Specialised Neural Network for Learning Word Frequency Measures in Large Document Collections

P. Baranyi [1, 2, 3, 5]
P. Aradi [4,5]
L.T. Kóczy [3, 5]
T.D. Gedeon [5]

[1] Computer and Automation Institute
Hungarian Academy of Sciences

[2] Dept. of Automation
[3] Dept. of Telecommunication and Telematics
[4] Dept. of Systems and Control Engineering
Technical University of Budapest
Mûegyetem rkp. 3., Budapest, H-1521 Hungary

[5] Dept. of Information Engineering
School of Computer Science and Engineering
The University of New South Wales
Sydney, 2052 Australia

Keywords: Neural Network

Abstract

For very large document collections or high volume streams of documents, finding relevant documents is a major information filtering problem. One of the main types of information retrieval systems produces a word frequency measure estimated by some important parts of the document using neural network approaches. This paper reports a new network structure for this task. It is specialised considering the main difficulties of these kinds of applications, namely, the calculation time complexity. It will be pointed out that the calculation, hence, the learning time is much reduced applying the new algorithm, however, the result is significantly improved compared to the former approaches, which offer a possibility to increase the number of considered words, hence, improve the

effectiveness of information filtering systems.

1. Introduction

An information system allows users to efficiently retrieve documents from a very large collection, that are relevant to their current interests. The information retrieval system has to determine the relevant documents by using the matching of key words or phrases of user interest specified in the query, that are fragments of natural language texts, and the words and phrases used in documents of the collection. The main difficulty is that an effective search for matching has to consider the whole topic determined by the queried words and their synonyms, or else relevant documents are lost. If i.e. only synonyms are used in the query, very relevant documents might be completely left out of consideration. In order to alleviate this problem many research works have emerged on the field of creating and maintaining information filters, specifying categories, building synonym lists, and so on. In these systems it is inevitable to deal with the problem of automatic indexing [1,2]. The aim of indexing text items is to (implicitly) summarise their content. Once important keywords are found and their occurrence frequencies are known or estimated, it is possible to build up co-occurrence maps, hierarchical co-occurrence relations, etc. [3]

The proposed automatic indexing algorithms are based on the combination of some significant measures such as frequency-keyword approach, title-keyword, location-method, cue-method, etc. [2]. One of the main ideas to determine the most appropriate way of combining the indexing parameters is based on neural network approaches that can learn a composition function of significant measures. One important part of these algorithms is to train a neural network to produce the word frequency measure component of a retrieval index. The network uses a relatively small proportion of the document texts as its input (e.g. the first and last 20%). Further, all of these measures can be calculated locally and do not require global document collection information as it is required for the frequency-keyword calculations. This may provide efficiency gains in the future in highly parallel implementation.

Thus, the efficiency of information retrieval significantly depends on the effectiveness of the frequency-keyword measure, hence, the applied neural network approaches. In this paper a neural network based algorithm will be discussed for this task.

The advantage of applying neural networks 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 standardised 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. Moreover, to achieve a good approximation, some approaches may overly assign the number of hidden units and layers, thereby resulting in a large neural network causing problems in computational time complexity. Applying learning algorithms to a large number of units in a neural network or to a difficult function included in the neurons needs considerable calculation time even in small applications. As it is mentioned above, the problem is that the collection of documents where the

selected ones have to be retrieved from might be extremely large. Thus, two important aims have to be taken into consideration to generate a neural representation. One is to achieve a sufficient estimation of the frequency of considered words including their co-occurrence relations. The other is to use simplified functions in the neurons and to reduce the number of layers and neurons as much as possible to reduce the required computational effort.

The former proposed neural network approaches for estimating the word frequency have the same neural network structure [1, 2]. In this paper another network structure is proposed that reduces the computational time complexity. The proposed network method has less units than the former ones, and it has simplified functions in the units rather than exponential function in [1, 2]. Further on, the learning time is also considerably reduced and the obtained results are significantly improved as compared to the former approaches.

This paper is organised as follows: Section 2 briefly describes the main concept of the former neural network application. Section 3 presents the proposed neural network concept. Section 4 evaluates the result by the proposed neural network in comparison with the former approaches. Section 5 gives a conclusion. Section 6 contains the acknowledgments.

2. Description of the former network concept [2].

This section briefly describes the concept used in the former neural network approaches to effectuate an easy understanding of the proposed new algorithm.

Characterisation of the network

Suppose that a feedforward network (Fig. 1), contains three (input, hidden and output) layers, and in each layers I, H and O neurons, respectively. The network input vector is $\underline{\mathbf{x}} = [x_i]$. 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 and of output neuron N_k^O is y_k. The connection weight matrix between neurons of the input and hidden layers is $\underline{\underline{\mathbf{W}}}^I = [w_{j,i}^I]$, between the hidden and output layers: $\underline{\underline{\mathbf{W}}}^H = [w_{k,j}^H]$. The hidden and output layers are biased, that means there is an additional neuron in both the input and hidden layers with a constant output of 1. The bias weight vector for the hidden layer is $\underline{\mathbf{w}}^{I,bias} = [w_j^{I,bias}]$, for the output layer $\underline{\mathbf{w}}^{H,bias} = [w_k^{H,bias}]$. The input values of neuron N_j^H $(j = 1,\ldots,H)$ in the hidden layer are $x_{j,i}^H = y_i^I w_{j,i}^I$, and from the bias neuron $x_{j,0}^H = 1 \cdot w_j^{I,bias}$. Similarly, the input values of neuron N_k^O in the output layer are $x_{k,j}^O = y_j^H w_{k,j}^H$, and from the bias neuron $x_{k,0}^O = 1 \cdot w_k^{H,bias}$.

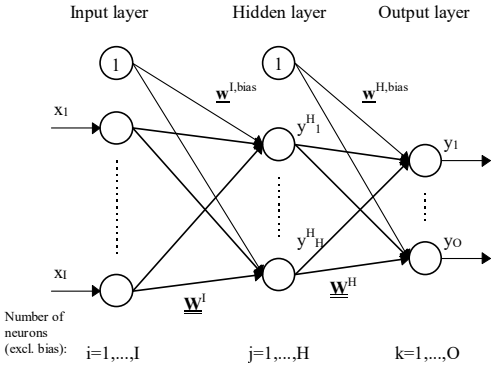

Fig. 1 Network layout

The hidden and output neurons can be divided into two main functional parts (Fig. 2). The first part contains the summarisation of input values as

$$y' = x_0 + \sum_{i=1}^{n} x_i = 1 \cdot w^{bias} + \sum_{i=1}^{n} w_i x_i$$

The second part is the transfer function

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

The output of the neuron is

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

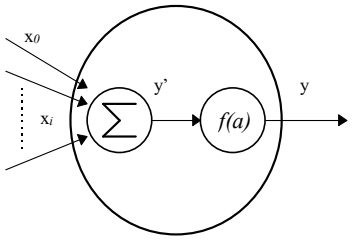

Fig. 2 Functional units of a single neuron

Thus, for the hidden layer

$$\underline{\mathbf{y}'}^{H} = \left[\underline{\underline{\mathbf{W}}}^{I} \quad \underline{\mathbf{w}}^{I,bias}\right]\begin{bmatrix}\underline{\mathbf{y}}^{I} \\ 1\end{bmatrix}$$

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

Similarly, for the output layer

$$\underline{\mathbf{y}'}^{O} = \left[\underline{\underline{\mathbf{W}}}^{H} \quad \underline{\mathbf{w}}^{H,bias}\right]\begin{bmatrix}\underline{\mathbf{y}}^{H} \\ 1\end{bmatrix}$$

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

Let us take the last improved network from [2]. The selection of the words and training parameters were as follows [1]: A collection of 352 documents (sections of a legal textbook) comprising a total of some 1901 different words was chosen. By the removal of stop words and stemming, the number of words was reduced. Words occurring only once or twice in the whole document collection were removed. The resulting number of words was still too large to input into a supervised training neural network. Each word occurring only once or in more than half of the documents was removed. So, finally 75 words were listed. These words are used to index 350 documents in the overall collection. The chosen neural network had three inputs for each of the 75 words. These were for the title-keywords, location- and cue-frequencies.

The title lines of the documents were scanned to determine the frequency of occurrence of the 75 words, and the resulting vector of values was normalised to 1. This accounted for multiple occurrences of words in titles (rare), and if there is a large number of words in the title, the significance of each of these is less.

The location measure was calculated using only the first and last 20% of each document, and a normalised frequency measure was produced. Similarly, for the cue method, a window of 5 words on both sides of all words examined for the presence of cue words, the frequency of proximity to cue words was then normalised as above.

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

Characterisation of input and output

The input-output structure used in the neural network application was as follows: $n_w = 75$ words were considered each with $n_p = 3$ input parameters $x^0_{i_w,i_p} \in [0,1]$, such as title-keywords, location keyword- and cue keyword measures. Thus, the total number of input neurons was $I = n_w n_p = 225$. The

output values $y_k \in [0,1]$ $(k = 1,\ldots,n_w)$ were the estimated frequency-keyword measures for each word. The number of output neurons was $O = 75$. The number of hidden neurons was $H = 4$.
Suppose that the *t*-th training data pairs were:

$$\underline{\mathbf{x}}^t = \left[x_1^t \ldots x_s^t \ldots x_I^t\right], \tag{1}$$

where $x_s^t = x_{(i_w-1)n_w+i_p}^t = x_{i_w,i_p}^0$, and
$i_w = 1,\ldots,n_w$, $i_p = 1,\ldots,n_p$, $t = 1,\ldots,n_t$ and

$$\underline{\mathbf{y}}^t = \left[y_1^t \ldots y_O^t\right]$$

as the desired output.

The steps of forward propagation were as follows.

Let $\underline{\tilde{\mathbf{y}}}$ be the output calculated by the network as:

1. $\underline{\mathbf{y}'}^H = \left[\underline{\underline{\mathbf{W}}}^I \ \underline{\mathbf{w}}^{I,bias}\right]\begin{bmatrix}\underline{\mathbf{y}}^I \\ 1\end{bmatrix}$ (2)
2. $y_j^H = f\left(y_j'^H\right) = \dfrac{1}{1+e^{-y_j'^H}}$,
3. $\underline{\mathbf{y}'} = \left[\underline{\underline{\mathbf{W}}}^H \ \underline{\mathbf{w}}^{H,bias}\right]\begin{bmatrix}\underline{\mathbf{y}}^H \\ 1\end{bmatrix}$
4. $\tilde{y}_k = f\left(y_k'\right) = \dfrac{1}{1+e^{-y_k'}}$

The error is

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

Let the error between the two parts of neuron N_k be δ_k' and the error at the output of this neuron be δ_k. For the neurons in the output layer the error vectors are $\underline{\boldsymbol{\delta}'}^O$ and $\underline{\boldsymbol{\delta}}^O$, respectively. Thus, based on error back propagation

6. $\delta_k'^O = \tilde{y}_k\left(1-\tilde{y}_k\right)\delta_k^O$

7. $\underline{\boldsymbol{\delta}}^H = \left[\underline{\underline{\mathbf{W}}}^H\right]^{\mathrm{T}}\left[\underline{\boldsymbol{\delta}'}^O\right]$

8. $\delta_j'^H = y_j^H\left(1-y_j^H\right)\delta_j^H$

9. $\Delta\left[\underline{\underline{\mathbf{W}}}^H \ \underline{\mathbf{w}}^{H,bias}\right] = \eta\left[\underline{\boldsymbol{\delta}'}^O\right]\left[\underline{\mathbf{y}}^H\right]^{\mathrm{T}}$

10. $\Delta\left[\underline{\underline{\mathbf{W}}}^I \ \underline{\mathbf{w}}^{I,bias}\right] = \eta\left[\underline{\boldsymbol{\delta}'}^H\right]\left[\underline{\mathbf{y}}^I\right]^{\mathrm{T}}$

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. The modified neural network concept

In order to reduce the calculation time the second part of the neurons which is an exponential function relatively requiring much computation was omitted. Further, if an input value is 0 then it does not have any contribution in the result. In this, the bias weights were taken out of consideration, as the value of bias inputs was set to 0 instead of 1. This is possible as the significance of co-occurence of words is directly coupled, hence constraining the network hyperplanes to pass through the origin is not problematic. As a matter of fact, in this case the hidden layer does not have any role and the output is calculated without steps 3 and 5 and with bias inputs set to 0:

$$\underline{\tilde{\mathbf{y}}} = \underline{\underline{\mathbf{W}}}^H \underline{\underline{\mathbf{W}}}^I \underline{\mathbf{x}} = \underline{\underline{\mathbf{W}}}\underline{\mathbf{x}} \tag{3}$$

However, the hidden layer can be used for further connection, hence, computational complexity reduction.

The size of $\underline{\underline{\mathbf{W}}}^{I}$ is $I \times H$, the size of $\underline{\underline{\mathbf{W}}}^{H}$ is $H \times O$. So, the total number of connections is $I \cdot H + H \cdot O$. If only $\underline{\underline{\mathbf{W}}}$, the size of which is $I \times O$ is used the total number of connections is $I \cdot O$.
Thus, if

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

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

Determination of the number of hidden units

A two layer network with connection matrix $\underline{\underline{\mathbf{W}}}$ is used to learn the training parameters. Singular value decomposition (SVD) is applied to choose the optimal H in the sense of connection complexity reduction. Applying SVD on $\underline{\underline{\mathbf{W}}}$ results in

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

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

$$\underline{\underline{\mathbf{U}}} = \left| \underline{\underline{\mathbf{U}}}_{r} \quad \underline{\underline{\mathbf{U}}}_{d} \right|$$

$$\underline{\underline{\mathbf{V}}} = \left| \underline{\underline{\mathbf{V}}}_{r} \quad \underline{\underline{\mathbf{V}}}_{d} \right|$$

$$\underline{\underline{\mathbf{D}}} = \begin{vmatrix} \underline{\underline{\mathbf{D}}}_{r} & \underline{\underline{\mathbf{O}}}_{n_r \times (n_v - n_r)} \\ \underline{\underline{\mathbf{O}}}_{(n_u - n_r) \times n_r} & \underline{\underline{\mathbf{D}}}_{d} \end{vmatrix}$$

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

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

The approximation is exact if all non zero singular values are kept as

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

Consequently, if $n_r < \frac{I \cdot O}{I + O}$ then let

$$\underline{\underline{\mathbf{W}}}^{I} = \underline{\underline{\mathbf{U}}}_{r} \text{ and } \underline{\underline{\mathbf{W}}}^{H} = \underline{\underline{\mathbf{D}}}_{r}\,\underline{\underline{\mathbf{V}}}_{r}$$

or obviously

$$\underline{\underline{\mathbf{W}}}^{I} = \underline{\underline{\mathbf{U}}}_{r}\,\underline{\underline{\mathbf{D}}}_{r} \text{ and } \underline{\underline{\mathbf{W}}}^{H} = \underline{\underline{\mathbf{V}}}_{r}$$

If n_r is not less than $\frac{I \cdot O}{I + O}$, non-zero singular values can be discarded. In this case the error bound of SVD reduction has to be considered.

Reduction error bound if non-zero singular values are discarded

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

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

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

$$\left|\underline{\underline{\mathbf{W}}} - \underline{\underline{\widetilde{\mathbf{W}}}}\right| = \left|\underline{\underline{\mathbf{U}}}_d\right| \left|\underline{\underline{\mathbf{D}}}_d\right| \left|\underline{\underline{\mathbf{V}}}_d^{\mathrm{T}}\right| \le 1_{n_u \times (n_u - n_r)} \underline{\underline{\mathbf{D}}}_d 1_{(n_v - n_r) \times n_v}$$

$$1_{n_u \times (n_u - n_r)} \underline{\underline{\mathbf{D}}}_d 1_{(n_v - n_r) \times n_v} = \left(\sum_{i=n_r+1}^{n_{SVD}} \sigma_i \right) 1_{n_u \times n_v} \qquad (4)$$

It is not necessary to take the absolute value of $\underline{\underline{\mathbf{D}}}_d$, as each singular value is positive. Equation (4) implies

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

Thus, the error between the output of the original network $\underline{\mathbf{y}}$ and the output of the reduced network $\underline{\mathbf{y}}^r$ is

$$\left| y_i - \widetilde{y}_i \right| \le \sum_{i=1}^{I} dx_i$$

where I is the number of inputs.

The learning phase of the application results in a strongly non-singular matrix $\underline{\underline{\mathbf{W}}}$. Discarding only one singular value results in a large error. (In our case $\frac{I \cdot O}{I + O} = 56.25$, that means at least 19 singular values have to be discarded for reduction.) This is quite obvious from the behaviour of the training parameters. Suppose that the network has one hidden layer, where the number of neurons is H.

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

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

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

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

In the same way it can be seen that the number of neurons in the hidden layers (4 in this example) applied in the former network structures [1, 2] is also not enough. It should have been close to 75. The neurons in the former networks contained exponential functions, and applying 75 neurons in the hidden layer instead of 4 would have resulted in a considerable increase of calculation time.

Note that this explains some of the improvements we find even though our new structure can only be applicable to problems which have a linear separable input space – that is, previously with only 4 hidden neurons our networks were clearly capable of producing an internal representation which is in some sense deficient in comparison with a good linear separation. Clearly, this improvement is problem specific, and could not be assumed to work for all problems.

Let us assume the algorithm of the proposed network.

1. $\underline{\tilde{\mathbf{y}}} = \underline{\underline{\mathbf{W}}}\underline{\mathbf{x}}$

The learning phase is then:

2. $\underline{\boldsymbol{\delta}} = \underline{\mathbf{y}} - \underline{\tilde{\mathbf{y}}}$
3. $\Delta\underline{\underline{\mathbf{W}}} = \eta[\underline{\boldsymbol{\delta}}][\underline{\mathbf{x}}]^{\mathrm{T}}$

It can be seen that the proposed method is much simplified compared to (2). Negative word frequency measurements are not interpretable. As the new method may result in negative values the output must be bounded below.

4. Experiments and results

As it was shown the new neural network is more simplified than the former network, hence, the learning time was much less than in former cases. Further, the simplified method require 400-500 epochs training to converge, unlike the former method that uses 1000 epochs.

The same experiments have been conducted using the proposed .neural network as by using the former network. The results for a randomly chosen document by the former and the proposed networks are shown in Figures 3 and 4.

Note that we would like to show the results for all documents in the collection, for example, the average measures of the all documents, as the whole performance cannot be ascertained from the results for only one document. However, for frequency measures the average (which is constant) looses all the distinctive information as it is the appropriate deviations from the average which are significant. Even measures of deviation are not useful as it is possible to produce values with the right range of deviations but be useless in practice. The only real measure is in terms of user satisfaction with the documents retrieved by an information retrieval system which used frequency measures produced by our technique. This analysis is beyond the scope of this paper, hence we are reduced to providing a randomply chosen example, and stating that the results are representative of the results for the population of documents.

These figures show the result calculated by the new neural network system and the old network system, respectively. The horizontal axis means the set of considered words, while the vertical axis contains the frequency-keyword measure.

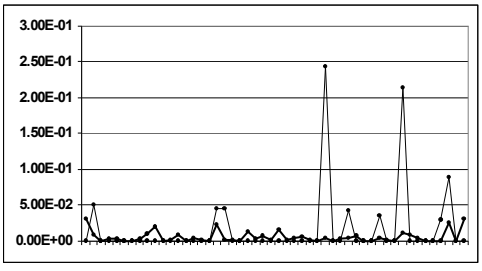

Fig. 3 Result by the original neural network

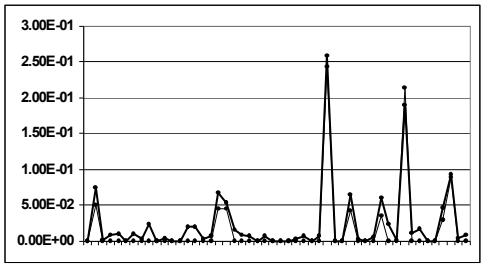

Fig. 4 Result by the proposed neural network

To see the difference between the results, the points on the 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 the estimated measures.
It can be seen in these figures that the estimations by the new neural network are much improved compared to the results by the former method.

5. Conclusions

In this paper a new method to estimate the frequency keyword measure is reported, using a simplified neural network implementation. It was shown this algorithm is much simplified and requires considerably less computation effort than the former proposed neural networks, however, the result is significantly improved from the former ones.

6. Acknowledgements

This work was supported by the Australian Research Council, Grant No. A49600961.

References

[1] Gedeon, T.D. and Ngu, A.H.H. "Index Generation is better than Extraction," Proceedings of the International Conference on Non-Linear Theory and Applications, Hawaii, 1993, pp. 771-774.

[2] Bustos, R.A., Gedeon, T.D., "Learning Synonyms and related Concepts in Document Collections" in Alspector, J., Goodman, R. And Brown, T.X. Applications of neural Networks to Telecommunications 2, Lawrence Erlbaum, 1995, pp. 202-209.

[3] Kóczy, L.T., Gedeon, T.D., "Information retrieval by fuzzy relations and hierarchical co-occurence" Technical Report, Dept. of Information Engineering, SCSE, UNSW, 1997.