SOMLib: A Distributed Digital Library System based on Self-Organizing Maps

Andreas Rauber
Institut für Softwaretechnik, Technische Universität Wien
Resselgasse 3/188, A-1040 Wien, Austria
e-mail: rauber@ifs.tuwien.ac.at

Abstract

We describe an architecture for a distributed digital library system based on an unsupervised neural network model, namely the Self-Organizing Map. The system allows the clustering of text documents forming the basis for intelligent information retrieval. User profiles can be combined with full text queries or sample texts to locate documents within the library system. Contrary to conventional approaches consisting of one central library, the distributed SOMLib allows the combination and integration of several independent SOM-based library systems in a not necessarily hierarchical order. Furthermore, the system is not limited to a specific type or setup of the basic SOM architecture. Rather, variations of the basic architecture which might serve special needs of some users can be combined.

Keywords: Digital Libraries, Information Retrieval, Self-Organizing Map (SOM)

1 Introduction

The last years have seen an uninterrupted rise of the amount of information available in electronic form, with both the number as well as the size of such information collections continuing to grow. Similar to huge conventional libraries, which are interconnected by some organizational network, the combination of several independently managed information repositories must be facilitated, allowing to access and query a collection via one single interface. This requirement arises not only from the need for convenient interaction, but also from the very nature of the information repositories currently in use: Apart from the desire to combine specialized libraries, we are faced with the situation of publishers offering their annual publications in electronic form, be it online or on CD, which users want to combine into one collection. Finally, the mere size of such information repositories calls for their separation into smaller parts which are then again integrated into one single system, in order to keep them in manageable orders of magnitude. Most systems in use today rely on central databases to retrieve text documents based on keyword queries. However, with the ever increasing amount of information available, these systems face some severe limitations due to computational demands, since all queries have to be dealt with by a central search engine or its mirrors.

On the other hand research in neural networks has produced promising results in the the field of full text archives. A number of projects [1, 6] especially show the capabilities of one specific type of unsupervised neural network for dealing with the the problems of text document clustering, namely the Self-Organizing Map.[2] Apart from its applicability for query processing, this network architecture provides the advantage of intuitive visualization of the structure of the information repository [4], allowing the implementation of intuitive user interfaces for browsing collections. One of the most prominent examples in this application arena is the WEBSOM[1], providing access to over 1 million articles from 83 Usenet Newsgroups. However, the mere size of the map and the huge computational demands pose limitations from the technical point of view. Recent studies [3] successfully try to reduce the size of the maps and thus the computational demands by using a modified architecture, e.g. the Hierarchical SOM [5], consisting of a hierarchy of (small) maps being trained consecutively layer for layer top down. However, all training data must be processed centrally.

In this paper we propose a model of a distributed SOM-based library system (SOMLib), which combines both local text documents as well as remote library systems, which use a conforming feature vector representation. The system is not limited to a specific architecture of SOM nor does it need central parsing of all documents to create the training vector structure. The overall goal is the integration of several independently managed libraries using the learning capabilities of Neural Networks. Analogously, each user can create both a personal library built up of other library systems as well as create a user-profile covering her fields of interest to enhance keyword queries.

The rest of the paper is built up as follows: Section 2 describes the principles of our Neural Network approach for such a library system. Section 3 details the architecture of the system with a model setup and application scenario detailed in Section 4. Finally we present a short summary in Section 5.

2 A Neural Network Approach

The self-organizing map is an unsupervised neural network providing a mapping from a high-dimensional input space to a usually two-dimensional output space while preserving topological relations as faithfully as possible. Input signals $x \in \Re^n$ are presented to the map, consisting of a grid of units with n-dimensional weight vectors, in random order. An activation function based on some metric (e.g. the Euclidean Distance) is used to determine the winning unit (the 'winner'). In the next step the weight vector of the winner as well as the weight vectors of the neighboring units are modified following some learning rate in order to represent the presented input signal more closely.

Basically, the entries to be included in the library system are represented in the form of feature vectors, which are created by parsing the texts and processing the resulting word histograms to provide a compact and effective rep-

[1] http://nodulus.hut.fi

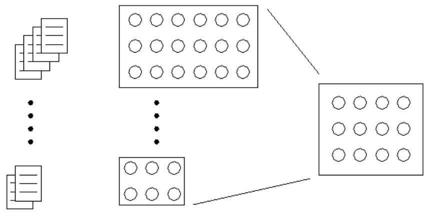

Figure 1: SOMLib-Architecture: Integration of several lower order SOMs

resentation of the texts, using a term frequency / inverse document frequency ($tf \times idf$) representation. These feature vectors are used as input vectors to train a standard self-organizing map, whereby the very architectures used for representing specific libraries do not necessarily have to be identical. Various modifications of the basic architecture, especially hierarchical SOMs, can be used to suit the differing needs of users concerning size, training time and map representation. This approach has proven successful in a number of applications on free form documents with much attention being paid to the design of intuitive user interfaces allowing easy browsing of large collections of documents and simple retrieval of similar pieces of texts. However, so far these systems require all processing, be it feature vector extraction or map training, to take place centrally. When building larger libraries including the documents of several independent repositories (e.g. university libraries, electronic archives of journals) limits are reached, be it the availability of all texts for training as well as the necessary computational power. Thus we need a method to combine different libraries which are based on differing feature vectors made up of the vocabulary of the documents used for training those individual libraries. By combining the feature vectors, the resulting new vectors will grow to form a kind of more universal alphabet as the number of included libraries increases. However, to keep training times low, we use the weight vectors of the SOMLib library maps as input vectors to the higher-order library instead of using all original vectors, resulting in a hierarchy of maps. Still it is not a strict hierarchy in the sense used in the hierarchical SOM model, where each node in a higher level map represents exactly one map at a lower level of the hierarchy. Rather, we train higher level maps bottom-up, obtaining a topographic ordering of all the weight vectors used in the various lower order maps. Thus one node in the higher order map can reference several nodes in different lower order maps. Furthermore, the maps need not necessarily be arranged in a hierarchical order, allowing mutual referencing.

3 SOMLib - The Architecture

Basically, there are two different types of SOMs in the SOMLib architecture. (Figure 1) First, there is a set of independent, small SOMs, referred to as *first order maps*, which are trained with the feature vectors obtained by parsing the documents. Thus every node represents a set of documents, with the whole map representing a topographically ordered mapping of all documents in the library. In a second step, *higher order maps* are trained using the weight vectors of those first order maps as input vectors. Note, however, that the vocabulary and thus the vector structures of those separate libraries differ from each other. Thus, a unique feature vector setup has to be created based upon the different vectors of the libraries to be included by merging the vector structures to train the higher-order map. The resulting map is conceptually identical to the various library maps it is based upon, with the nodes now representing a set of other nodes from the various lower order maps. Analogously, small SOMs trained with relevant documents can be used as user profiles to enhance keyword queries.

For any higher order map two different types of referencing must be considered: with the *static referencing scheme* each node passes on the documents it represents to any higher order map that uses its weight vector for training. In this case every node in any map directly represents a set of documents. With the *node referencing scheme*, higher order maps contain references to the nodes of the maps they are built upon, in which case a node represents a certain topic-area of any lower order map. Static referencing allows direct access to the represented documents while avoiding problems that arise if any of the lower order maps is retrained, in which case the assignment of files to nodes changes. With the node-referencing scheme this requires either the 'old' map to remain available for referencing higher order maps or a re-routing of the corresponding referenced nodes by providing appropriate 'substitute nodes' has to be implemented. This can easily be achieved by mapping all nodes of the 'old' map onto the new, retrained map. An advantage provided by the node-referencing scheme is the fact, that it allows the retrieval of documents that were added to the lower order library document collection after the higher order map was trained. Such documents are not automatically added to the list of referenced documents in the static referencing scheme, but require an update process instead. A combined referencing scheme might provide all the benefits while avoiding the shortcomings: static referencing allows a fast browsing of a high order library map and - with appropriate information given by the static referencing scheme - fast access to the desired up-to-time information on the corresponding lowest order library map in an interactive drill down manner.

4 SOMLib at Work

From the user's point of view, SOMLib facilitates both intuitive library browsing interfaces as well as query processing tools. The topographic mapping provided by a SOM locates similar documents close to each other on neighboring nodes. Thus, SOMLib proves to be a good analogy to what we are used to find in conventional libraries, where books are sorted by topical areas. Furthermore, the implicit hierarchy - which in fact is rather a web of sub-libraries interconnected by reference - allows quick and intuitive access to relevant information by choosing relevant sub-libraries when manually selecting library section after library section. To use a conventional analogy, a user interested in neural networks will start from a high-level map, where he finds (by query or simply by looking at the labels of the nodes) a node covering that topic and several related ones in the close surroundings. Opening this node (section) he either finds a list of documents covering that topic, which are available at different libraries (i.e. static referencing) or a list of sections of other libraries containing documents on that topic. These links can be followed to finally reach the very sections of first order libraries, where access to the desired documents is provided (node referencing).

A query to the library system can consists of a sample text which is parsed using the vector structure of the library system. The resulting query vector is then presented to the library to calculate the winning node. In a first order SOM, the documents mapped onto the winning node are returned as query results. The same applies for higher order maps implementing only static referencing. If both referencing schemes are implemented, in higher order maps the statically referenced documents are returned as first hit query results together with the next order referenced nodes from the node-referencing scheme, i.e. the nodes of the lower order maps that are mapped onto the winning node. In an interactive search the user can now browse through the library 'hierarchy' either manually or by passing the query vector on to desired lower order maps with appropriate conversion of the query vector to match the lower order libraries' vector structure. For automatic retrieval, all references are followed to retrieve documents that are mapped on winning nodes in the lowest order SOM.

From the library administrator's point of view there are two different situations to be considered. On the one hand there are the first level SOMs which are trained with the feature vectors created by parsing (a subset of) existing documents. The resulting maps are relatively small since they only need to represent the very documents present in the library. New documents can be added to the map by parsing them using the previously extracted vector structure and mapping the resulting feature vectors. As long as the general scope of the library does not change extensively, new documents can be added without destroying the topology preserving mapping. As new topics emerge, the small first level libraries need to be retrained. The 'old' SOMLib map can either be retained to serve other referencing higher order maps, or the nodes of the old map can be mapped onto the corresponding nodes in the new SOMLib map by determining the winning node on presentation of the (modified to match the new vector structure) weight vectors of the old map's nodes. If a first order map tends to grow too big, one can choose to split the underlying documents into groups to create separate first order SOMLibs, which are then combined in a higher order map.

Secondly, there are higher order maps to be administered. These are based on several lower order maps, the structure vectors of which are merged to create a new vector. The modified weight vectors of the lower order maps are then used to create the higher level SOMLib map. In many cases a natural hierarchy will evolve in institutional arenas, say several university departments will have their own SOMLibs as first order maps, which are then integrated in a single second order map at university level, which in turn may be combined at a national level and so on. Others may choose to combine first or higher order SOMLibs of institutions covering a certain topic of interest, with the possibility for mutual referencing, to create their personal library system.

5 Conclusion

We have presented a model of a new library organization tool called SOMLib, which is based on a prominent unsupervised neural network architecture, the self-organizing map (SOM). The applicability of SOM and modified architectures to the problem of library organization has been evaluated and shown by many projects so far. SOMLib goes one step further by integrating several independent SOM-based libraries to form higher order library maps. This allows the maintenance of the very libraries locally where training requirements can be met since the maps are relatively small and only a comparably small amount of documents has to be considered. Higher order maps only use the trained lower order maps as basis for their training data to keep both map size and training time low, creating a kind of 'super library'. These higher order libraries do not necessarily need to form a strict hierarchy but can include any other SOMLib system, forming a network of library maps. Users can browse these libraries utilizing the intuitive user interfaces facilitated by the topology preserving mapping of SOMs. Furthermore, documents can be retrieved based on queries, which can either be presented as sample texts, or as keyword queries which in turn can be enhanced by merging the query with a user profile SOM. Due to its modular structure, management of the various libraries is greatly simplified, since they can be independently enlarged and retrained. Finally the system does not pose limitations on architecture variation, so local libraries can be built to best suit the needs of the individual users.

References

[1] T. Honkela, S. Kaski, K. Lagus, and T. Kohonen. Newsgroup exploration with WEBSOM method and browsing interface. Report A32, Helsinki Univ. of Technology, Lab. of Computer and Information Science, Otaniemi, Finland, 1996.

[2] T. Kohonen. *Self-Organizing Maps.* Springer Verlag, Berlin, Germany, 1995.

[3] D. Merkl. Exploration of text collections with hierarchical feature maps. In *Proc. of the 20th Int'l ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 186 – 195, Philadelphia. PA, 1997.

[4] D. Merkl and A. Rauber. Alternative ways for cluster visualization in self-organizing maps. In *Proc. of the Workshop on Self-Organizing Maps (WSOM97)*, Helsinki, Finland, 1997.

[5] R. Miikkulainen. Script recognition with hierarchical feature maps. *Connection Science*, 2:83 – 101, 1990.

[6] A. Rauber and D. Merkl. Finding structure in text archives. In *Proc. European Symp. on Artificial Neural Networks (ESANN98)*, Bruges, Belgium, 1998.