Text Data Mining *

Dieter Merkl
Institut für Softwaretechnik
Technische Universität Wien
Resselgasse 3/188, A-1040 Wien, Austria
dieter@ifs.tuwien.ac.at

Abstract

Classification is one of the central issues in any system dealing with text data. The need for effective approaches is dramatically increased nowadays due to the advent of massive digital libraries containing free-form documents. What we are looking for are powerful methods for the exploration of such libraries whereby the discovery of similarities between groups of text documents is the overall goal. In other words, methods that may be used to gain insight in the inherent structure of the various items contained in a text archive are needed. In this paper we demonstrate the applicability of unsupervised neural networks for the task of text document clustering. Specifically, we describe the results from using self-organizing maps for the exploration of document archives. We further argue in favor of paying more attention to the fact that text archives lend themselves naturally to a hierarchical structure. We take advantage of this fact by using a hierarchically organized network built up from self-organizing maps to represent the contents of a text archive in order to enable the true establishment of a document taxonomy.

1 Introduction

During recent years we have witnessed an ever increasing flood of written information culminating in the advent of massive digital libraries. Powerful methods for organizing, exploring, and searching collections of textual documents are needed to deal with that information. The classical way of dealing with textual information as developed in the information retrieval community is defined by means of keyword-based document representations. These methods may be

*To appear in: R. Dale, H. Moisl, and H. Somers (editors), *A Handbook of Natural Language Processing: Techniques and Applications for the Processing of Language as Text.* New York: Marcel Dekker, 1998.

enhanced with proximity search functionality and keyword combination according to Boole's algebra. Other approaches rely on document similarity measures based on a vector representation of the various texts. What is still missing, however, are tools providing assistance for explorative search in text collections.

Exploration of document archives may be supported by organizing the various documents into taxonomies or hierarchies. In parentheses we should note that such an organization has been in use by librarians for centuries. In order to achieve such a classification a number of approaches are available. Among the oldest and most widely used ones is statistics, especially cluster analysis. The use of cluster analysis for document classification has a long tradition in information retrieval research and its specific strength and weaknesses are well explored [38, 44].

The field of artificial neural networks in a wide variety of applications has attracted renewed interest, which is at least partly due to increased computing power available at reasonable prices. There is general agreement that the application of artificial neural networks may be recommended in areas that are characterized by (i) noise, (ii) poorly understood intrinsic structure, and (iii) changing characteristics. Each of these characteristics is present in text classification. Noise is present due to the fact that no completely satisfying way of representing text documents has been found so far. Secondly, poorly understood intrinsic structure is due to the non-existence of an authority knowing the contents of each and every document. In other words, in document classification we cannot rely on the availability of a person who is able to describe the decisions needed for automating the classification process in an algorithmic fashion based on that person's knowledge of the contents of each document. Hence approaches based on learning techniques have their potential in document classification. Finally, the changing characteristics of document collections are due to the fact that the collections are regularly enlarged to include additional documents. As a consequence, the contents of the document collection varies constantly. Learning techniques are favorable in such an environment compared to algorithmic or knowledge-based approaches.

From the wide range of proposed artificial neural network architectures we regard unsupervised models as especially well suited for the exploration of text collections. This is due to the fact that in a supervised environment one would have to define proper input-output-mappings anew each time the text archive changes, and such changes can be expected to occur quite frequently. By *input-output-mapping* we refer to the manual assignment of documents to classes which, obviously, is only possible when one assumes the availability of considerable insight in the structure of the text archive. On the other hand, in an unsupervised environment it remains the task of the artificial neural network to uncover the structure of the document archive. Hence, the unrealistic assumption of being able to provide proper input-output-mappings is obsolete in an unsupervised environment. A number of successful applications of unsupervised neural networks in the area of text archive exploration have already been

reported in literature [14, 15, 17, 20, 21, 24, 25, 27].

One of the most versatile and successful unsupervised neural network architectures is the *self-organizing map* [18]. It is a general unsupervised tool for ordering high-dimensional statistical data in such a way that similar input items are grouped spatially close to one another. In order to use the self-organizing map to cluster text documents, the various texts have to be represented by means of a histogram of word occurrences. With this data, the network is capable of performing the classification task in a completely unsupervised fashion.

From an exploratory data analysis point of view, however, this neural network model may have its limitations because it commonly uses a two-dimensional plane as the output space for input data representation. In such a restricted low-dimensional space it does not come as a surprise that at least some relationships between various input items are visualized in a rather limited fashion. The limitations are due to the fact that the self-organizing map performs a mapping from a very high dimensional input space represented by the various words used to describe the documents to a two-dimensional output space. Such a mapping, obviously, cannot mirror the similarity between the various documents exactly. As a result, one might come up with an erroneous perception of the similarities inherent in the underlying document collection. A recently proposed neural network architecture that overcomes these limitations while still relying on the self-organizing map as the general underlying model is the *hierarchical feature map* [31]. A specific benefit of this model is the hierarchical organization which imposes a hierarchical structure on the underlying document archive. By making use of this feature, the establishment of a reliable document taxonomy is made feasible, enabling a straight-forward exploration of document similarities.

The remainder of this discussion is organized as follows. In Section 2 we give a very short description of the text representation used for the experiments. Because it is a common feature-based text representation, we feel that the quick review provided in this section is sufficient. In Section 3 we give a brief description of the neural network architectures used for document space exploration. These networks are referred to as *topology preserving neural networks* because of their inherent ability to map similar input patterns onto adjacent regions of the network. In Section 4 we give a detailed exposition of a series of experiments in text archive exploration based on an experimental text collection consisting of the manual pages describing various software components. In particular, we provide a comparison of the effects of using either self-organizing maps or hierarchical feature maps. Finally, we give some conclusions in Section 5.

2 Text Representation

Generally, the task of text clustering aims to discover the semantic similarities between various documents. In the spirit of [8, 9, 22, 23] we regard clustering as one of the essential techniques in the data mining process for discovery of

useful data patterns. Due to the fact that the various documents comprising text archives do not lend themselves to immediate analysis, some pre-processing is necessary.

In order to enable further analyses the documents have, in the first instance, to be mapped onto some representation language. One of the most widely used representation languages is still single-term full-text indexing. In such an environment, the documents are represented by feature vectors $x = (\xi_1, \xi_2, \ldots, \xi_n)^T$. Thereby the ξ_i, $1 \leq i \leq n$, refer to terms[1] extracted from the various documents contained in the archive. The specific value of ξ_i corresponds to the importance of this feature in describing the particular document at hand. Numerous strategies for assigning degrees of importance to features in any particular document are available [39]. Without loss of generality, we may assume that importance is represented as a scalar in the range of [0, 1], where zero means that a particular feature is unimportant for describing the document. Any gradation from zero to one is proportional to the increased importance of the feature in question.

In what is probably the most basic text representation, i.e. binary single-term indexing, the importance of a specific feature is represented by one of two numerical values. Hence, the notion of "importance" is reduced to whether or not a particular word, i.e. document feature, is contained in the document. A value of one represents the fact that the corresponding feature was extracted from the document at hand. On the other hand, a value of zero means that the corresponding feature is not contained in that document.

Such a feature-based document representation is known as the vector-space model for information retrieval. In this model, the similarity between two text documents corresponds to the distance between their vector representations [40, 41].

In the experiments described later we will rely on a binary feature-based document representation. This representation is used as the input to an artificial neural network. There are, however, other document representation techniques available. The utilization of a feature-based document representation may lead to a very high-dimensional feature space. This feature space is generally not free from correlations. As a consequence, one might be interested in the transformation of the original document representation into a lower dimensional space. Statistical techniques that can be used to perform such a transformation include multidimensional scaling [6] and principal component analysis [16]. The former is the underlying principle of *Latent Semantic Indexing* [7, 1]. The latter has been described and successfully applied to text corpora analysis in [2]. In [26] the feature space for document representation is reduced by means of an autoassociative multilayer perceptron. In such a network configuration the activations of the hidden layer units, which are numerically fewer that those in the input and output layer, represent an approximation to the principal components of

[1] In the remainder of this discussion we will use the words *term* and *keyword* interchangeably to refer to entities that are selected to represent the contents of documents. Collectively these terms represent the feature space to describe the document collection.

the input data. These activations are then used for self-organizing map training instead of the original input patterns yielding substantially reduced training time while still enabling comparable results as far as text archive organization is concerned. Recently, a number of papers have been published on the utilization of the self-organizing map for large scale document representation [15] based on the seminal work of [36] and subsequent interactive exploration [14, 20].

3 Topology Preserving Neural Networks

3.1 Competitive Learning Basics

Competitive learning [37], or *winner-takes-all* as it is termed quite often, may be regarded as the basis of a number of unsupervised learning strategies. In its most rudimentary form, a competitive learning network consists of k units with weight vectors m_k of dimension equal to the input data, $m_k \in \Re^n$. During learning, the unit with its weight vector closest to the input vector x is adapted in such a way that the weight vector resembles the input vector more closely after the adaptation. To determine the distance between the vectors any distance (or similarity) metric may be selected, though the most common choices are marked by the Euclidean vector norm and the inner product of the vectors. The unit with the closest weight vector is dubbed the *winner* of the selection process. This learning strategy may be implemented by gradually reducing the difference between weight vector and input vector. The actual amount of difference reduction at each learning step may be guided by means of a so-called learning-rate α in the interval [0, 1]. When given such a learning environment, the various weight vectors converge towards the mean of the set of input data represented by the unit in question.

Obviously, the term *winner-takes-all* refers to the fact that only the winner is adapted whereas all other units remain unchanged. As a severe limitation of this basic learning strategy consider a situation where there are some units which, due to random initialization of their weight vectors, are never selected as the winner and whose weight vectors are consequently never adapted. Strictly speaking, such units may be referred to as *dead units* since they do not contribute to the learning process and thus do not contribute to input data representation. This possibility has led to the development of a number of learning rules and network architectures that overcome this limitation by enlarging the set of units that are affected by adaptation at each learning step.[2] Apart from the winner, adaptation is performed with units in some defined vicinity around the winner. This type of learning rule may be referred to as *soft competitive learning*, representatives of which are described in the following subsections.

[2]We have to admit that this is a rather sloppy formulation as far as historical chronology is concerned. In fact the *self-organizing map*, the neural network model to be described in the next subsection, appeared earlier in literature as the basic *winner-takes-all* learning rule.

3.2 Self-Organizing Maps

The *self-organizing map* as proposed in [18] and described thoroughly in [19] is one of the most widely used unsupervised artificial neural network models. It consists of a layer of input units each of which is fully connected to a set of output units. These output units are arranged in some topology where the most common choice is represented by a two-dimensional grid.

Input units receive the input patterns x, $x \in \Re^n$, and propagate them as they are onto the output units. Each of the output units i is assigned a weight vector m_i. These weight vectors have the same dimension as the input data, $m_i \in \Re^n$.

During each learning step, the unit c with the highest activity level with respect to a randomly selected input pattern x is adapted in such a way that it will exhibit an even higher activity level at future presentations of x. Commonly, the activity level of a unit is computed as the Euclidean distance between the input pattern and that unit's weight vector. Hence, the selection of the winner c may be written as given in expression 1.

$$c : ||x - m_c|| = \min_i \{||x - m_i||\} \tag{1}$$

Adaptation takes place at each learning iteration and constitutes a gradual reduction of the difference between the respective components of the input vector and the weight vector. The amount of adaptation is guided by a learning-rate parameter α that gradually decreases in the course of learning. This decrease ensures large adaptation steps at the beginning of the learning process where the weight vectors have to be tuned from their random initialization towards the actual requirements of the input space. Furthermore, the ever smaller adaptation steps towards the end of the learning process enable a fine-tuned input space representation.

As an extension to standard competitive learning, units in a time-varying and gradually decreasing neighborhood around the winner are adapted too. During the learning steps of the self-organizing map a set of units around the winner is tuned towards the currently presented input pattern enabling a spatial arrangement of the input patterns such that similar inputs are mapped onto regions close to each other in the grid of output units. Thus, the training process of the self-organizing map results in a topological ordering of the input patterns. According to [35] we may thus refer to the self-organizing map as a neural network model performing a spatially smooth version of k-means clustering.

The neighborhood of units around the winner may be described implicitly by means of a neighborhood kernel h_{ci} taking into account the distance between unit i under consideration and unit c, the winner of the current learning iteration. This neighborhood kernel assigns scalars in the range of [0, 1] that are used to determine the amount of adaptation, ensuring that nearby units are adapted more strongly than units further away from the winner. A Gaussian

may be used as the neighborhood kernel. It is common practice that at the start of the learning process, the neighborhood kernel is selected large enough to cover a wide area of the output space. The spatial width of the neighborhood kernel is reduced gradually during the learning process so that, towards the end of the learning process, just the winner itself is adapted. This strategy enables the formation of large clusters at the beginning and fine-grained input discrimination towards the end of the learning process.

Combining these principles of self-organizing map training, we may write the learning rule as given in expression 2. Please note that we use a discrete time notation with t denoting the current learning iteration. The other parts of this expression are α representing the time-varying learning-rate, h_{ci} representing the time-varying neighborhood kernel, x representing the current input pattern, and, finally, m_i denoting the weight vector assigned to unit i.

$$m_i(t+1) = m_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot [x(t) - m_i(t)] \tag{2}$$

A simple graphical representation of a self-organizing map's architecture and its learning process is provided in Figure 1. In this figure the output space consists of a square of 36 units, depicted as circles. One input vector $x(t)$ is randomly chosen and mapped onto the grid of output units. In a subsequent step the winner is selected, which is depicted as the black node in the figure. The weight vector of the winner, $m_c(t)$, is now moved towards the current input vector. This movement is symbolized in the input space in Figure 1. As a consequence of the adaptation, this unit c will produce an even higher activation with respect to input pattern x at the next learning iteration, $t+1$, because the unit's weight vector, $m_c(t+1)$, is now nearer to the input pattern x in terms of the input space. Apart from the winner, adaptation is performed with neighboring units too. Units that are subject to adaptation are depicted as shaded nodes in the figure. The shading of the various nodes corresponds to the amount of adaptation and thus to the spatial width of the neighborhood kernel. Generally, units in close vicinity of the winner are adapted more strongly, and consequently they are depicted with a darker shade in the figure.

3.3 Hierarchical Feature Maps

The self-organizing map is a neural network model capable of arranging high-dimensional input data within its (usually) two-dimensional output space in such a way that the similarity of the input data is mirrored as faithfully as possible in terms of the topographic distance between the respective winning units. The utilization of this model is thus especially well suited in application areas where one is highly dependent on a convenient data representation and visualization for subsequent exploration of the input data space. The exploration of text collections exemplifies this type of application.

With respect to such exploration, however, we have to note a deficiency of the

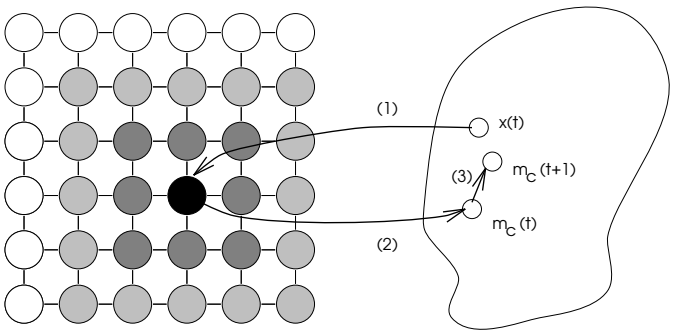

Figure 1: Architecture of a self-organizing map

self-organizing map. This deficiency is the inability of the standard model to represent cluster boundaries explicitly. A substantial amount of research addressing the representation of cluster boundaries has emerged as a natural consequence. Without going into greater detail, we may distinguish three different approaches to overcoming this deficiency. The first focuses on improved visualization of the training result. Examples of this line of research are [5, 29, 42]. A second approach addresses cluster boundary detection by means of adaptive network architectures. More precisely, these models rely on incrementally growing and splitting architectures, where the final shape of the network is determined by the specific requirements of the input space. The work of [3, 4, 10, 11] describe this line of research. The major problem with these models is that they require a few more parameters to be specified prior to network training. Consequently, the utilization of these models is more demanding as far as the necessary experience of the users is concerned [17]. The third and final approach to cluster boundary uses individual self-organizing maps assembled into a neural network with a layered architecture. We refer to [31, 43] as proponents of this line of research.

With respect to our goal of text data mining we regard the *hierarchical feature map* as described in [31, 32, 33, 34] as the most promising among the approaches to cluster boundary recognition as outlined above. The reason is that this model imposes a hierarchical structure on the underlying input data which resembles the organizational principle in use by librarians for centuries for text corpora organization. Moreover, the time needed to train the network is substantially shorter than for self-organizing maps, thus providing a highly convenient environment for text analysis and exploration.

The key idea of the hierarchical feature map is to arrange a number of self-organizing maps in a hierarchy such that for each unit on one level of the hierarchy a two-dimensional self-organizing map is added to the next level. The

resulting architecture may thus be characterized as having the shape of a pyramid as depicted in Figure 2.

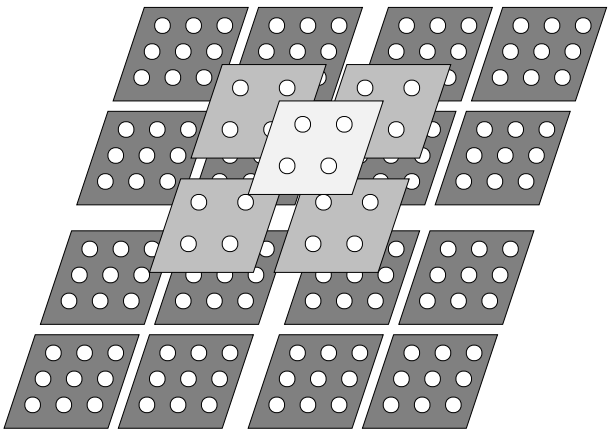

Figure 2: Architecture of a hierarchical feature map

The training of the hierarchical feature map is performed sequentially from the first layer downwards in the hierarchy. The maps on each layer are trained according to the standard learning rule of self-organizing maps as outlined above. As soon as the first layer map has reached a stable state, training continues with the maps forming the second layer. Within the second layer, each map is trained only with that subset of the original input data that was mapped onto the corresponding unit of the first layer map. Moreover, the dimension of the input patterns may be reduced at the transition from one layer to the next by omitting that portion of the document representation that is equal in the patterns mapped onto the same unit. In other words, the keywords that are found to represent each document mapped onto the same unit may be removed, and the next layer is trained only with the now shorter remaining portion of the document representation. The rationale behind this reduction is that the omitted vector components represent features that are already learned by the higher layer map. This characteristic is of especial importance when dealing with text as the underlying input data because, firstly, text documents are represented in a very high-dimensional feature space, and, secondly, some of these features are common to each text belonging to a particular subject. Due to this dimension reduction, the time needed to train the maps is reduced too. Training of the second layer is completed when each map has reached a stable state. Analogously, the same training procedure is used to train the third and any subsequent layers of the hierarchical feature map.

A valuable property of the hierarchical feature map is the substantial speed-

up of the training process as compared to conventional self-organizing maps. An explanation that goes beyond the obvious dimension reduction of the input data emerges from an investigation of the general properties of the self-organizing training process. In self-organizing maps, the units that are subject to adaptation are selected by means of the neighborhood kernel. It is common practice that, at the start of the training process, almost the whole map is affected by the presentation of an input vector. With this strategy, the map is forced to establish initial clusters of similar input items at the outset of learning. By reducing the width of the neighborhood kernel in the course of learning, the training process is able to learn ever finer distinctions within the clusters while the overall topology of cluster arrangement is maintained. The flipside of the coin, however, is that units along the boundary between two clusters tend to be occasionally modified as belonging to either one of these clusters. This interference is the reason for the time-consuming self-organizing process. Such an interference is dramatically reduced in hierarchical feature maps. This reduction is due to the architecture of this neural network. The topology of the high-level categories is represented in the first layer of the hierarchy. Each of its sub-categories are then independently organized within separate maps at lower levels within the hierarchy. These maps in turn are free from having to represent the overall structure, as this structure is already determined by the architecture of the hierarchical feature map. As a consequence, a set of documents that is mapped onto the same unit in any of the layers is further separated in the following layer of the hierarchical feature map. Assuming that such a unit represents a meaningful cluster of documents, i.e. a set of documents covering the same subject matter, this cluster is further organized in the next layer of the artificial neural network. In summary, much computational effort is saved due to the fact that the overall structure of clusters is determined by the architecture of the artificial neural network rather than by its learning rule.

4 A Voyage Through Document Spaces

In this section we will describe some of the results we achieved when using unsupervised neural networks for the exploration of a document archive. In particular, we will give examples from text archive organization by means of self-organizing maps and hierarchical feature maps.

4.1 An Experimental Document Collection

Throughout the remainder of this work we will use the various manual-pages of the NIH Class Library [12], *NIHCL*, as a sample document archive. The *NIHCL* is a collection of classes developed in the C++ programming language. The class library covers classes for storing and retrieving arbitrarily complex data structures on disk, generally useful data types such as `String`, `Time`, and `Date`,

and, finally, a number of container classes like, for example, `Set`, `Dictionary`, and `OrderedCltn`. More general information concerning the *NIHCL* may be found in [13].

The selection of the *NIHCL* is motivated by the fact that software libraries represent a convenient application arena for information retrieval systems. The reason is that much of the information about a particular software component is available in textual description organized as manual pages. Moreover, the results of the classification process may easily be evaluated because the semantics of the software component, i.e. its functionality, is well known.

The full text of the various manual pages describing the classes was accessed and indexed in order to generate a binary vector-space representation of the documents. The indexing process identified 489 distinct content terms, and each component is thus represented by a 489-dimensional feature vector. These vectors are subsequently used as the input data to the artificial neural network. Note that we do not make use of the meta-information contained in the manual such as references to *base class*, *derived classes*, and *related classes*. Such information is not generally available in document archives, and thus, for the sake of general applicability, we excluded this type of meta-information from our work. Such information might well be of importance, however, if one were specifically interested in an application fine-tuned to this particular document archive.

4.2 A Map of the Document Collection

A typical result from the application of self-organizing maps to this type of data is represented in Figure 3. In this case we have used a 10×10 self-organizing map to represent the document collection. Generally, the graphical representation may be interpreted as follows. Each output unit of the self-organizing map is represented by means of either a dot or a class name. The class name appears where the respective unit is the winner for that specific input pattern. By contrast, a dot marks units that have not won the competition for an input pattern. These units, however, have an important role in determining the spatial range of the various data clusters.

For convenience of discussion we have marked some of the interesting regions manually in Figure 3. In the upper right part of the map we recognize the arrangement of all classes performing file I/O; these classes are designated by the '`OIO`'-part of their respective class names. The region itself is organized in such a way that a class performing an input operation is mapped neighboring its output counterpart, e.g. `OIOifd` and `OIOofd`. Just below the file I/O region we find a large area of the map consisting of the various classes representing data structures. Within this larger region we have marked the area comprising classes that allow an access to their elements via a key-attribute, i.e. `Dictionary`, `IdentDict`, and `KeySortCltn`. Within this area we find the classes that actually implement this type of access too, i.e. `Assoc`, `AssocInt`, and `LookupKey`. Finally, we want to shift the attention towards the left-hand

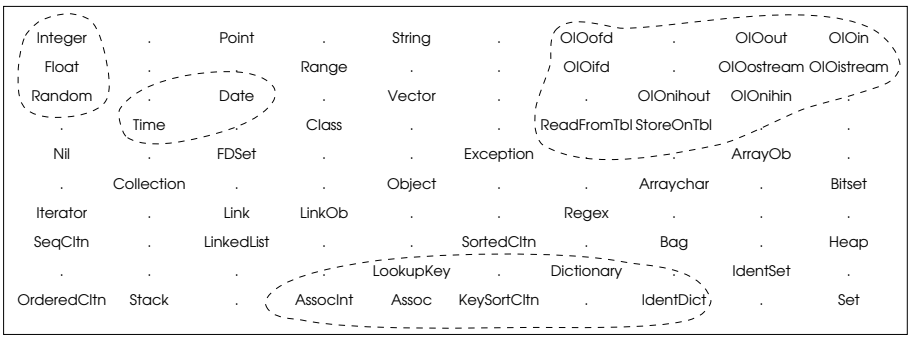

Figure 3: *NIHCL* presented in a 10×10 self-organizing map

side of the map where the classes representing data types are located. Again, their arrangement reflects their mutual similarity. For instance, note the placement of class `Random`, a random number generator producing numbers of `Float` data type. This class is mapped onto a unit neighboring the basic numerical data types `Float` and `Integer`. Many more interesting areas may be located in this representation. They cannot be covered in any detail here, so the reader is referred to [25], where experimental results are compared to those obtained from statistical cluster analysis.

4.3 A Taxonomy of Documents

In Figure 4 we show a typical result from organizing the *NIHCL* document archive by means of the hierarchical feature map. Due to the hierarchical arrangement of a number of independent self-organizing maps, the inherent hierarchical organization of the text archive is mirrored in almost perfect fashion. In the top-layer map, shown in the center of Figure 4, the four main groups of classes are clearly separated. Each group in its turn is further detailed within its own self-organizing map in the second layer of the hierarchy.

More precisely, we find the classes performing file I/O in the left upper map of the second layer depicted in Figure 4. Within this map, the various classes are arranged in pairs according to their respective functionality. In the right upper map we recognize the classes representing data types. The large number of entries in the right upper unit of this particular map is due the fact that the library's general base classes, i.e. `Object`, `Class`, and `Exception`, are regarded as "data types", too. These classes, however, are clearly separated in the third layer. We refrain from its graphical representation here for space considerations. In the lower left map of the second layer we find mostly the data structures that allow access via a key attribute. As with the self-organizing map, they are

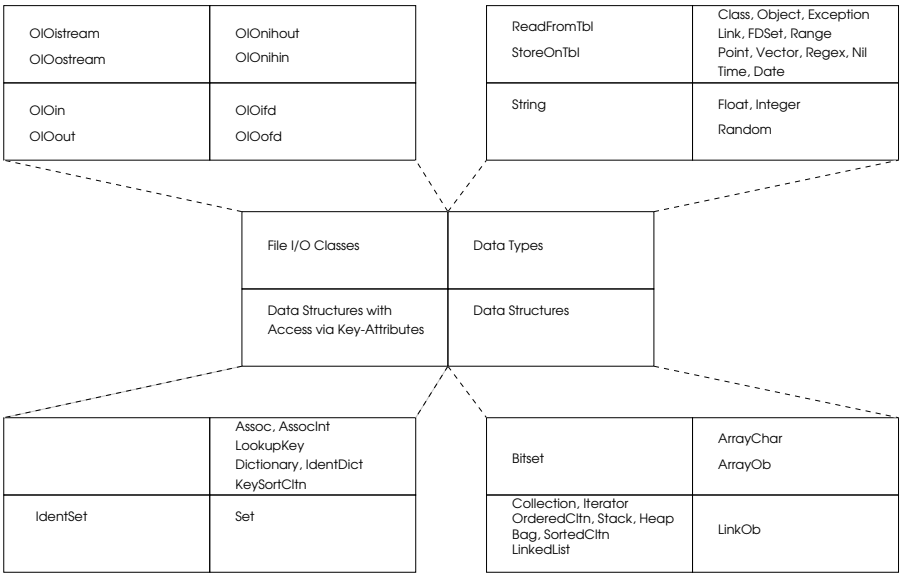

Figure 4: *NIHCL* presented in a hierarchical feature map

represented together with the classes that actually perform the key attribute based access. Finally, the remaining data structures are located in the lower right map of the second layer. Here again we find a unit that represents a large number of classes. In this case, the base class for all container classes, i.e. `Collection`, and the class providing the uniform means of access for all container classes, i.e. `Iterator`, are mapped onto this unit as well. Again, these classes are clearly separated in the next layer of the hierarchy.

4.4 A Comparison of both Approaches

In the previous subsections we have described the results of both neural network architectures without favoring either one. We feel, however, that the results from the hierarchical feature map are preferable for a number of reasons. Firstly, imagine Figure 3 without the lines indicating the cluster boundaries as shown in Figure 5. Such a representation may be regarded as the standard form of visualization with self-organizing maps. In such a representation, a number of erroneous perceptions concerning the mutual similarity of various documents in the collection might occur because it is often difficult to demarcate region boundaries of similar documents. Consider for instance the borderline between the file I/O classes and the data types. A casual user might conclude that the

classes `OIOofd` and `String` are comparatively similar as the classes `String` and `Point` are because both pairs are mapped onto equidistant units in the upper middle of the map. The similarity of the latter pair is obvious in that both are data types. The former pair, though, consists of rather unrelated classes, the one being a data type, the other a file I/O class. To draw the borderline between clusters some insight into the underlying document archive is definitely necessary.

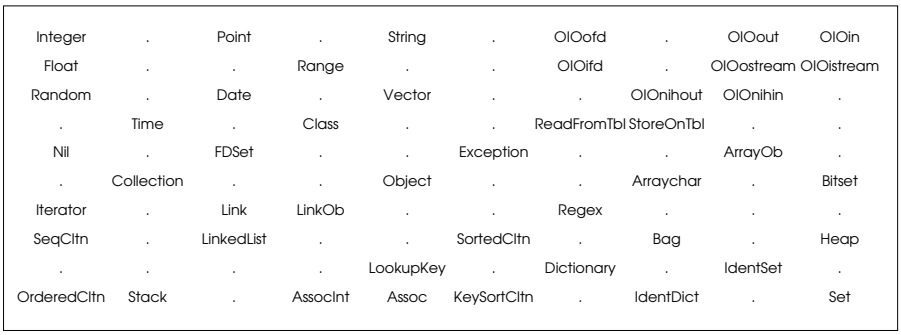

Figure 5: *NIHCL* without manually provided cluster boundaries

Conversely, with the hierarchical feature map this separation into distinct clusters comes for free because of the architecture of the neural network. A hierarchical organization of text collections is especially appealing because it emerges naturally. The inherent hierarchical organization of text collections arises from their decomposition into various topics that are dealt with within the documents. This is, obviously, the underlying organizational principle used by librarians to structure the contents of a library. In this sense, we rely on the user's experience in coping with hierarchically arranged data spaces.

Finally, a strong argument in favor of the hierarchical feature map emerges from the time needed to train the network. With the data set and the network dimensions as described above, the time needed to finish the training process is about one hour for the self-organizing map and about 10 minutes for the hierarchical feature map [28]. This timing was done on an otherwise idle SPARC-20 workstation. These characteristics scale up quite nicely as indicated in [30] with a larger document collection both in terms of the number of documents and in terms of the dimension of the feature space used to describe the various texts.

5 Conclusion

In this paper we have described how unsupervised neural networks can be used in order to reveal the relative similarity of documents contained in a text archive. More precisely, we contrasted the results obtained from self-organizing maps with those from hierarchical feature maps. In a nutshell, both models proved to be successful in organizing the various documents according to their relative similarities. As far as an explicit and intuitive visualization is concerned, however, we noted a specific benefit of hierarchical feature maps. This benefit is marked by the inherent hierarchical organization of the underlying input data space. Such a hierarchical organization seems to be well suited to an application area such as text archive exploration, where a corresponding organization according to the various topics that the documents deal with may be found. Librarians have made use of exactly this property of text archives for centuries in order to organize the contents of libraries. Hierarchical feature maps were, moreover, found to require much shorter training times than self-organizing maps. Both of the neural network models described in this paper assist the user in uncovering the inherent structure of a text archive. They thus represent useful tools for an interactive exploration of document collections.

Acknowledgments

Part of this work has been done while the author was visiting the Department of Computer Science at the Royal Melbourne Institute of Technology.

References

[1] B. T. Bartell, G. W. Cottrell, and R. K. Belew. Latent semantic indexing is an optimal special case of multidimensional scaling. In *Proc ACM SIGIR Int'l Conference on Research and Development in Information Retrieval (SIGIR'92)*, Copenhagen, Denmark, 1992.

[2] T. Bayer, I. Renz, M. Stein, and U. Kressel. Domain and language independent feature extraction for statistical text categorization. In *Proc Workshop on Language Engineering for Document Analysis and Recognition*, Sussex, UK, 1996.

[3] J. Blackmore and R. Miikkulainen. Incremental Grid Growing: Encoding high-dimensional structure into a two-dimensional feature map. In *Proc IEEE Int'l Conference on Neural Networks*, San Francisco, CA, 1993.

[4] J. Blackmore and R. Miikkulainen. Visualizing high-dimensional structure with the Incremental Grid Growing network. In *Proc Int'l Conference on Machine Learning*, Lake Tahoe, NV, 1995.

[5] M. Cottrell and E. de Bodt. A Kohonen map representation to avoid misleading interpretations. In *Proc European Symposium on Artificial Neural Networks (ESANN'96)*, Brugge, Belgium, 1996.

[6] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling.* Chapman & Hall, London, 1994.

[7] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Hashman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6), 1990.

[8] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining.* AAAI Press, Menlo Park, CA, 1996.

[9] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, 39(11), 1996.

[10] B. Fritzke. Kohonen feature maps and Growing Cell Structures: A performance comparison. In C. L. Gibs, S. J. Hanson, and J. D. Cowan, editors, *Advances in Neural Information Processing Systems 5.* Morgan Kaufmann, San Mateo, CA, 1993.

[11] B. Fritzke. Growing Cell Structures: A self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9), 1994.

[12] K. E. Gorlen. *NIH class library reference manual.* National Institutes of Health, Bethesda, MD, 1990.

[13] K. E. Gorlen, S. Orlow, and P. Plexico. *Abstraction and Object-Oriented Programming in C++.* John Wiley, New York, 1990.

[14] T. Honkela, S. Kaski, K. Lagus, and T. Kohonen. Newsgroup exploration with WEBSOM method and browsing interface. Technical Report A32, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland, 1996.

[15] T. Honkela, V. Pulkki, and T. Kohonen. Contextual relations of words in Grimm tales analyzed by self-organizing maps. In *Proc Int'l Conference on Artificial Neural Networks (ICANN'95)*, Paris, France, 1995.

[16] I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, Berlin, 1986.

[17] M. Köhle and D. Merkl. Visualizing similarities in high dimensional input spaces with a growing and splitting neural network. In *Proc Int'l Conference on Artificial Neural Networks (ICANN'96)*, Bochum, Germany, 1996.

[18] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 1982.

[19] T. Kohonen. *Self-organizing maps*. Springer-Verlag, Berlin, 1995.

[20] K. Lagus, T. Honkela, S. Kaski, and T. Kohonen. Self-organizing maps of document collections: A new approach to interactive exploration. In *Proc Int'l Conference on Knowledge Discovery and Data Mining (KDD-96)*, Portland, OR, 1996.

[21] X. Lin, D. Soergel, and G. Marchionini. A self-organizing semantic map for information retrieval. In *Proc ACM SIGIR Int'l Conference on Research and Development in Information Retrieval (SIGIR'91)*, Chicago, IL, 1991.

[22] H. Mannila. Data mining: Machine learning, statistics, and databases. In *Proc Int'l Conference on Scientific and Statistical Database Management*, Stockholm, Sweden, 1996.

[23] H. Mannila. Methods and problems in data mining. In *Proc Int'l Conference on Database Theory*, Delphi, Greece, 1997.

[24] D. Merkl. Structuring software for reuse – The case of self-organizing maps. In *Proc Int'l Joint Conference on Neural Networks (IJCNN'93)*, Nagoya, Japan, 1993.

[25] D. Merkl. A connectionist view on document classification. In *Proc 6th Australasian Database Conference (ADC'95)*, Adelaide, Australia, 1995.

[26] D. Merkl. Content-based document classification with highly compressed input data. In *Proc 5th Int'l Conference on Artificial Neural Networks (ICANN'95)*, Paris, France, 1995.

[27] D. Merkl. Content-based software classification by self-organization. In *Proc IEEE Int'l Conference on Neural Networks (ICNN'95)*, Perth, Australia, 1995.

[28] D. Merkl. Exploration of text collections with hierarchical feature maps. In *Proc ACM SIGIR Int'l Conference on Research and Development in Information Retrieval (SIGIR'97)*, Philadelphia, PA, 1997.

[29] D. Merkl and A. Rauber. On the similarity of eagles, hawks, and cows – Visualization of similarity in self-organizing maps. In *Proc Int'l Workshop Fuzzy-Neuro-Systems'97*, Soest, Germany, 1997.

[30] D. Merkl and E. Schweighofer. The exploration of legal text corpora with hierarchical neural networks – A guided tour in public international law. In *Proc Int'l Conference on Artificial Intelligence and Law*, Melbourne, Australia, 1997.

[31] R. Miikkulainen. Script recognition with hierarchical feature maps. *Connection Science*, 2, 1990.

[32] R. Miikkulainen. Trace feature map: A model of episodic associative memory. *Biological Cybernetics*, 66, 1992.

[33] R. Miikkulainen. *Subsymbolic Natural Language Processing: An integrated model of scripts, lexicon, and memory.* MIT-Press, Cambridge, MA, 1993.

[34] R. Miikkulainen. Script-based inference and memory retrieval in subsymbolic story processing. *Applied Intelligence*, 5, 1995.

[35] B. D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, UK, 1996.

[36] H. Ritter and T. Kohonen. Self-organizing semantic maps. *Biological Cybernetics*, 61, 1989.

[37] D. E. Rumelhart and D. Zipser. Feature discovery by competitive learning. In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. I. Foundations.* MIT Press, Cambridge, MA, 1986.

[38] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer.* Addison-Wesley, Reading, MA, 1989.

[39] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5), 1988.

[40] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983.

[41] H. R. Turtle and W. B. Croft. A comparison of text retrieval models. *Computer Journal*, 35(3), 1992.

[42] A. Ultsch. Self-organizing neural networks for visualization and classification. In O. Opitz, B. Lausen, and R. Klar, editors, *Information and Classification – Concepts, Methods, and Applications.* Springer-Verlag, Berlin, 1993.

[43] W. Wan and D. Fraser. Multiple Kohonen self-organizing maps: Supervised and unsupervised formation with application to remotely sensed image analysis. In *Proc Australian Conference on Neural Networks*, Brisbane, Australia, 1994.

[44] P. Willet. Recend trends in hierarchic document clustering: A critical review. *Information Processing & Management*, 24, 1988.