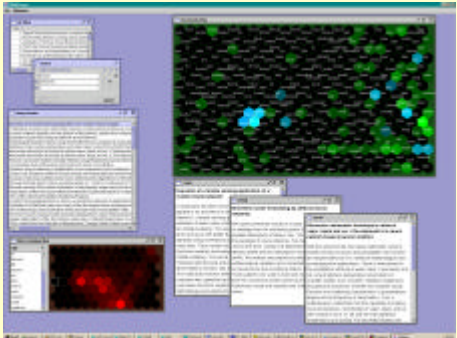
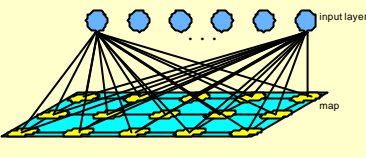
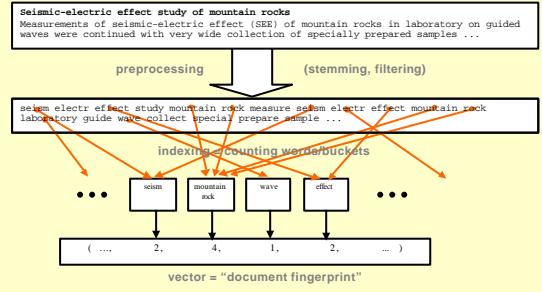
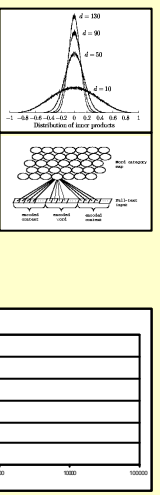
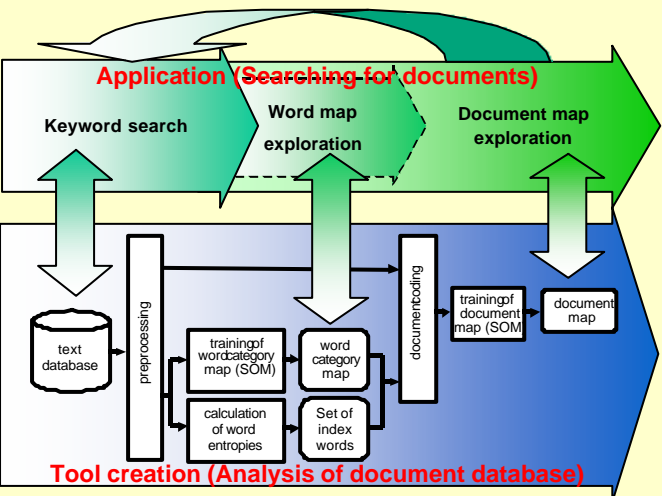
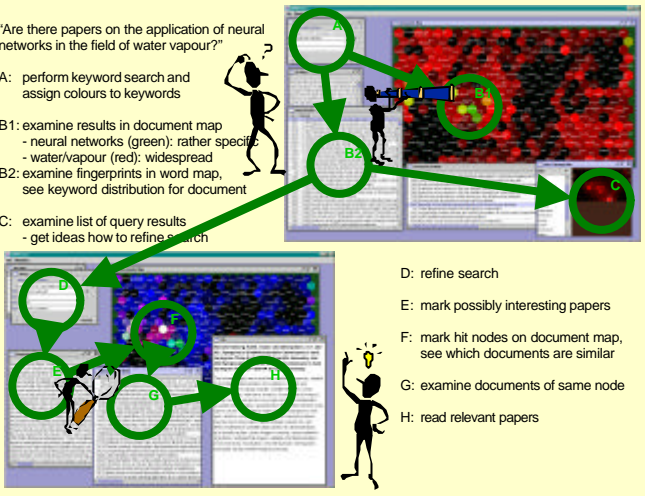


Interactive Text Retrieval Based on Document Similarities

A. Nürnberger⁽¹⁾, A. Klose, R. Kruse, G. K. Hartmann⁽²⁾, M. L. Richards

<p>Motivation: Limitations of standard text retrieval methods.</p> <p>Idea: Grouping/arranging documents based on a similarity measure. This helps the user to navigate through similar documents. The navigation, especially the search for the "first appropriate document", should be supported by conventional keyword search methods.</p> <p>Realisation: Development of an interactive software tool based on self-organising maps: - interactive associative search - visualization for better overall view</p> 	<p>Self-organizing maps (SOM)</p> <p>Artificial neural network model to project high-dimensional data vectors to lower dimensional data space (usually two dimensions) under preservation of neighbourhood relations.</p> <p>General structure: The SOM consists of an n-dimensional input layer of neurons and a map usually defined by a regular grid of neurons. The weight vector w_i from a neuron i in the map to the input layer represents a prototype. During the learning process a mapping from the input data space to the map is created so that similar input vectors are close to each other and dissimilar input vectors far from each other.</p>  <p>Learning method (competitive learning): The weights (prototypes) w_i are randomly initialised. The adaptation of the model vectors is usually carried out by a sequential regression process. All vectors of the input data set are processed consecutively. Let $t = 1, 2, \dots$ be the step index: For each input vector $x(t)$, first the winner index c (best match) is identified by the condition: $\forall i: \ w_i - x(t)\ \leq \ w_c - x(t)\$ Then the assigned vector w_c is adjusted such that for the next presentation of the same input vector an even higher degree of similarity will be obtained. Furthermore, all vectors i in a neighbourhood of the winner neuron c are adjusted: $\forall i: w_i = w_i + v(c, i) \cdot d \cdot (w_i - x(t))$ where $v(i, c)$ is a neighbourhood function, which decreases with the distance of the nodes i and c on the map and d is a learning rate.</p>
<p>Document preprocessing and coding</p> <p>After a preprocessing step (stemming and filtering) a list of bins is created for the document database (see "Defining the bins"). Based on these bins all documents are coded as vectors. These vectors can be seen as the fingerprints of each document.</p>  <p>Arranging the documents: The document map</p> <p>The fingerprints of the documents are used as input vectors for a two-dimensional self organising map: The document map. After training of this map, documents with similar fingerprints are close to each other (possibly assigned to the same neuron). So, when a user has discovered a document of interest on the map, he or she can search the surrounding area.</p>	<p>Defining the bins (Creating a word category map)</p> <p>a) Grouping similar words according to 3-word-contexts</p> <ul style="list-style-type: none"> Encode words as high-dimensional random vectors (Ritter and Kohonen, 1989) → Encoding does not imply any word ordering: the vectors are "quasi-orthogonal" For each word calculate the expectation value vectors e_i and e_o over all random vectors of enclosing words (in all documents) and create a context vector v based on these vectors and the random vector w of the considered word: $v = \{e_i, e_o\}$ (Honkela et al., 1996) → Words that occur in similar contexts have similar expectation values and therefore similar vectors v Map these vectors v_i to two dimensions using a self organising map → Words that are used in similar contexts are mapped to the same (or nearby) neuron on the word category map. Each neuron of the resulting map is used as a bin for fingerprint counting <p>b) Selection of index words based on entropy</p> <ul style="list-style-type: none"> Calculate entropy for each word (Lochbaum and Streeter, 1989). This is a measure for the importance of this word: $W(w) = 1 + \frac{1}{\ln(m)} \sum_{i=1}^m p_i(w) \cdot \ln(p_i(w))$ with $p_i(w) = \frac{n_i(w)}{\sum_{i=1}^m n_i(w)}$ $n_i(w)$: frequency of word w in document i m: number of documents Choose words that have a high entropy relative to their overall frequency Use these words as bins for fingerprint counting 
<p>System overview</p> 	<p>Sample search</p> <p>"Are there papers on the application of neural networks in the field of water vapour?"</p> <p>A: perform keyword search and assign colours to keywords</p> <p>B1: examine results in document map - neural networks (green): rather specific - water/vapour (red): widespread</p> <p>B2: examine fingerprints in word map, see keyword distribution for document</p> <p>C: examine list of query results - get ideas how to refine search</p> <p>D: refine search</p> <p>E: mark possibly interesting papers</p> <p>F: mark hit nodes on document map, see which documents are similar</p> <p>G: examine documents of same node</p> <p>H: read relevant papers</p> 

Contact:

A. Nürnberger ⁽¹⁾
 FIN IWS
 Universitätsplatz 2
 D-39106 Magdeburg

Prof. Dr. Gerd K. Hartmann ⁽²⁾
 Max-Planck-Institut für Aeronomie
 Max-Planck-Str. 8a2
 D-37191 Katlenburg-Lindau, Germany