AN APPROACH FOR EFFICIENT RANKING OF XML DOCUMENTS USING USING BPN BASED RANN

Mary Posonia A\textsuperscript{1}\textsuperscript{*}, Vigneshwari S\textsuperscript{1}, Jyothi V.L.\textsuperscript{2}

\textsuperscript{1}Dept. of Computer Science & Engineering, Sathyabama Institute of Science & Technology, Tamilnadu, INDIA
\textsuperscript{2}Dept. of Computer Science & Engineering, Jeppiaar Engineering College, Tamilnadu, INDIA

ABSTRACT
There is a semantic gap between the implications of the keywords in the recovered documents and the implications of the terms utilized as a part of users' queries. Ranking algorithms are an important step in search engines so that the user could retrieve the pages most relevant to the query. The proposed novel algorithm, Back Propagation Network (BPN) based Ranking algorithm using Neural Networks (RANN) system is used to rank the XML documents, both in a time efficient and cost efficient manner. The existing XML based document indexing approaches focus only on partial input queries which lead to irrelevancy problem. To overcome this issue, efficient document retrieval is achieved with the help of BPN trained RANN and it is proven in the results. The overall performance of RANN is measured in comparison with Vector Space Model (VSM). The results of the devised approach show remarkable improvement in the performance with the use of synthetic records and benchmark dataset with an overall improvement of 7% raise in Precision, Recall and F-Measure rates when compared to the existing VSM based approach.

INTRODUCTION

In the usual web based information retrieval frameworks, the user's needs are not met as they are ranked in view of the conventional string matching approach of the user's query. This has led to a semantic gap between the implications of the keywords in the recovered records and the implications of the terms utilized as a part of user’s queries. With the rapid growth of the World Wide Web there comes the need for a fast and accurate way to retrieve the information required, which is made possible with the help of Search engines. Ranking algorithms are essential for the users to retrieve the pages that are most relevant to the query. Information Retrieval (IR) for XML has increased noteworthy consideration and has rose as one of the research subjects that have been examined by Keyword researchers and Query researchers. The objective of this research is to apply the IR utilizing the Ranking Algorithm of Neural Network (RANN) Model on characteristic XML documents to solve the issues in document retrieval. In this paper, IR using RANN is applied for document ranking and retrieval.

This paper is organized as follows: related work, followed by materials and methods discussing about the implementation of BPN based RANN model which is followed by the evaluation of results and conclusion.

RELATED WORK

XML ranking can be supported through the “inverted element, frequency” and “weighted term frequency” as proposed by Chen et al [1]. Here, the weight of the term is dependent on the location and frequency in an XML element, and its popularity is also known among the similar elements in an XML dataset. Chen’s idea is followed in this paper to find the term-document index of XML documents.

A delay may occur before an information extraction is available based on the updated documents. Shortening of the delay as proposed in [2-6], using a method which recycles the intermediate results of the snapshots taken in the past.

A Vector Space Model was proposed in [7-12] which was known as the orthogonal factorization matrix, and could be used for the retrieval of information from a large database. The VSM model has been used for comparison with the proposed model in our BPN based RANN model.

The authors in [13-20,23,24] presented an information retrieval technique using Vector Space Model (VSM). Firstly, the similarity scores are computed through the use of a weighted average for every item. Later, the cosine measure helps to compute the similarity measure and determines the document vector and the query vector angle, on the basis of geometry. The use of IR and Neural Network (NN) improved the performance of the information retrieval. However, there is a drawback with respect to the IR systems, in conjunction with natural languages, especially of Arabic type. Hence, to overcome those drawbacks a feed forward training Network like BPN is necessary for yielding accuracy which has been proposed in our current paper.

A hybrid model and its application were presented by Karegowda et al. [21] and Shanthi et al.[22]. Here, the context of Artificial Neural Networks on the subject of Evolving Connection Weights was applied to the prediction of stroke disease; a comparison was made between the desired and real output of the Hybrid ANN-GA and ANN. The accuracy in classification with respect to the surfaces was found to be improved in
this case. Hence it is inferred from the literature survey that the results of experiments in the existing literature indicate an improvement in the performance upon the utilization of Neural Networks.

MATERIALS AND METHODS

General neural network architecture for document ranking

Data retrieval utilizing the RANN Model for XML Document is a strategy to acquire significant measures between a query and the documents recovered. The model consists of three layers; Query Terms Layer, Documents Terms Layer and Documents Layer. Cosine similarity measure is utilized as a part of the RANN model to ascertain the similarity between document query vectors.

Enhanced RANN based information retrieval system

The Back Propagation Network (BPN) follows the delta learning rule of Neural Networks in order to reduce the error by weight adjustments in the hidden and output layers. BPN is preferred in the current paper since the sigmoid function of BPN deals with non-linear models like XML tree structures [Fig. 1].

The input interface is added, which allows the user to enter a query. The information retrieval system is then upgraded which empowers the user to locate the significant documents. Towards the end, an output interface is created, which sorts the significant documents and sends them to the user as a result.

When it is necessary to give a query input involving at least two words, it is important to include more input neuron bunches into the first neural network where each gathering depicts a single word. At that point each word is independently mapped cognizant to the keywords of each query.
Detailed BPN based RAAN model

Fig. 2: Detailed BPN based RANN model

[Fig. 2] describes the detailed framework for document retrieval based on BPN algorithm. Here $q_i$ represents the set of input queries $i = 1$ to $n$, $h_{id}$ represents the $j$th node of the $i$th hidden layer, $i = 1$ to $m$ and $j = k$ to $m$. For binary inputs a threshold of 0.5 is set. If the tf-idf measure is greater than 0.5 then the binary input is 1 and 0 otherwise.

And, $w_{tk}$ represents the hidden output weights for ‘o’ output documents. The resultant documents are Trained Document (TD) indexed ones. The TD index is calculated based on the precision rate of the documents which are all True-Positive (i.e based on relevancy in the retrieved documents). For calculating query document similarity cosine similarity is used, for ranking and for Index, B+ Tree algorithm is used [15].

\[
TP = \frac{TP}{TP + FP}
\]

Where TP represents True Positiveness and FP represents False Positiveness of the relevant retrieved documents.

BPN based RANN algorithm

Input : ‘n’ number of input queries
‘m’ number of hidden nodes in each hidden layer k
‘o’ number of retrieved documents as output layer

Output: TD indexed documents
Read ‘n’ number of input nodes
Read ‘h’ number of hidden nodes
Read ‘m’ number of output nodes

Step 1: Read the input vector of queries $q_i$
Step 2: Read the output vector $t_{od}$ (Desired output documents with a set of pre-trained documents from the training knowledge base)

Step 3: Read the input hidden weights $q_{dw}$, where $q_{dw}$ is the query document weight which calculated based on the tf-idf measure.
Step 4: Read the output hidden weights $w_{tk}$
Step 5: Calculate netvalh$_{jk}$ (net value in hidden layer)
\[
netvalh_{jk} = \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} \cdot h_{id} \quad \forall \ k = 1 \text{ to } l
\]
Step 6: Calculate the $f(netvalh_{jk})$ : ‘netvalh$_{jk}$’ in hidden layer (sigmoidal function)
\[
f(netvalh_{jk}) = \frac{1}{(1 + e^{-f(netvalh_{jk})})}
\]
Step 7: Calculate the net td-output: “$t_{do}$”
\[
t_{do} = \sum_{i=1}^{n} \sum_{j=1}^{m} t_{di} \cdot h_{id} \quad \forall \ k = 1 \text{ to } o
\]
Step 8: Calculate the actual output $a_{o}$
Step 9: Calculate error in output layer \(e_{o}\) which is the difference between Desired output and Actual output (Delta rule)

\[ e_{o} = \frac{1}{2} (y_{o} - d_{o})^{2} \]

Step 10: Calculate error in hidden layer \(e_{hid_{jko}}\)

\[ err_{hid_{jko}} = \sum_{k=1}^{n} t_{d_{o} \cdot hid_{jko}} \forall o = 1 \text{ to } o \text{ and } j = 1 \text{ to } m \]

Step 11: Calculate the adjusted weights \((nwo_{jko})\) for the hidden output layer \(nwo_{jko}\)

For \(j=1\) to \(h\)

For \(k=1\) to \(m\)

\[ nwt_{jko} = wt_{jko} + (\eta \cdot t_{d_{o}} \cdot netv_{alh_{jko}}) \]

Step 12: Adjusted weight for input hidden layer is calculated as follows: \(awh_{ijk}\)

\[ awh_{ijk} = wt_{ijk} + (\eta \cdot q_{i} \cdot netv_{alh_{ijk}}) \]

The new weight obtained for hidden output layer is \(nwt_{jko}\) and the new weight obtained for input hidden layer is \(awh_{ijk}\)

Step 13: The earlier weights are replaced with the adjusted weights in both the hidden-input and output hidden layers and Step 5 to Step 13 are continued until saturation is reached.

Let \(q_i\) be the input query. Based on the similarity of query with the documents the input values are designed based on Cosine similarity.

\[ CosRann(doc, query) = \frac{\sum_{doc,i} \times query_{j}}{\sum_{doc,i} + \sum_{query,j}} \] (2)

The input weights are determined with the tf-idf value of the input queries with that of the documents mapped.

\[ qdw_{ijk} = TermFrequency_{i} \times InverseDocumentFrequency_{i} \] (3)

\[ TermFrequency_{i} = \frac{\text{Number of Document Terms which are similar}}{\text{Total number of Document Terms}} \] (4)

\[ InverseDocumentFrequency_{i} = \log \left( \frac{\text{Number of documents}}{\text{DocumentFrequency}_{i}} \right) \] (5)

The hidden weight calculation is based on the approximated weights based on the input queries. The documents are trained and the trained document index is calculated as TD index (Equation (5.1)).

Based on the TD index, the documents are sorted based on B+ Tree algorithm[17] and ranked based on the precision value[25]. The time complexity of the devised BPN based RANN algorithm is \(O(qdw^{k})\) where \(qd\) is the query document weight for \(k\) number of hidden layers. Total of 10 sample queries were utilized with total 1800 documents presented in the [Table 1].

<table>
<thead>
<tr>
<th>Queries</th>
<th>True Positiveness</th>
<th>False Negativeness</th>
<th>False Positiveness</th>
<th>False Negativeness</th>
<th>TD index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>980</td>
<td>890</td>
<td>51</td>
<td>43</td>
<td>0.95</td>
</tr>
<tr>
<td>Q2</td>
<td>999</td>
<td>865</td>
<td>88</td>
<td>87</td>
<td>0.92</td>
</tr>
<tr>
<td>Q3</td>
<td>945</td>
<td>903</td>
<td>243</td>
<td>117</td>
<td>0.80</td>
</tr>
<tr>
<td>Q4</td>
<td>1008</td>
<td>976</td>
<td>157</td>
<td>257</td>
<td>0.87</td>
</tr>
<tr>
<td>Q5</td>
<td>899</td>
<td>897</td>
<td>47</td>
<td>52</td>
<td>0.95</td>
</tr>
<tr>
<td>Q6</td>
<td>907</td>
<td>878</td>
<td>50</td>
<td>80</td>
<td>0.95</td>
</tr>
<tr>
<td>Q7</td>
<td>956</td>
<td>856</td>
<td>44</td>
<td>46</td>
<td>0.96</td>
</tr>
<tr>
<td>Q8</td>
<td>1013</td>
<td>834</td>
<td>66</td>
<td>68</td>
<td>0.94</td>
</tr>
<tr>
<td>Q9</td>
<td>989</td>
<td>912</td>
<td>145</td>
<td>120</td>
<td>0.87</td>
</tr>
<tr>
<td>Q10</td>
<td>976</td>
<td>908</td>
<td>67</td>
<td>66</td>
<td>0.94</td>
</tr>
</tbody>
</table>
Fig. 3: TD index comparison before training and after training

[Fig. 3] shows a significant improvement of the TD index after training with BPN based RANN with an average improvement of 2%.

Vector space model concepts overview

Vector Space Model (VSM) is a method used to interact with documents and queries as vectors in multidimensional space, whose measurements are the terms which are utilized to build an index to interact with the documents [7]. It is the most widely utilized procedure for information retrieval because of its effortlessness; effectiveness and pertinence over substantial document accumulations. The viability of the VSM depends generally on the term weighting connected to the term of the document vectors. The three phases of VSM are (VSM 2017):

- Document Term extraction
- Document Term weighting
- Ranking of documents based on the query-document similarity measure

<table>
<thead>
<tr>
<th>Table 2: Difference between VSM and BPN-RANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSM</td>
</tr>
<tr>
<td>Linear Model</td>
</tr>
<tr>
<td>Weights are not binary</td>
</tr>
<tr>
<td>Cosine similarity is followed</td>
</tr>
<tr>
<td>Allows partial matching</td>
</tr>
</tbody>
</table>

RESULTS

Comparison of precision, recall, F-measure between neural network model and vector model

This study was done with the following configuration with Windows 7 operating system, Intel Pentium(R) processor, CPU G2020 with processor speed of 2.90 GHz. The server pre-processes the data and stores it in the database. This process generates keywords, indexes XML documents and rank the documents based on the devised RANN algorithm. The datasets used for comparison is Sigmoid dataset which is freely downloadable from http://aiweb.cs.washington.edu/research/projects/xmltk/xmldata/data/sigmod-record/SigmodRecord.xml.

The performance of RANN is compared with the traditional VSM techniques, and the results prove that the retrieval rate and relevancy of documents using RANN is more efficient with that of VSM is shown in the [Table 3].

Table 3: Comparison of RANN and VSM for Precision, Recall and F-measure

<table>
<thead>
<tr>
<th>Model</th>
<th>Query</th>
<th>Number of Documents</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANN</td>
<td>Q1</td>
<td>150</td>
<td>89.84%</td>
<td>98%</td>
<td>93.74%</td>
</tr>
<tr>
<td></td>
<td>Q2</td>
<td>300</td>
<td>88.78%</td>
<td>97.20%</td>
<td>92.80%</td>
</tr>
<tr>
<td></td>
<td>Q3</td>
<td>450</td>
<td>85.04%</td>
<td>92.16%</td>
<td>88.46%</td>
</tr>
<tr>
<td></td>
<td>Q4</td>
<td>600</td>
<td>84.04%</td>
<td>92.16%</td>
<td>87.89%</td>
</tr>
<tr>
<td></td>
<td>Q5</td>
<td>750</td>
<td>86.04%</td>
<td>92.16%</td>
<td>88.99%</td>
</tr>
<tr>
<td>VSM</td>
<td>Q1</td>
<td>150</td>
<td>79%</td>
<td>85%</td>
<td>81.89%</td>
</tr>
<tr>
<td></td>
<td>Q2</td>
<td>300</td>
<td>76.00%</td>
<td>86.15%</td>
<td>80.76%</td>
</tr>
<tr>
<td></td>
<td>Q3</td>
<td>450</td>
<td>75.71%</td>
<td>87.50%</td>
<td>81.18%</td>
</tr>
<tr>
<td></td>
<td>Q4</td>
<td>600</td>
<td>75.71%</td>
<td>87.50%</td>
<td>81.18%</td>
</tr>
<tr>
<td></td>
<td>Q5</td>
<td>750</td>
<td>75.71%</td>
<td>87.50%</td>
<td>81.18%</td>
</tr>
</tbody>
</table>

Table 3, illustrates the Precision and Recall values of Ranking Algorithm of Neural Network and Vector Space Model keeps on decreasing as the total number of documents increases. The effect is caused by the keyword expansion from the ranking process. This causes the results to be error bounded which leads to inaccuracies in the user search. The system search is relevant even though the accuracy is not excelling. Also, the F-Measure values of Ranking Algorithm of Neural Network and Vector Space Model decreases when total number of documents increases. These issues are caused by the keyword expansion from the ranking process, thus giving results which are not accurate with respect to the user search but relevant to the systematic search. However, it is evident from [Table 3] that the Precision-recall and F value of Ranking Algorithm of Neural Network is higher when compared to the Vector Space Model. The results showed increased precision, recall, accuracy and F-measure rates and reduced response time and memory utilization with RANN is illustrated in [Table 4].

Table 4: Performance analysis of RANN compared with VSM and ORA in terms of Response Time and Accuracy

<table>
<thead>
<tr>
<th>Query</th>
<th>RANN</th>
<th>VSM</th>
<th>VSM</th>
<th>RANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>0.0061</td>
<td>0.00695</td>
<td>0.006919</td>
<td>0.006836</td>
</tr>
<tr>
<td>Q2</td>
<td>0.0054</td>
<td>0.0064</td>
<td>0.006133</td>
<td>0.006094</td>
</tr>
<tr>
<td>Q3</td>
<td>0.0071</td>
<td>0.00795</td>
<td>0.007802</td>
<td>0.00793</td>
</tr>
<tr>
<td>Q4</td>
<td>0.0064</td>
<td>0.0069</td>
<td>0.006422</td>
<td>0.005664</td>
</tr>
<tr>
<td>Q5</td>
<td>0.0074</td>
<td>0.0075</td>
<td>0.007043</td>
<td>0.00558</td>
</tr>
</tbody>
</table>

CONCLUSION

Urbanization In this paper, a novel algorithm called BPN based RANN algorithm for ranking the retrieved documents is introduced. It is a non linear model and therefore binary weights are allowed which is important for performing ranking on the dynamic incoming real time data. The proposed work has been compared with existing VSM model based on precision, recall and f-measure percentages and the results prove that the proposed algorithm shows an average of 7% improvement in the performance when compared with the existing VSM based approach.

CONFLICT OF INTEREST
The authors have no conflict of interest regarding this manuscript.

ACKNOWLEDGEMENTS
The authors would like to thank Sathyabama Institute of Science & Technology and its School of Computing for providing this great opportunity to do our research in a successful manner.
REFERENCES


