Automatic Generation of Local Internet Catalogues using Hierarchical Radius-based Competitive Learning

Udo Heuser and **Wolfgang Rosenstiel** [1]

Abstract. This work presents a way to cluster HTML document sets in an hierarchical manner. The hierarchical clustering is performed using the *Hierarchical Radius-based Competitive Learning (HRCL)* neural network that has been developed by the authors. After a detailed discussion of the algorithm, HRCL clustering as well as retrieval results will be presented. The HRCL clustering results in a hierarchical multi-resolution view of the underlying (local) HTML data collection, consisting of clusters (with its cluster centroids), subclusters, sub-subclusters and so forth. The results can be combined and rendered in the way of an Internet catalogue resembling the known *Yahoo* directory. The Internet search can finally be accelerated using the automatically generated (sub-)cluster centroids.

1 STATE OF THE ART

Conventional Internet search engines, like *Altavista*^{TM}, usually show problems in handling the huge amount of the already available data of the Internet adequately: Preliminary experiments show that Altavista is not capable of presenting the homepage of our department on its first result pages[2]. Similar drawbacks can be seen with todays Internet directories: *Yahoo.com*^{TM} did not even enter the homepage on its list. Search engines are primarily based on inverted indexes that contain web pages as copies of those visited by crawlers. After copying, search engines basically use the *location/frequency* method to judge the relevance of its documents: Documents that contain search keywords in its titles or in one of the first sections (*location*) are considered more relevant than others. Documents can increase their relevances, if keywords occur more *frequently* in the named sections. Some engines, like *WebCrawler*^{TM}, complete the relevance calculation by *link popularities (relevancy boosters)*: HTML documents that are often referred to from other pages are considered more relevant than others [1].
Directories, on the other hand, are edited manually. Due to this situation, only a small (subjective) part of the World Wide Web (WWW) can be considered relevant and entered on its lists.

1.1 Intelligent information retrieval

Techniques that try to circumvent mentioned drawbacks already exist, to name above all the *Information Retrieval (IR)* techniques. In general, IR methods can be defined as those trying to detect unknown correlations or a hidden knowledge of document sets that are heterogenous, "unordered" and dynamic (i.e. possibly growing and shrinking). The WWW can be deemed as such. As the central part of the overall "intelligent" IR process, the Data Mining (DM) as the knowledge extraction step, may consist of a classification or cluster analysis process. Before clustering, input documents have to be translated to vectors embedded in a n-dimensional vector space, thus obeying the so-called *vector space model*. The cluster analysis can now define clusters as natural groupings (or high probability densities) of the vector space, commonly exploiting similarity metrices (cf. [2, 3]). Detected clusters may finally represent those input documents that are topically closely related to each other. Hence, clustering can improve the overall retrieval performance.

1.2 Statistical and neural cluster analysis

To name just a few statistical clustering methods that can be or are already used to improve searching, there are among others the K-means algorithm [4], the single- or one-pass cluster analysis used in the SMART retrieval system (cf. [5], p. 127 ff.), density estimations or mode seeking techniques (cf. [6]), the minimum spanning tree [7] or the nearest neighbor approach [8].

In general, statistical methods suffer from the fact that the obtained results are depending on the order of input data fed into the system. Thus, statistical methods are called *model-driven* in contrast to the *data-driven* and *non-linear neural networks*: Above all statistical mode seeking methods can be extended by *competitive learning* neural nets, also known as *vector quantization* methods. These try to describe clusters by placing reference or codebook vectors at every cluster, thus minimizing the expected quantization error. Among these are prominent e.g. the self-organizing map (SOM; [9, 10]), the neural gas (NG; [11]) and the growing neural gas (GNG; [12]).

Unfortunately, although sufficiently solving the input order dependency problem, competitive learning methods do not in general place one neuron at each input cluster's center. Instead they confine themselves to place its neurons to locations such that cluster distributions are described best, thus not meeting the vector quantization demand completely. It soon becomes clear that it would be desirable not only to describe well the cluster distributions, but also to detect the cluster centers at the same time. To solve this problem, we developed the *Hierarchical Radius-Based Competitive Learning (HRCL)* as neural cluster analysis method. The HRCL hierarchical clustering will be described in details in section 3. Beforehand, we will motivate our general approach.

2 MOTIVATION

The OASIS project has been designed to ameliorate Internet searching. As section 1 showed, centralized Internet search engines, like Al-

[1] Wilhelm-Schickard-Institute for Informatics, Department of Computer Engineering, University of Tübingen, Sand 13, 72076 Tübingen, Germany. email: {heuser|rosen}@informatik.uni-tuebingen.de

[2] The homepage was shown at the 83^{rd} place (9^{th} results page). The search was initiated using the boolean search option *"technische informatik"* **AND** *"universität tübingen"* together with "any language".

tavista, can not compete with the tremendously expanding Internet. OASIS circumvents this by distributing the information among several OASIS servers. Every server contains its own local HTML data collection (favourably not intersecting with any other), whose documents will be collected by crawlers running as background processes. Each local collection will be clustered by HRCL that automatically generates a hierarchical tree of cluster, sub-cluster prototypes etc. – a catalogue comparable to the Yahoo directory. User queries will finally be directed to those OASIS server(s) whose collections (or prototypes) fit best. Relevant documents will be merged reasonably and issued to the user (cf. fig. 1 and [13, 14]).

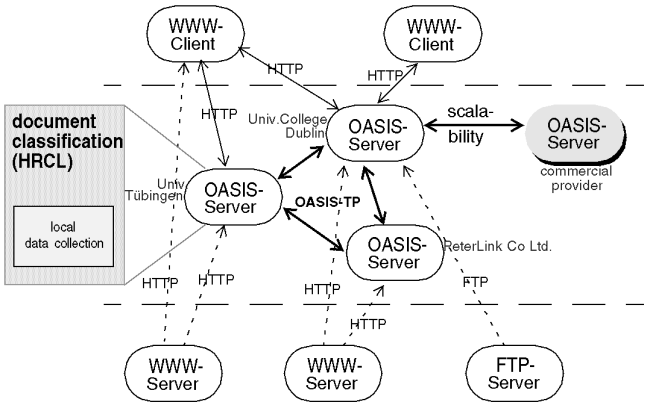

Figure 1. OASIS distributed and intelligent Internet search system

3 HRCL

HRCL is primarily based on the neural gas approach and is, like NG, a soft competitive learning method without existing network dimensionality. Neuron weights $w_{c_i} \in R^n$ of a given HRCL neuron set $A := \{c_i\}$ are ordered relative to their distances to the input vector $\xi_i \in R^n$, each time an input is randomly chosen and presented to HRCL. Contrary to NG, each HRCL neuron consists of an hypercubical environment $U_c(r) := (2r)^n$ with user defined radius r. The neuron's hypercube $U_c(r)$ has dimension n of the (scaled) input vector space $D \subset R^n$. Before training, an "equipartition" threshold $\Theta(r)$, relative to radius r, is defined that drives the neuron's adaptations: If a neuron c_i can collect more input vectors ξ_i inside its hypercubical environment $U_{c_i}(r)$ than given by $\Theta(r)$, then HRCL fixes c_i. If neuron c_i is fixed at training step t, it will be adapted towards the current input ξ_i if and only if c_i can collect at its new position at time $t+1$ not fewer input samples *and* if its hypercubical environment does not intersect with another neuron's hypercube. If it will intersect with one of the existing neuron's hypercube and the neuron in concern c_i has not been fixed at time t, c_i will be repelled from the current input ξ_i. For the adaptation as well as the repelling of neurons with weights w_i at training step t we use equations 1 and 2 respectively:

$$w_i(t+1) = w_i(t) + f_{adapt}(\xi, A) \cdot g_{adapt}(t) \cdot (\xi - w_i) \quad (1)$$

$$w_i(t+1) = w_i(t) - f_{repel}(\xi, A) \cdot g_{repel}(t) \cdot (\xi - w_i) \quad (2)$$

$f_{adapt}(\xi, A)$ and $g_{adapt}(t)$ are exponentially decreasing, relative to the order of neurons $c_i \in A$ at training step t and current input ξ_i. $f_{repel}(\xi, A)$ and $g_{repel}(t)$ are exponentially decreasing relative to A, but increasing in training time t.

The initial number of HRCL neurons (seeds) N is given through parameter p and is relativ to the number of input vectors, thus: $N := |D|^p$, $(p \in [0, 1])$. Neuron seeds c_i are placed at positions marked by N input vectors ξ_i at random. After initializing of neurons, the "equipartition" threshold $\Theta(r)$ is calculated and equals the median of input vectors enclosed by hypercubes (e.g. $|U_{c_i}(r)|$) of all seeds c_i. Moreover, HRCL features neurons c_i to be pruned while and after training. Pruning is performed for any neuron c_i that is unfixed for more than a given period of time $\Delta(t)$ or that tries to leave the scaled input vector space D in one of the input dimensions n.

After training, a top-down *hierarchical refinement* is realized, if the number of resulting HRCL neurons $|A|$ is greater than 1. Each hypercube $U_c(r)$ defines the new input vector space D that is scaled to $[-1; 1]^n$. HRCL training is continued for each new D until the refinement termination condition is yielded.

4 RESULTS

Below, we will present both HRCL clustering as well as retrieval results. HRCL clustering will use artificial test data, whereas subsection 4.2 uses real HTML document collections to allow comparisons with existing Internet directories.

4.1 Clustering results

In this part of our work, we want to compare the HRCL hierarchical clustering results with those obtained by a standard SOM as well as by the single-pass clustering analysis used in SMART retrieval. Because of a better visuability, we use artificially arranged 2-dimensional multi-modal input data, where each mode is gaussian distributed. Modes are potentially overlapping and globular, i.e. consist of further modes, called sub-modes or sub-clusters.

4.1.1 HRCL

Figure 2 shows that HRCL with given radius $r = 0.25$ is capable of detecting all three existing globular clusters of an input data set consisting of $1,800$ data samples. Shown are the mode data as well as the HRCL neurons as circles together with their hypercubical environments $U_c(0.25)$. HRCL training starts with 8 input neurons at the 0^{th} hierarchical level. 5 neurons are pruned during training that requires 634 training steps. After training, globular clusters can be represented by one neuron each placed at cluster centroids. Each globular cluster consists of 5 sub-modes that can be detected by HRCL at the next HRCL hierarchical refinement step (cf. fig. 3): HRCL reduces after $70 - 80$ training steps $5 - 6$ initial neurons at the 1^{st} hierarchy to 5 neurons that are placed to the sub-cluster's centroids. Thus, each sub-mode can be described by one neuron or cluster centroid only. The hierarchical refinement follows a third hierarchy, where the distributions of each sub-mode is approximated (not shown).

4.1.2 SOM

Contrary to HRCL, a standard 10×10 SOM, trained during $7,000$ training steps, can not achieve a real vector quantization: the SOM puts its neurons not only to locations of high probability density, but also to sparcely coded parts of the vector space, to be seen between the existing input modes at figure 4. This is mainly due to the fact,

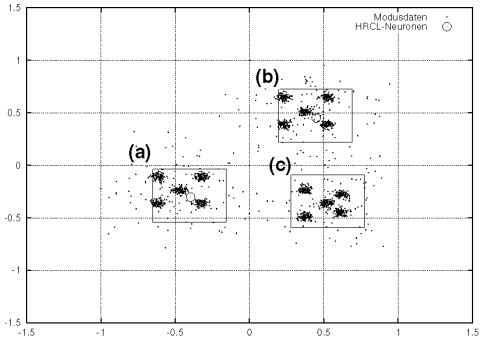

Figure 2. HRCL clustering results at 0^{th} HRCL hierarchy.

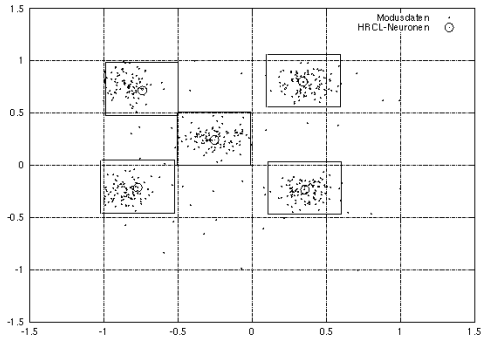

Figure 3. HRCL clustering results at 1^{st} HRCL hierarchy: Detection of sub-clusters of cluster (a) at fig. 2.

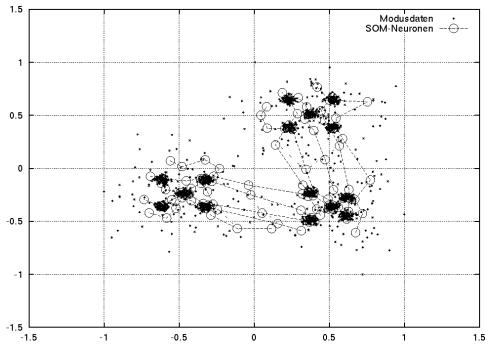

Figure 4. SOM clustering results.

because the SOM is not able to prune its neurons while or after training. For the same reason, input modes are described by more than one neuron in general (cf. figure 4).

4.1.3 Single-pass clustering

In our last experiment, we use the single-pass clustering: This method processes the input data in serial order, i.e. the first input data becomes the first cluster centroid. The second input is mapped to the first cluster, if a threshold distance r is not exceded, otherwise it determines a second cluster (centroid). This procedure is continued for all following input samples. Our clustering results could confirm statements in [5] which say that the method is highly dependent on the order of input fed into the system: It could be seen that allthough each single-pass clustering run can place its centroids inside the existing modes, thus describing well the mode's distributions, some centroids were placed at outliers and the centroid positions variied considerable between different runs (not shown here). Several other neural networks were used for clustering a similar input, e.g. the NG or GNG or the *growing grid*, whose results can not be shown here. None of these were able to represent globular clusters by exactly one neuron put at its centroids.

The shown results seem to validate our assumption that HRCL is better able than other (neural or statistical) clustering methods to describe multi-modal input data. In contrast to the latters, HRCL supplies a mult-resolution view of the input: Globular clusters can be detected and represented by exactly one cluster centroid. Further HRCL hierarchies can deliver "zoomed-in views" of detected clusters until a further zooming can not detect further "details". The rather theoretical results of this section shall be extended to "naturall" input data in the next section that will put more stress on the retrieval effects of the obtained clustering.

4.2 HRCL retrieval results

The retrieval results are evaluated using three different input document sets: the CISI test data collection, a confined HTML document set consisting of 242 input documents as well as HTML documents stemming from Yahoo sub-directories.

4.2.1 CISI test data collection

In order to evaluate the retrieval results of the HRCL clustering system, we use the known CISI test data collection, whose $1,460$ test documents are mainly engaged in "digital library" topics and come with 112 test queries. The input data is coded using the known term frequency-inverse document frequency (TF-IDF) indexing in combination with the Porter stemmer [15], translating the test queries and documents into vector space model. Resulting high-dimensional (document and query) fingerprint vectors are compressed by the *latent semantic indexing (LSI)*, resulting in 20-dimensional vectors. We use the *singular value decomposition (SVD; [16])* as efficient implementation of LSI. HRCL, initiated with input radius $r = 0.5$, yields 3 hierarchies with 8 detected clusters at the 0^{th}, 47 sub-clusters at the 1^{st} and further 41 sub-subclusters at the last hierarchy. For each detected HRCL cluster c at every hierarchy *Voronoi sets* R_c are calculated, determining those input vectors ξ that are more adjacent to the corresponding cluster centroid w_c than to any other cluster centroid w_i of the same hierarchical level: $R_c := \{\xi \mid \|\xi - w_c\| < \|\xi - w_i\| \; \forall i \neq c\}$ To obtain *precision* and *recall* retrieval measures

(cf. [2]), query-to-document vector comparisons (using cosine correlations) are carried out using Voronoi set R_c of cluster c whose cluster centroid w_c has maximum correlation with the query vector. The document vectors $\xi \in R_c$ are ordered relative to their correlations and precision-recall measures are finally calculated for the first 10 %, 30 %, 50 %, 70 % and 100 % of the ordered set. The resulting values are connected and rendered as precision-recall graph depicted in figure 5.

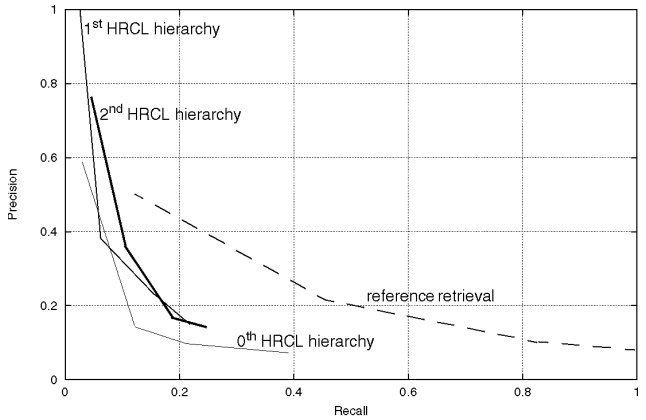

Figure 5. HRCL retrieval results using CISI test data collection.

It can be seen with figure 5 that the HRCL retrieval results at the 0^{th} hierarchical step are always worse than the optimal reference retrieval (that uses the total input vector space and thus calculates $1,460 \times 112$ document-to-query correlations). But above all precisions for the 1^{st} and 2^{nd} HRCL hierarchy (regarding the first 10 % of the clustered documents) are yet above the reference (whose first 10 % of retrieved documents obtain a maximum precision of about 0.5). Calculating retrieval measures for the found documents only, the HRCL precision reaches almost maximum value. Secondly, the overall retrieval can (slightly) be improved with ascending hierarchical levels. And finally, HRCL retrieval (using the already clustered and evaluated document set) can tremendously be speeded up: HRCL retrieval needs a few seconds, whereas the reference retrieval (that still has to calculate and order $163,520$ correlations) needs up to several hours. This suggests to divide document retrieval using the local HTML data collection of figure 1 into two parts: The HRCL clustering has to be accomplished prior to searching and can be run at low load times as background process. After that, user searches may use the obtained clustering results that speeds up the document lookup alone.

4.2.2 Local HTML document set

The HRCL retrieval is further compared with a 5×5 SOM, using a confined local HTML document set consisting of 242 16-dimensional document vectors. For term indexing we use the *character trigram coding* (cf. [18]) in combination with the *Generalized Generalized Hebbian Algorithm (GGHA; [17])* as compression. The SOM training is finished after $10 \times 242 = 2,420$ training steps. Documents that are mapped on the SOM cluster which contains the document to be searched for as well as all documents of the neighboring SOM neurons are defined as "found documents". The SOM precision yields a value of 0.5, the recall equals 0.26. On the other hand, 5 initial HRCL neurons with input radius 0.4 and hypercubical overlap of 0.2 are trained during 399 training steps at the 0^{th} HRCL hierarchy, of which 3 neurons are pruned during training. HRCL generates 2 hierarchies with 2 detected clusters at the 0^{th} and 7 sub-clusters at the 1^{st} hierarchical level. The retrieval considers only those documents as "found" that are located inside the same HRCL cluster of the same HRCL hierarchy than the document to be searched for. The 0^{th} HRCL hierarchy yields precision 0.22 and recall 0.85, the 1^{st} hierarchy yields precision 0.6 and recall 0.11. Several other experiments using different HRCL input radii (and different hypercubical overlaps) were performed and are presented at table 1.

Table 1. HRCL retrieval results using 242 HTML input documents.

HRCL results								
radius	0.05		0.1		0.3		0.4	
overlap	0.1		0		0.2		0.2	
hierarchy	0	1	0	1	0	1	0	1
precision	0.28	—	0.33	—	0.25	0.42	0.22	0.6
recall	0.78	—	0.7	—	0.78	0.3	0.85	0.11

The retrieval results show that HRCL is always able to outperfom the SOM's recall measure at the 0^{th} hierarchical level: Each of the recall values of HRCL using different radii is better than 0.26 obtained by the SOM. With the improved recall, precision of the 0^{th} HRCL hierarchy must suffer. However, the precision can be improved at the next hierarchy and can even outperform the SOM's precision with HRCL input radius 0.4 (together with hypercubical overlap of 0.2; see table 1).
This points out that the 0^{th} HRCL hierarchical level (lowest resolution) can deliver more relevant documents than the SOM clustered document set. Additionally, the user is able with HRCL – contrary to the SOM – to select an appropriate sub-cluster of the 1^{st} HRCL hierarchy that contains documents he estimates relevant to increase precision. The "zoomed view" of the 1^{st} HRCL hierarchy may then even be able to exceed the general SOM precision.

4.2.3 Yahoo directory

For our last experiment, we use $5 \times 380 = 1,900$ HTML documents obtained from the Yahoo sub-directories "benchmarks", "travelogues", "museums", "programming languages" and "research groups". The input documents are indexed and compressed analogous to the CISI collection, resulting in $1,900$ 76-dimensional document vectors with which HRCL using radius 0.15 is trained. HRCL reduces after 986 training steps 9 initial to 6 final neurons of hierarchy 0 and generates further 2 hierarchies, detecting further 10 sub-clusters and 6 sub-subclusters. In order to evaluate the quality of the resulting HRCL clustering, the coherence of each cluster is measured. To achieve this, all document vectors of Voronoi sets R_c of cluster c have to be ordered relative to their distances to the cluster centroid vector. After that, the "cluster coherence" can be defined as the percentage of the first 25 % of documents of the Voronoi set belonging to the same Yahoo directory than the cluster centroid document. The average coherence for the 0^{th} HRCL hierarchy yields 86.3 % and 81.2 % for the 1^{st} hierarchy. At the 0^{th} hierarchy, 3 out of 5 Yahoo categories can be detected and represented by exactly one neuron, while directory "benchmark" generates 3 different HRCL clusters at the same hierarchy. Remaining input directory "programming languages" generates 3 sub-clusters at the 1^{st} hierarchy. Figure 6 shows a part of the generated Internet catalogue: the hierarchical refinement of the 0^{th} cluster of the 0^{th} hierarchy. The cluster can be

resolved to 8 sub-clusters and further 6 sub-subclusters. To be seen is the (abreviated) URL of the cluster centroid document at the 1^{st} row, the centroid document's title at the 2^{nd} row, the number of clustered documents at the 3^{rd} and the 3 document vectors closest to the cluster centroid vector at following rows. Indented are all (sub-)cluster centroids of the 0^{th} and 1^{st} hierarchical refinement of cluster 0.

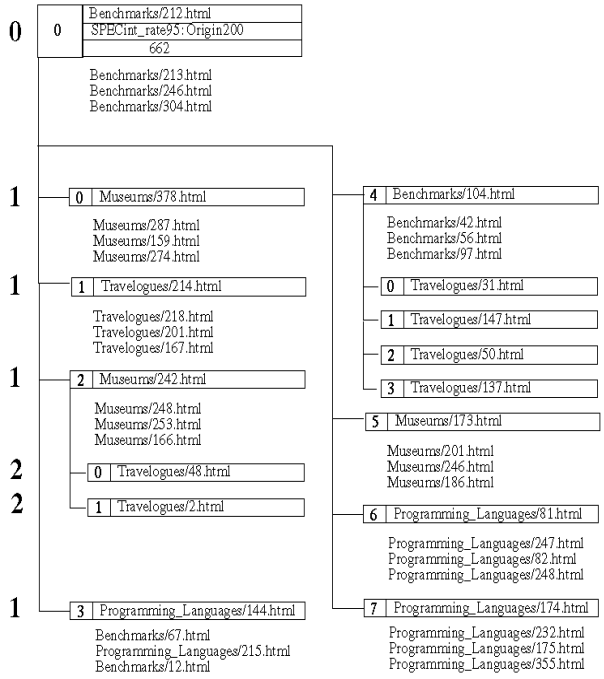

Figure 6. Part of the automatically generated Internet catalogue using HRCL clustering and Yahoo input documents.

Further results, using $5,508$ HTML documents out of 11 Yahoo sub-directories (compressed to dimension 46) could confirm the above results: HRCL can then detect 9 out of 11 input categories (by one neuron each for hierarchy 0), 2 remaining ones are distributed among 5 sub-clusters of the 2^{nd} hierarchy each. Cluster coherences can yield 89 % for the 0^{th} HRCL hierarchy.

5 CONCLUSION

This work presented a feasible way to automatically generate a Yahoo-like local Internet directory. This has been managed by the *Hierarchical Radius-based Competitive Learning (HRCL)* neural network that has been developed by the authors. Clustering as well as retrieval results showed that the method is able to surpass neural as well as conventional statistical clustering methods. Finally, dividing document retrieval into two parts – clustering and searching – user query search alone can significantly be accelerated.

ACKNOWLEDGEMENTS

This work has been undertaken under the project *OASIS* (*"Open Architecture Server for Information Search and Delivery"*) that was granted as project no. PL96 1116 in frames of the EU INCO Copernicus Programme. Further information concerning the OASIS project may be obtained at `http://www.oasis-europe.org`.

REFERENCES

[1] D. Sullivan (Editor): *Search Engine Watch*, Mecklermedia Corporation, Westport, CT, 1996–1998, `http://www.searchenginewatch.com`

[2] C. J. van Rijsbergen: *Information Retrieval*, 2nd Edition, Butterworths, London, 1979
`http://www.dcs.glasgow.ac.uk/Keith/Preface.html`

[3] R. Ferber: *Information Retrieval*, GMD-IPSI, August 1996, `http://www.darmstadt.gmd.de/~ferber/ir-bb/frame.html`

[4] J. B. McQueen: Some methods of classification and analysis of multivariate observations, in *Proc. of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281-297, 1967

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

[6] R. O. Duda, P. E. Hart: *Pattern Classification and Scene Analysis*, John Wiley & Sons Inc., New York, 1973

[7] C. T. Zahn: Graph-theoretical methods for detecting and describing Gestalt clusters, *IEEE Transactions on Computers*, **C 20**, 68-86, 1971

[8] S. Y. Lu, K. S. Fu: A sentence-to-sentence clustering procedure for pattern analysis, *IEEE Transactions on Systems, Man and Cybernetics*, **SMC 8**, 381-389, 1978

[9] T. Kohonen: Self-organized Formation of Topology Correct Feature Maps, *Biological Cybernetics*, 43:59-69, 1982

[10] T. Kohonen: *Self-Organization and Associative Memory*, Springer Verlag, 1984

[11] T. M. Martinetz, K. J. Schulten: A "neural-gas" network learns topologies, in T. Kohonen, K. Mäkisara, O. Simula and J. Kangas (Eds.): *Artificial Neural Networks*, pp. 397-402, North-Holland, Amsterdam, 1991

[12] B. Fritzke: A growing neural gas network learns topologies, in G. Tesauro, D. S. Touretzky and T. K. Leen (Eds.): *Advances in Neural Information Processing Systems* 7, pp. 625-632, MIT Press, Cambridge MA, 1995

[13] M. Bessonov, U. Heuser, I. Nekrestyanov, A. Patel: Open Architecture for Distributed Search Systems, in: H. Zuidweg, M. Campolargo, J. Delgado, A. Mullery (Eds.): *Intelligence in Services and Networks*, Lecture Notes in Computer Science 1597, pp. 55-69, Springer Verlag, 1999

[14] A. Patel, L. Petrosjan, W. Rosenstiel (Eds.): *OASIS - Distributed Search System in the Internet*, St. Petersburg State University Published Press, St. Petersburg, 614 pp., 1999

[15] M. F. Porter: An algorithm for suffix stripping, *Program*, 14(3):130-137, 1980
`http://www.muscat.com/~martin/stem.html`

[16] M. Berry, T. Do, G. O'Brien, V. Krishna, S. Varadhan: *SVDPACKC (Version 1.0) User's Guide*, Univ. of Tennessee Tech. Report CS-93-194, April 1993 (Revised October 1996)
`http://www.netlib.org/svdpack/svdpackc.tgz`

[17] H. Hyötyniemi: Constructing Non-Orthogonal Feature Bases, *ICANN'96*, Washington DC, pp. 1759-1764, June 1996

[18] J. Mayfield, University of Maryland, UMBC: *Research on N-Grams in Information Retrieval*,
`http://www.cs.umbc.edu/~mayfield/ngrams.html`