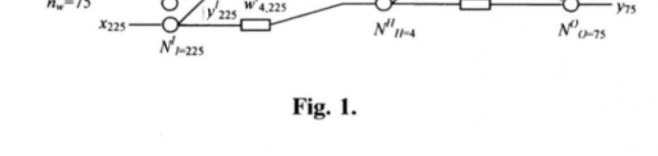

Fig. 1.

$(i_w - 1)n_w + i_p$, and $i_w = 1, K, n_w, i_p = 1, K, n_p)$ The output value of input neuron N_i^I is $y_i^I = x_i$, of hidden neuron N_j^H is $y_j^H (j = 1..H)$ and of output neuron N_k^O is $y_k (k = 1..O)$. The connection weight matrix between neurons of input and hidden layers is $\underline{\underline{\mathbf{W}}}^I = [w_{j,i}^I]$, between hidden and output layers: $\underline{\underline{\mathbf{W}}}^H = [w_{k,j}^H]$. Hidden and output layers are biased, meaning there is an additional neuron in both input and hidden layers providing a constant output of 1. The bias weight vector for the hidden layer is $\underline{\mathbf{w}}^{I,\,bias} = [w_j^{I,\,bias}]$, and for the output layer $\underline{\mathbf{w}}^{H,\,bias} = [w_k^{H,\,bias}]$. The input values of neuron N_j^H in the hidden layer are

$$s_{j,i}^H = w_{j,i}^I\,,\, y_i^I,$$

and from the bias neuron $x_{j,0}^H = w_j^{I,\,bias} \cdot 1$Similarly, the input values of neuron $N_k O$ in the output layer are $x_{k,j}^O = w_{k,j}^H\, y_j^H$, and from the bias neuron $x_{k,0}^O = w_k^{H,\,bias} \cdot 1$.

The steps of forward propagation are as follows: Let $\underline{\tilde{\mathbf{y}}}$ be the output calculated by the network as input neurons have no transfer function):

$$1.\ \underline{\mathbf{y}}'^H = [\underline{\underline{\mathbf{W}}}^I\ \underline{\mathbf{w}}^{I,\,bias}] \begin{bmatrix} \underline{\mathbf{y}}^I \\ 1 \end{bmatrix} \quad \dots\dots\dots\dots\dots (1)$$

where $\underline{\mathbf{y}}'^H$, $\underline{\underline{\mathbf{W}}}^I$, $\underline{\mathbf{w}}^{I,\,bias}$, and contain elements y'^H_j, $w_{j,i}^I$, $w_j^{I,\,bias}$ and y_i^I, respectively.

$$2.\ y_j^H = f(y'^H_j) = \frac{1}{1+e^{-y'^H_j}},$$

$$3.\ \underline{\mathbf{y}}' = [\underline{\underline{\mathbf{W}}}^H\ \underline{\mathbf{w}}^{H,\,bias}] \begin{bmatrix} \underline{\mathbf{y}}^H \\ 1 \end{bmatrix}$$

where $\underline{\mathbf{y}}'^O$, $\underline{\underline{\mathbf{W}}}^H$, $\underline{\mathbf{w}}^{H,\,bias}$, and $\underline{\mathbf{y}}^H$ contain elements y'^O_k, $w_{k,j}^H$, $w_k^{H,\,bias}$, and y_j^H, respectively.

$$4.\ \tilde{y}_k = f(y'_k) = \frac{1}{1+e^{-y'_k}}$$

The error is

$$5.\ \underline{\delta}^O = \underline{\mathbf{y}} - \underline{\tilde{\mathbf{y}}}$$

Thus, based on error back propagation

$$6.\ \delta'^O_k = \tilde{y}_k(1 - \tilde{y}_k)\delta_k^O$$
$$7.\ \underline{\delta}^H = \underline{\underline{\mathbf{W}}}^{H^T} \underline{\delta}'^O$$
$$8.\ \delta'^H_j = y_j^H(1 - y_j^H)\delta_j^H$$
$$9.\ \Delta[\underline{\underline{\mathbf{W}}}^H\ \underline{\mathbf{w}}^{H,\,bias}] = \eta \underline{\delta}'^O\, \underline{\mathbf{y}}^{H^T}$$
$$10.\ \Delta[\underline{\underline{\mathbf{W}}}^I\ \underline{\mathbf{w}}^{I,\,bias}] = \eta \underline{\delta}'^H\, \underline{\mathbf{y}}^{I^T}$$

where $0 < \eta < 1$ is the learning parameter. After training, the network has formed an internal representation of the relative importance of the different significance measures.

3. Simplified Neural Network (SNN) Approach

To reduce calculation time as opposed to the standard approach described in Section 2 and applied in our former approach,[5,6] the transfer function of neurons, which is an exponential function requiring relatively great computation, is omitted. From the nature of the problem, we concluded that bias weights may not be necessary, because zero occurrences of any particular word have no effect on the significance of other words. If a document included most of the possible words, we could clearly say that any exceptions were significant. Since real documents contain few words, the absence of any particular word compared to the number of all possible absent words provides little real information. Hence, we can ignore bias weights as the neural network hyperplanes necessarily pass through or near the origin. In this case the hidden layer does not have any role and the output is calculated without steps 3 and 5 (1) and with bias inputs set to 0:

$$\underline{\tilde{\mathbf{y}}} = \underline{\mathbf{W}}^H \underline{\mathbf{W}}^I \underline{\mathbf{x}} = \underline{\mathbf{W}}\,\underline{\mathbf{x}} \quad \dots\dots\dots\dots\dots (2)$$

However, the hidden layer can be used to reduce computational complexity. The size of $\underline{\mathbf{W}}^I$ is $I \times H$, of $\underline{\mathbf{W}}^H$ is $H \times O$. If only $\underline{\mathbf{W}}$ used the total number of connections is $I \cdot O$. Thus, if

$$H < \frac{I \cdot O}{I + O}$$

then connection complexity is reduced compared to when only $\underline{\mathbf{W}}$ is used.

Singular value decomposition (SVD) is applied to choose the optimal H in reducing connection complexity. Applying SVD on $\underline{\mathbf{W}}$ results in

$$\underline{\mathbf{W}} = \underline{\mathbf{U}}\,\underline{\mathbf{D}}\,\underline{\mathbf{V}}^{\mathrm{T}}$$

where $\underline{\mathbf{U}}$ and $\underline{\mathbf{V}}$ are orthogonal matrices with size $n_u \times n_u$ and $n_v \times n_v$ respectively. The $n_u \times n_v$ $\underline{\mathbf{D}}$ matrix contains the singular values of $\underline{\mathbf{W}}$ in decreasing order. The maximum number of nonzero singular values is $n_{SVD} = \min(n_u, n_v)$. Singular values indicate the importance of corresponding columns in matrices $\underline{\mathbf{U}}$ and $\underline{\mathbf{V}}$. To achieve a simpler connection, singular values equal to or near to zero should be discarded, resulting in an approximation of $\underline{\mathbf{W}}$. Let n_r be the number of singular values to be retained. Partition the matrices $\underline{\mathbf{U}}$, $\underline{\mathbf{V}}$ and $\underline{\mathbf{D}}$ as

$$\underline{\mathbf{U}} = |\underline{\mathbf{U}}_r\ \underline{\mathbf{U}}_d|,\ \underline{\mathbf{V}} = |\underline{\mathbf{V}}_r\ \underline{\mathbf{V}}_d| \text{ and}$$
$$\underline{\mathbf{D}} = \begin{vmatrix} \underline{\mathbf{D}}_r & \underline{\mathbf{O}}_{n_r \times (n_v - n_r)} \\ \underline{\mathbf{O}}_{(n_u - n_r) \times n_r} & \underline{\mathbf{D}}_r \end{vmatrix}$$

Here, $\underline{\mathbf{O}}_{a \times b}$ is an $a \times b$ matrix of zeros, and $\underline{\mathbf{U}}_r$, $\underline{\mathbf{U}}_d$, $\underline{\mathbf{V}}_r$ and $\underline{\mathbf{V}}_d$ are $n_u \times n_r$, $n_u \times (n_u - n_r)$, $n_v \times n_r$ and $n_v \times (n_v - n_r)$ matrices, respectively. Matrix $\underline{\mathbf{D}}_r$ is diagonal and contains retained n_r singular values. $\underline{\mathbf{U}}_r$ and $\underline{\mathbf{V}}_r$ contain the corresponding columns of $\underline{U}$ and $\underline{\mathbf{V}}$. Matrix $\underline{D}_d$ contains the remaining $(n_{SVD} - n_r)$ singular values to be discarded. An approximation of $\underline{\mathbf{W}}$, $\underline{\tilde{\mathbf{W}}}$ is obtained by

$$\underline{\tilde{\mathbf{W}}} = \underline{\mathbf{U}}_r \underline{\mathbf{D}}_r \underline{\mathbf{V}}_r^{\mathrm{T}}$$

The approximation is exact if all nonzero singular values are kept as $\underline{\tilde{\mathbf{W}}} = \underline{\mathbf{U}}_r \underline{\mathbf{D}}_r \underline{\mathbf{V}}_r^{\mathrm{T}}$. Consequently, if $n_r < (I \cdot O)/(I + O)$ then let $\underline{\mathbf{W}}^I = \underline{\mathbf{U}}_r$ and $\underline{\mathbf{W}}^H = \underline{\mathbf{D}}_r \underline{\mathbf{V}}_r$ or obviously $\underline{\mathbf{W}}^I = \underline{\mathbf{U}}_r \underline{\mathbf{D}}_r$ and $\underline{\mathbf{W}}^H = \underline{\mathbf{V}}_r$. If n_r is not less than $(I \cdot O)/(I + O)$, nonzero singular values can be discarded, but the error bound of SVD reduction must be considered.

As mentioned above, if $\underline{\mathbf{D}}_d$ contains nonzero elements, the product of retained matrices is an approximation of $\underline{\mathbf{W}}$. So,

$$\underline{\mathbf{W}} - \underline{\tilde{\mathbf{W}}} = \underline{U}_d \underline{D}_d \underline{\mathbf{V}}_d^T$$

As the columns of $\underline{\mathbf{D}}_d$ and $\underline{\mathbf{V}}_d$ have the Euclidean norm of unity, the absolute values of their elements must be bounded by 1. Denote by σ_i the i-th singular value of $\underline{\mathbf{D}}$, and we obtain

$$|\underline{\mathbf{W}} - \underline{\tilde{\mathbf{W}}}| = |\underline{\mathbf{U}}_d| \underline{\mathbf{D}}_d |\underline{\mathbf{V}}_d^{\mathrm{T}}| \le 1_{n_u \times (n_u - n_r)} \underline{\mathbf{D}}_d 1_{(n_v - n_r) \times n_v}$$
$$1_{n_u \times (n_u - n_r)} \underline{\mathbf{D}}_d 1_{(n_v - n_r) \times n_v} = \left(\sum_{i = n_r + 1}^{n_{SVD}} \sigma_i \right) 1_{n_u \times n_v} \quad . \ (3)$$

It is not necessary to take the absolute value of $\underline{\mathbf{D}}_d$, because each singular value is positive. Equation (3) implies

$$\left| w_{i_l, i_{l-1}} - \tilde{w}_{i_l, i_{l-1}} \right| \le d = \sum_{i = n_r + 1}^{n_{SVD}} \sigma_i$$

Thus, $\left| y_i - \tilde{y}_i \right| \le \sum_{i=1}^{I} dx_i$, where I is the number of inputs. The learning phase of the application results in a strongly nonsingular matrix $\underline{\mathbf{W}}$. Discarding only one singular value results in a large error. (In our case $(I \cdot O)/(I + O) = 56.25$, that means at least 19 singular values must be discarded for reduction.) This is quite obvious from the behavior of training parameters. Suppose that the network has one hidden layer, where the number of neurons is H.

Look at matrices $\underline{\mathbf{Y}}_T = [\underline{\mathbf{y}}_1 \dots \underline{\mathbf{y}}_{n_t}]$ and $\underline{\mathbf{Y}}_{T2} = \left[\underline{\mathbf{y}}_1^H \dots \underline{\mathbf{y}}_{n_t}^H\right]$, where $\underline{\mathbf{y}}_t^H$ contains the output values of the hidden layer for the t-th training input. In ideal training, we obtain

$$\underline{\mathbf{Y}}_T = \underline{\mathbf{W}}^H \underline{\mathbf{Y}}_{T2}$$

where $n_t >> O$ and $n_t >> H$.

Suppose that $H < I$ and $H < O$. The rank of $\underline{\mathbf{W}}^H$ and $\underline{\mathbf{Y}}_{T2}$ is the maximum of H. Thus, the rank of $\underline{\mathbf{W}}^H \underline{\mathbf{Y}}_{T2}$ is the maximum of H. Training parameters are from randomly chosen sample documents, so the rank of $\underline{\mathbf{Y}}_T$ is almost certainly larger than H but not larger than O. This matrix is most probably nonsingular, meaning H must be close to O. In our case, $O = 75$, so H must be close to 75. For reduction, H must be less than 57 (as calculated above) or the use of a hidden layer increases the number of connections. Consequently, we do not use any hidden layer.

In the same way, the number of neurons (4) in hidden layers in the standard network approach is also insufficient. It should have been closer to 75. The neurons in former networks contained exponential functions, and with 75 neurons in the hidden layer instead of 4, this would have resulted in considerablly increased calculation time.

Assume the algorithm of the proposed SNN:

1) $\underline{\tilde{\mathbf{y}}} = \underline{\mathbf{W}}\,\underline{\mathbf{x}}$,
2) $\underline{\delta} = \underline{\mathbf{y}} - \underline{\tilde{\mathbf{y}}}$,
3) $\Delta\underline{\mathbf{W}} = \eta\,\underline{\delta}\,\underline{\mathbf{x}}^{\mathrm{T}}$

SNN is much simpler than in 1). Negative word frequency measurements are not interpretable. The new method may result in negative values, so output must be bounded below.

4. Fuzzy Logic Algorithm (FLA)

Input and output are characterized the same way as in the SNN approach.

Characterization of fuzzy sets

To reduce computation, we use triangular fuzzy sets as $\left| A_{i,k=1} \ \dots \ A_{i,k=K_i} \right|$, where K_i is the number of antecedent sets on input universe $X_i = [0,1]$ such that

$$\text{Core}\{A_{i,k}\} = a_{i,k},\ \text{support}\{A_{i,k}\} = [a_{i,k-1}, a_{i,k+1}],\ \text{support}\{A_{i,1}\} = [a_{i,1}, a_{i,2}\},\ \text{support}\{A_{i,K_i}\} = [a_{i,K-1}, a_{i,K_i}].$$

Considering that input values are most frequently between 0 and 0.1 in each input universe, let $a_{i,1} = a_1 = 0$, $a_{i,2} = a_2 = 0.05$, $a_{i,3} = a_3 = 0.1$, $a_{i,4} = a_4 = 1$ and $K_i = K = 4$. The consequent fuzzy sets $B_{o,m}$ ($0 = 1..n_w$ and $m = 1..M$, where M is the number of consequent sets on each output universe y_o) are singleton sets as:

$$\mu_{B_{o,m}}(y_o) = \delta(y_{o,m});$$

$y_o \in Y_o$. For the observation A^*_i we use the singleton set as core$\{A^*_i\} = x_i$, where x_i is the input value on X_i.

Characterization of rules

The number of rules obtained by all combination of antecedent sets is $\prod_i K_i = K^I = 4^{225}$, which is unrealistically high. To reduce the rule base, we consider only the rules as follows:

$$\text{If } A_{i,k} \text{ then } B_{o,m} \Rightarrow \delta(y_{o,m})$$

where $m = (i-1)K + k$, so $M = I \cdot K$. Thus, the number of rules is 67500.

Characterization of inference

We use fuzzy inference based upon product-sum-gravity.[4)]

Product: Each rule gives a contribution set of the corresponding consequents $B_{o,m}$ multiplied by the input membership degree. If antecedent $A_{i,k}$ has membership degree $\mu_{A_{i,k}}(x_i)$, then this rule yields a contribution set $B_{o,m}$ in the form

$$\mu_{A_{i,k}}(x_i)\delta(y_{o,m})$$

Sum: All contribution sets are summed to produce the fuzzy set conclusion $\widetilde{B}_o$ as

$$\widetilde{\mu}_{B_o}(y_o) = \sum_{i,k} \mu_{A_{i,k}}(x_i) B_{o,m} = \sum_{i,k} \mu_{A_{i,k}}(x_i)\delta(y_{o,m})$$

which might not be directly interpretable as a fuzzy set, because function $\widetilde{\mu}_{B_o}(y_o)$ may be larger than 1. In this case, it should be normalized from the theoretical viewpoint to obtain a real fuzzy set B_o:$\mu_{B_o}(y_o)$.

Gravity: We apply center of gravity defuzzification to obtain output value y_o. Thus, output values are calculated as:

$$y_o = \frac{\sum_{i,k} \mu_{A_{i,k}}(x_i) y_{o,m}}{\sum_{i,k} \mu_{A_{i,k}}(x_i)} \quad \dots\dots\dots\dots\dots \quad (4)$$

Considering the characterization of the antecedent sets the membership degrees of the antecedent sets at any value within the universe sum to 1. Thus,

$$\sum_{j,k} \mu_{A_{j,k}}(x_i) = I$$

means this rule base performs a piece-wise linear approximation as from 4)

$$y_o = \sum_{i,k} \mu_{A_{i,k}}(x_i) \frac{y_{o,m}}{I}$$

To train the FLA the same algorithm can be used as for the neural network. Learning does not tune all sets, but only the position of the consequent sets. Let $\underline{\mathbf{m}}$ be a vector that contains membership degrees as: $\underline{\mathbf{m}} = [m_1 \dots m_m \dots m_M]$, where $m_m = \mu_{A_{i,k}}(x_i)$ and $m = (i-1)K + k$. Matrix $\underline{\underline{\mathbf{B}}} = [y_{o,m}]$contains the core of $B_{o,m}$. The steps of the algorithm are:

$$1)\ \underline{\tilde{\mathbf{y}}} = \underline{\underline{\mathbf{B}}}\,\underline{\mathbf{m}},$$
$$2)\ \underline{\delta} = \underline{\mathbf{y}} - \underline{\tilde{\mathbf{y}}},$$
$$3)\ \Delta\underline{\underline{\mathbf{B}}} = \eta\,\underline{\delta}\,\underline{\mathbf{m}}^{\mathrm{T}}$$

This is much simpler than 1).

In our application, if an input parameter has a zero value, it has zero contribution to the output value, implying that the position of the consequent of rules If $A_{i,1}$ then $B_{o,m=(i-1)K+1}$ is zero, namely, values $y_{o,(i-1)K+1}$ are not tuned.

5. SNN as a Special Case of the FLA and their Comparison

The number of trained parameters in the SNN is $I \cdot O$. In the fuzzy logic based algorithm it is $I \cdot O \cdot K$. Thus, the FLA requires K times more calculation time but much better results can be obtained. Equation (4) can be written as:

$$y_o = \sum_{i,k} \mu_{A_{i,k}}(x_i) \frac{y_{o,m}}{I} = \frac{1}{I}\sum_i f_{i,o}(x_i)$$

$$f_{i,o}(x_i) = \sum_k \mu_{A_{i,k}}(x_i) y_{o,m} = \frac{\sum_k \mu_{A_{i,k}}(x_i) y_{o,m}}{\sum_k \mu_{A_{i,k}}(x_i)}$$

where $\sum_k \mu_{A_{i,k}}(x_i) = 1$. Thus, $f_{i,o}(x_i)$ can be considered an explicit function for the rule base where the number of rules is K (**Fig.2**). This function is the contribution function of the i-th input universe to the o-th output.

This FLA can be represented as a neural network, where neurons have different types of function linearly approximated. The SNN uses linear contribution function as (**Fig.3**):

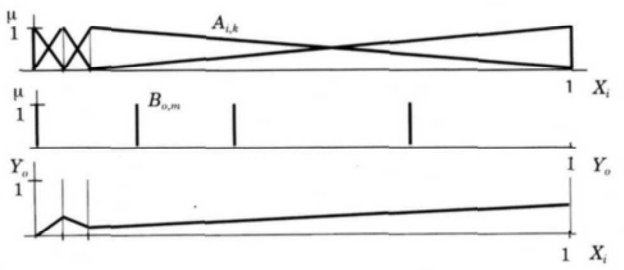

Fig. 2.

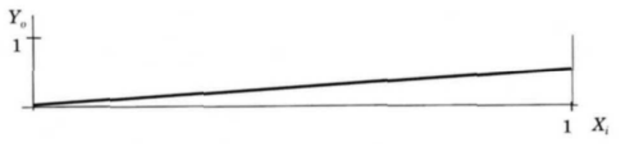

Fig. 3.

$$y_o = \sum_i f_{i,o}(x_i), \text{ where } f_{i,o}(x_i) = w_{o,i} x_i$$

It can be said, consequently, that a proper contribution function can be trained for each input-output in the fuzzy logic algorithm. The fuzzy logic algorithm can be considered as the sum of simplified neural networks as:

$$y_o = \frac{i}{I} \sum_i f_{i,o}(x_i) = \sum_i f_{i,o}(x_i);$$

where

$$f_{i,o}(a) = \begin{cases} f_{1,i,o}(a) = w_{1,i,o}a & \text{if } a_{i,1} \le a < a_{i,2} \\ & \text{otherwise } f_{1,i,o} = 0 \\ f_{2,i,o}(a) = w_{2,i,o}a + w^b_{2,i,o} & \text{if } a_{i,2} \le a < a_{i,3} \\ & \text{otherwise } f_{2,i,o} = 0 \\ \vdots & \vdots \\ f_{k-1,i,o}(a) = w_{k-1,i,o}a + w^b_{k-1,i,o} & \text{if } a_{i,K-1} \le a < a_{i,K} \\ & \text{otherwise } f_{k-1,i,o} = 0 \end{cases}$$

Thus

$$y_o = \sum_{z=1}^{K-1} \sum_i f_{z,i,o}(x_i)$$

which is the sum of K-1 neural networks, where input neurons have bias input value $w^b_{z,i,o}$, except in the first network.

From this viewpoint, the SNN is a special case of the FLA. Namely, the proposed network is a fuzzy algorithm where the number of antecedent sets on each input universe is K=2 and the supports are $[a_{i,1} = 0, a_{i,2} = 1]$. Consequently, increasing K, the FLA provides more powerful description than the SNN. The disadvantage is that, using the FLA, all pieces of the linearly approximated contribution functions must be separately tuned. This means that the training parameter collection must have input values in every $[a_{i,k},$

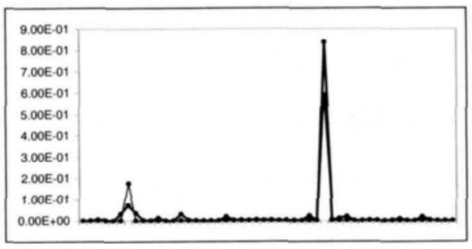

Fig. 4. Result using standard method

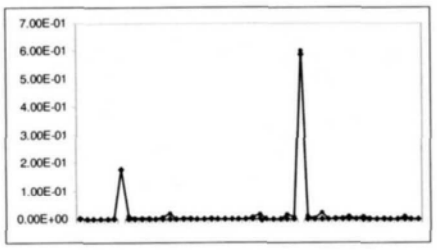

Fig. 5. Result using SNN

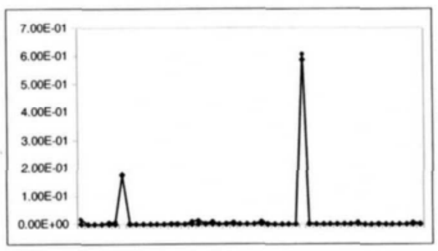

Fig. 6. Result using FLA

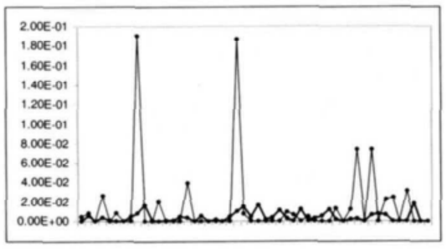

Fig. 7. Result using standard method

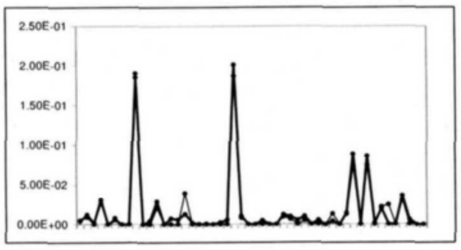

Fig. 8. Result using SNN

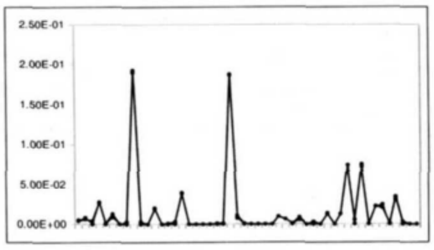

Fig. 9. Result using FLA

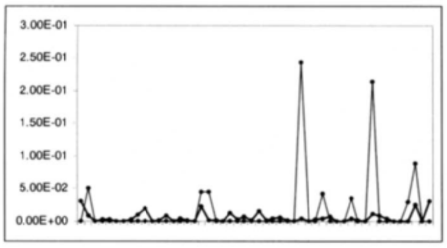

Fig. 10. Result using standard method

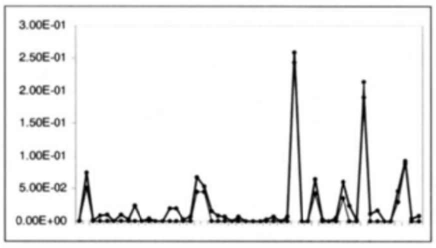

Fig. 11. Result using SNN

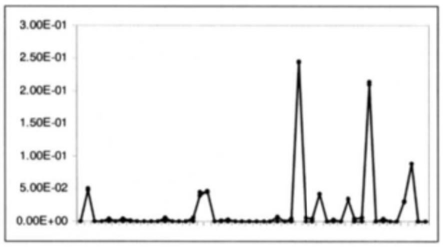

Fig. 12. Result using FLA

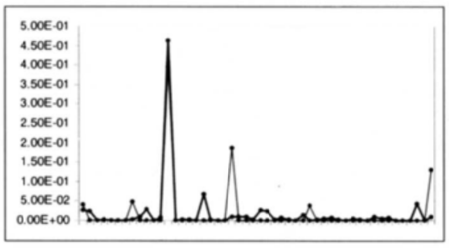

Fig. 13. Result using standard met

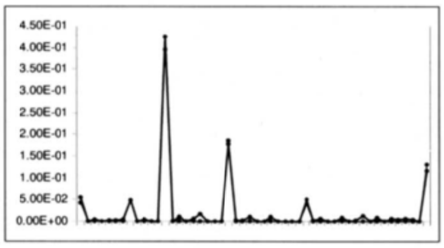

Fig. 14. Result using SNN

$a_{i,k+1}$] interval. In the SNN, only one interval is used on each input, as mentioned. Thus, it is enough if the training parameter collection has values for every input.

Consequently the FLA requires documents richer in the diversity of the frequency of considered words, but results in better estimation. In the next section, an example shows what happens when not all pieces of contribution functions are trained.

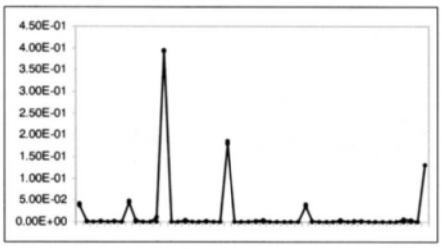

Fig. 15. Result using FLA

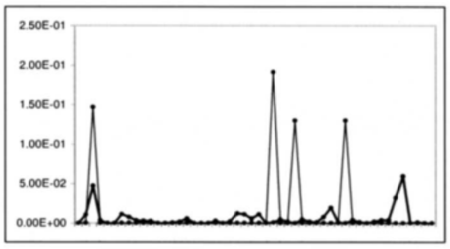

Fig. 16. Result using standard method

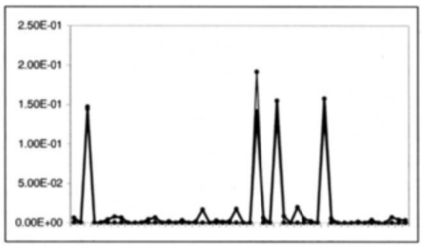

Fig. 17. Result using SNN

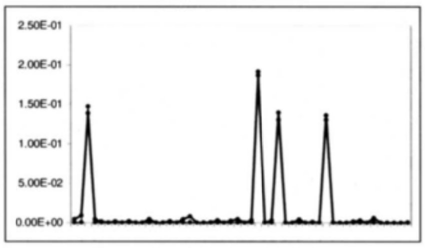

Fig. 18. Result using FLA

6. Experiments and Results

As shown, the SNN is simpler than the standard neural network approach, so learning time was much less. Further, the simplified method requires 400-500 epochs of training to achieve a sufficient estimation, instead of 1000 epochs.[2)] The same experiments were conducted using the SNN, FLA and standard approach. In figures the horizontal axis means the set of considered words and the vertical axis contains the frequency-keyword measure. To indicate the difference between results, points on diagrams are connected by lines. Points connected by thin lines show the real frequency-keyword measure from whole documents. Points connected by bold lines show estimated measures. **Figures 4-6** show a result where the input-output used are chosen from training parameters to test the effectiveness of training. It is a simple case where only one word dominates. The methods result in a similar estimation. **Figures 7-9** show a result for training

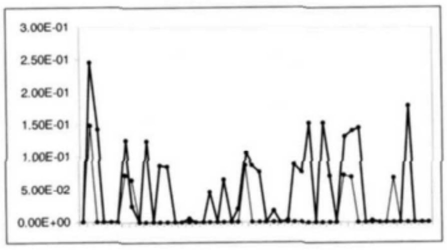

Fig. 19. Result using FLA and incomplete training parameters

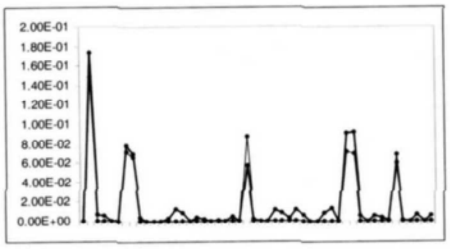

Fig. 20. Result using SNN

parameters used, but the input has more than one dominant value. The smallest deviation is obtained by the FLA. The result of the SNN is much improved over the standard approach. **Figures 10-18** show results where the input-output parameters come from "new" documents (unknown to the system). Note that the fuzzy algorithm has generally less noise in estimation.

As mentioned, the fuzzy algorithm would result in a much improved estimation, increasing the number of antecedent sets, if all pieces of contribution functions (Fig. 2) were trained. We tried to find intervals that were absolutely not reached by any training parameters. To test the estimation, we used a new document where most of the frequency measure values were in these intervals(**Fig.7**). We added extra training parameters to tune untrained pieces of functions (**Figures 19-21**). Figures 4-21, respectively

7. Conclusion

This paper introduced a FLA and its special case of SNN algorithms. The more fuzzy terms in each universe used in the FLA, the more improved the estimation. More special training parameters and calculation time are required. Using a SNN that is a special case of the FLA, namely, where two fuzzy terms are used in each input universe, does not require specially selected training parameters, but its result is not significantly different from the FLA, where the number of input terms is larger than two in the sense of word-frequency estimation. It was shown that the SNN algorithm is simpler and requires considerably less computation effort than the standard neural network implementations, but the result is significantly improved.

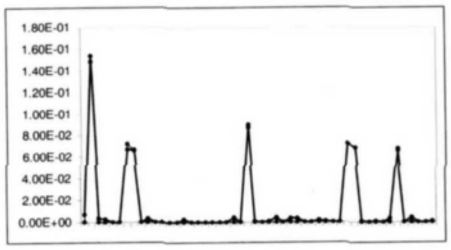

Fig. 21. Result using FLA and improved training parameters

References:

1) Gedeon, T.D. and Ngu, A.H.H. "Index generation is better than extraction," Proc. of NOLTA'93 Int. Conf. Non-Linear Theory and Applications, 771-777, (1993).
2) Bustos, R.A., Gedeon, T.D., "Learning Synonyms and Related Concepts in Document Collections," in Alspector, J., Goodmanf, R. And Brown, T.X. Applications of Neural Networks to Telecommunications 2, Lawrence Erlbaum, 202-209, (1995).
3) Kóczy, L.T., Gedeon, T.D., "Information retrieval by fuzzy relations and hierarchical cooccurence," IETR 97/3 Dept. Information Engineering School of Comp. Science and Ing. Univ. of NSW. TR 97/3 Sydney, (1993).
4) M. Mizumoto, ''Fuzzy contols by product-sum-gravity method,'' Advancement of Fuzzy Theory and Systems in China and Japan, Eds. Liu and Mizumoto, International Academic Publishers, c1.1-c1.4, (1990).
5) Gedeon, T.D. and Mital, V., "Information Retrieval Using a Neural Network Integrated with Hypertext," Proc. of Int. Conf. Neural Networks, 1819-1824, (1991).
6) Rose D. and Belew R., ''A Connectionist and symbolic Hybrid for Improving Research," Int. J. Man Machine Studies v. 35(1991)
7) Blair D.C., ''Language and Representation in Information Retrieval,'' Amsterdam, Elsevier, (1990).
8) Blair, D.C. and Marron, M.E., "An Evaluation of Retrieval Effectiveness for a Full-Text Document Retrieval System," CACM, **28**-3, 289-299, (1985).
9) Paice, C.D., "Constructing Literature Abstracts by Computer: Techniques and Prospects," Info. Proc. and Management, **26**-1, 171-186, (1990).
10) Fischer G. and Stevens C., "Information access in complex, poorly structured information spaces," Proc of CHI'91 Conference, (1991).
11) Foltz, P.W. and Dumais, S.T., "Personalized information delivery: an analysis of information filtering methods," CACM, **35**-12, 51-60, (1992).
12) Goldberg, D., Nichols, D., Oki, B.M. and Terry, D., "Using collaborative filtering to weave an information tapestry," CACM, **35**-12, 61-70, (1992).
13) Brookes, C., ''grapeVINE: Concepts and Applications,'' Office Express Pty. Ltd.(1991).
14) Salton, G., ''The SMART Retrieval System — Experiment in Automatic Document Processing,'' Englewood Cliffs, Prentice-Hall, (1971).
15) Führ, N. and Pfeifer, U., "Combining Model-Oriented and Description-Oriented Approaches for Probabilistic Indexing," Proc. of 14th International ACM/SIGIR Conference on Research and Development in Information Retrieval, 46-56, (1991).
16) Turtle, H., "Text Retrieval in the Legal World," Artificial Intelligence and the Law, **3**, 97-142, (1995).
17) Salton, G., ''Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer,'' Addison Wesley, (1989).

Name:
Péter Baranyi

Affiliation:
Ph.D. Student at the Technical University of Budapest and Computer and Automation Institute Hungarian Academy of Sciences

Address:
Dept. of Automation Technical University of Budapest, Budafoki u. 8, Budapest, H-1111, Hungary

Brief Biographical History:
1994- M.Sc. in Electrical Engineering, Technical University of Budapest
1995- M.Sc. "Teacher of Engineering Sciences," Technical University of Budapest
1996- Research assistant at the Chinese University of Hong Kong
1997- Research assistant at the University of New South Wales, Sydney, Australia

Main Works:
- P. Baranyi, T.D. Gedeon, L. T. Kóczy, "A General Interpolation Technique in Fuzzy Rule Bases with arbitrary Membership Functions," IEEE Int. Conf. System Man and Cybernetics (IEEE SMC'96), pp.510-515, Beijing, China, (1996).
- P. Baranyi and Yeung Yam, "Singular Value-Base Approximation with Non-Singleton Fuzzy Base," 7th Int. Fuzzy Systems Association World Congress (IFSA'97), pp.127-132, Prague, (1997).
- P. Baranyi, Iko Bavelaar, L. T. Kóczy and A. Titli, "Inverse Rule Base of Various Nonlinear interpolation techniques," 7th Int. Fuzzy Systems Association World Congress (IFSA'97), pp.121-126, Prague, (1997).

Membership in Learned Societies:
- Hungarian Society of IFSA (International Fuzzy Systems Association)
- Hungarian Neumann János Computer Science Association
- Hungarian Electrotechnic Association
- Hungarian Energetic Association

Name:
László T. Kóczy

Affiliation:
Professor of Telecommunications and Telematics

Address:
Technical University of Budapest
Department of Telecommunications and Telematics
Budapest, Sztoczek u. 2. 1111, Hungary

Brief Biographical History:
1975- M.Sc. in Electrical Engineering
1976- M.Phil. in Control Engineering
1977- Ph.D. in Engineering
1998- D.Sc. In Engineering

Main Works:
- Theory Fuzzy Control and Fuzzy Logic

Membership in Learned Societies:
- IEEE SMC Society
- International Fuzzy Systems Association

Name:
Tamás Domonkos Gedeon

Affiliation:
Professor. Department of Information Engineering, School of Computer Science and Engineering, The University of New South Wales

Address:
Sydney 2052. Australia

Brief Biographical History:
I did my undergraduate and PhD degrees at the University of Western Australia. After a few years at Brunel University, I took a position at the University of New South Wales, where I am now Head of the Department of Information Engineering. My research interests are mainly in neural and fuzzy techniques, and their application to legal information retrieval and petroleum engineering problems.

Main Works:
- Gedeon, TD, "Data Mining of Inputs: Analysing Magnitude and Functional Measures," International Journal of Neural Systems, **8**-2, 209-218, (1997).
- Gedeon, TD, "The Cyclic Towers of Hanoi: An iterative solution produced by transformation," The Computer Journal, **39**-4, 353-356, (1996).
- Gedeon, TD and Koczy, LT, "Conservation of fuzziness in rule interpolation," International Symposium on New Trends in Control of Large Scale Systems, pp.13-19, Herlany, (1996).
- Gedeon, TD, Wong, PM and Harris, D, "Balancing Bias and Variance: Network Topology and Pattern Set Reduction Techniques," in Mira, J and Sandoval, F, (eds.), From Natural to Artificial Neural Computation, pp. 551-558, Springer- Verlag, LNCS, vol.930, (1995).

Membership in Learned Societies:
- Assoc. Computing Machinery (ACM)
- Internat. Neural Net. Soc. (INNS)
- Aust. Comp. Soc. (MACS)