

Concept Based Adaptive IR Model Using FCA-BAM Combination for Concept Representation and Encoding

Rajapakse,R.K and Denham, M

rohan/mike@soc.plym.ac.uk

Centre for Neural and Adaptive Systems, University of Plymouth, UK

Abstract. The model described here is based on the theory of Formal Concept Analysis (FCA). Each document is represented in a Concept Lattice: a structured organisation of concepts according to a subsumption relation and is encoded in a Bidirectional Associative Memory (BAM): a two-layer heterogeneous neural network architecture. The document retrieval process is viewed as a continuous conversation between queries and documents, during which documents are allowed to learn a consistent set of significant concepts to help its retrieval. A reinforcement learning strategy based on relevance feedback information makes the similarity of relevant documents stronger and nonrelevant documents weaker for each query.

1. Introduction

The primary objective of our research is to improve the effectiveness of the document retrieval. Our approach is two-fold:

1. investigate for a better scheme for document (and query) representation. We attempt to make use of the FCA-BAM combination to achieve this goal.
2. attempt to make use of a reinforcement learning strategy based on relevance feedback information for learning well representative set of concepts for document representation.

Almost all the existing IR models make use of either single terms (keywords) or phrases (two or more adjoining terms) to represent the basic unit of matching: a *concept* [17]. Exceptional to these are the network models that try to learn the implicit relations between keywords to build a form of implicit concepts to assist the IR task and the Knowledge based techniques in which additional (related) keywords are incorporated with the keywords extracted from the text to enhance the query or document(s). Most of these approaches do not extract concepts and their relationships explicitly straight from the text and also do not store them in a hierarchical structure representing specificity-generality relationships between concepts which we suppose is important for deciding the relevancy of a document. In addition the said hierarchical structure must support efficient access to those specific-generic concepts as desired.

In our work, great deal of effort was extended to the direct extraction of meaningful ideas/concepts from text and their representation in a formal framework to assist the IR task. The major contribution of our work is the investigation of the applicability of concept lattices, that are based on the theory of the formal concept analysis and lattice theory, to the representation of concepts/ideas expressed in natural language (English) text, and encoding such concept lattices in Bidirectional Associative Memories (BAMs) in order to effectively and efficiently manipulate the concepts captured to support Information Retrieval. An interesting feature of concept lattices is that the concepts are hierarchically organized according to a subsumption order relation, that defines a specificity-generality relationship between concepts. BAMs have been recognised as a simple neural network that is able to learn a concept lattice in its two-layered architecture. Once a concept lattice is encoded in a BAM, it can directly give the most specific or most generic concept of the concept lattice it has learnt for a given set of interested objects or attributes, by presenting them to the corresponding layer(s) of the BAM. This avoids the complex searching and traversing otherwise required for accessing concepts.

Our model views the document retrieval process as a continuous conversation between queries and documents. Through these conversations, it attempts to learn a consistent set of significant concepts for each document resulting queries similar to already seen ones to retrieve the documents that were retrieved for those similar queries and judged as relevant in the past (improves recall) and conversely queries similar to already seen ones not to retrieve the documents that were retrieved for those similar queries but judged as not relevant (improves precision). This is achieved through a reinforcement learning strategy based on relevance feedback, in which the document representation is improved to make the similarity between a query and a retrieved relevant document stronger and that between a query and a retrieved nonrelevant document weaker.

We present the theoretical background for the techniques used (FCA & BAM) and discuss the suitability of such a representation scheme for Information Retrieval. The analogy between the representation of formal concepts in a structured manner in a concept lattice and the representation of ideas/concepts in human brain in the process of human understanding is briefed. The IR process, including what is (what concepts) matched with what (concepts) in the document, how a similarity value is computed and how the relevance feedback is used for reinforcement learning are detailed.

The work reported here is ongoing research work. Preliminary results using the proposed model and reinforcement learning process are encouraging. Future work will include making further improvements to the proposed model and finally evaluating its performances by comparing its results with published results on public document collections

2. Formal Concept Analysis (FCA)

Formal Concept Analysis was first proposed by Rudolf Wille in 1982 [19,11] as a mathematical framework for performing data analysis. It provides a conceptual analytical tool for investigating and processing given information explicitly. Such data is structured into units, which are formal abstractions of “concepts” of human thought allowing meaningful and comprehensible interpretation. FCA models the world as being composed of *objects* and *attributes*. It is assumed that an incident relation connects objects to attributes. The choice of what is an object and what is an attribute is dependent on the domain in which FCA is applied. Information about a domain is captured in a “formal context”. A formal context is merely a formalization that encodes only a small portion of what is usually referred to as a “context”.

A *Formal Context* $K = (G, M, I)$ consists of two sets G (set of objects) and M (set of attributes) and a relation I between G and M . A *Formal Concept* is defined on a Formal Context as a pair of sets (A, B) where A is a set of objects and B is a set of attributes, in which attributes in A are maximally possessed by the set of objects in B and the objects in A are the maximal set of objects possessing the set of attributes in B . A formal definition is given below.

A pair (A, B) of sets where $A \subseteq G$ and $B \subseteq M$ is a formal concept if :
 $A' = B$ and $B' = A$; (This is called the completeness constraint) (1)

where, $A' = \{ m \in M \mid gIm \text{ for all } g \in A \}$ (i.e. the set of attributes common to all the objects in A) AND $B' = \{ g \in G \mid gIm \text{ for all } m \in B \}$ (i.e. the set of objects which have all attributes in B). gIm means “ g is related to m ” (e.g. a binary relation).

The set of all concepts of a context (G, M, I) which consists of all pairs (A, B) where $A \subseteq G$ and $B \subseteq M$ s.t. $A = B'$ and $B = A'$ is denoted by $B(G, M, I)$.

FCA models the specificity and generality relationships between two related concepts by means of sub-super relationship, which is formally defined as:

If (A_1, B_1) and (A_2, B_2) are concepts of a context, then (A_1, B_1) is called a subconcept of (A_2, B_2) , if $A_1 \subseteq A_2$ (equivalently $B_1 \supseteq B_2$). In this case (A_2, B_2) is a superconcept of (A_1, B_1) and this sub-super concept relation is written as $(A_1, B_1) \leq (A_2, B_2)$, i.e. a subconcept is a concept with less objects than any of its superconcepts (equivalently, a subconcept is a concept with more attributes than any of its superconcepts).

2.1 Concept Lattice

A set of all concepts of the context (G, M, I) (denoted by $B(G, M, I)$) when ordered with the order relation \leq (a subsumption relation) defined above forms a *concept lattice* of the context and is denoted by $\underline{B}(G, M, I)$ [11].

A lattice is an ordered set V with an order relation in which for any given two elements x and y , the supremum and the infimum elements always exist in V [11].

Furthermore, such a lattice is called a *complete lattice* if supremum and infimum elements exist for any subset X of V [19,11]. The fundamental theorem of FCA states that the set of formal concepts of a formal context forms a complete lattice [11,9,19]. This complete lattice, which is composed of *formal concepts* is called a *concept lattice*.

Join/Meet Concepts

A Concept lattice can be visualized as a graph with nodes and edges/links (fig 1). Concepts at the nodes from which two or more lines run up are called *meet* concepts (i.e. nodes with more than one parent) and concepts at the nodes from which two or more lines run down are called *join* concepts (i.e. nodes with more than one child). An interesting feature of concept lattices is that we do not need to know explicitly all of these meet and join concepts to build the complete lattice, as they can be inferred from other concepts. A join (parent) node is inferred given all of its child nodes and a meet (child) node is inferred given all of its parent nodes. (See figure 1)

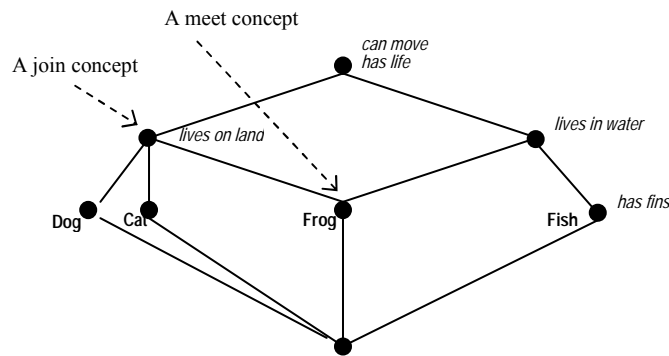


Fig. 1. example concept lattice to illustrate join and meet concepts

A Join concept groups objects sharing the same attributes and a meet concept separate out objects that have combined attributes from different parents (groups of objects). Each of these join and meet concepts creates a new sub- or super- category or class of a concept.

3. Bidirectional Associative Memories (BAMs)

Based on the early associative memory models [1,14], Kosko [15,16] proposed a bi-directional associative neural network called Bidirectional Associative Memory (BAM). A BAM consists of two layers of neurones. The states (activities) of the neurones in the first Layer (X) are denoted by x_i ($i=1, \dots, k$) and in the second layer (Y) by y_j ($j=1, \dots, l$) where k and l are the number of neurones in the layers. The states x_i and y_j can be encoded either a binary (0 or 1) or a bipolar (+1 or -1) encoding. Each (i^{th}) neuron of the first layer is connected to each (j^{th}) neuron of the second layer, by a connection weight. A number of different weighting schemes can be found in the

literature for setting the connection weights. Amongst are the one proposed by Kosko[15], the originator of BAMs and that proposed by Radim Bělohlávek [2].

A real threshold θ_i^x (θ_j^y) is assigned to the i^{th} neuron of the first layer and the j^{th} neuron of the second layer, respectively.

3.1 Dynamics of BAMs

Given a pair $\langle X, Y \rangle = \langle \langle x_1, \dots, x_k \rangle, \langle y_1, \dots, y_l \rangle \rangle \in \{0,1\}^k \times \{0,1\}^l$ of patterns of signals, the signal X is fed to the first layer to obtain a new pair $\langle X, Y' \rangle$, then Y' to the second layer to obtain $\langle X', Y' \rangle$, and so on. The dynamics is given by the formulas:

$$y'_i = \begin{cases} 1 & \text{for } \sum_{i=1}^k w_{ij} x'_i > \theta_j^y \\ y_j & \text{for } \sum_{i=1}^k w_{ij} x'_i = \theta_j^y \\ 0 & \text{for } \sum_{i=1}^k w_{ij} x'_i < \theta_j^y \end{cases} \quad x'_i = \begin{cases} 1 & \text{for } \sum_{j=1}^l w_{ij} y'_j > \theta_i^x \\ x_i & \text{for } \sum_{j=1}^l w_{ij} y'_j = \theta_i^x \\ 0 & \text{for } \sum_{j=1}^l w_{ij} y'_j < \theta_i^x \end{cases} \quad (2)$$

The pair of patterns $\langle X, Y \rangle$ is called a stable point if the states of neurones, when set to $\langle X, Y \rangle$, do not change under the above defined dynamics. Using appropriate energy function, Kosko [16] proved that such a network is stable for any weights w_{ij} and any thresholds θ_i^x , θ_j^y . Stability means that given any initial pattern $\langle X, Y \rangle$ of signals, the network eventually stops after a finite number of steps (feeding signal from layer to layer back and forth).

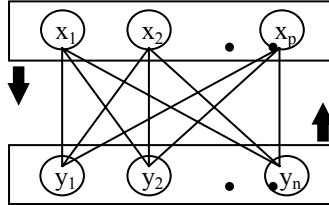


Fig. 2. Structure of a BAM

The aim of learning in the context of associative memories is to set the parameters of the network so that a prescribed training set of patterns is related in some way to the set of all stable points. Usually all training patterns have to become stable points (this is not always the case with learning concept lattices). Kosko proposed a kind of Hebbian learning, by which the weights w_{ij} are determined from the training set $T =$

$\{\langle X_p, Y_p \rangle | p \in P\}$ by

$$w_{ij} = \sum_{p \in P} bip(x_i^p) \cdot bip(y_j^p) \quad \text{where } bip() \text{ maps } 1 \text{ to } 1 \text{ and } 0 \text{ to } -1, \text{ i.e. changing}$$

the binary encoding to a bipolar one and $P = \{1, 2, 3, \dots, \text{no. of patterns in } T\}$. Thresholds are set to 0 [6].

3.2 BAMs for Storing Concept Lattices

In [2] Radim Belohlávek showed that BAMs can learn concept lattices. He observed that there are BAM stable points that cannot be interpreted as concepts. This raised the most important question "weather there is a BAM corresponding to each concept lattice $B(G,M,I)$ such that the set of all concepts of $B(G,M,I)$ is precisely the set of all the stable points of the BAM ?". Using the weight computation given below, Bělohlávek proved that there is a BAM, given by the weights W and thresholds θ such that $Stab(W, \theta) = \{ \langle A, B \rangle \mid \langle A, B \rangle \in B(G,M,I) \}$, corresponding to the concept lattice given by the context $\langle G, M, I \rangle$ with G and M finite.

Bělohlávek 's weight computation formula is given by

$$w_{ij} = \begin{cases} 1 & \text{if } \langle g_i, m_j \rangle \in I \\ -q & \text{if } \langle g_i, m_j \rangle \notin I \end{cases} \quad \text{For } i=1, \dots, k, j=1, \dots, l \quad (3)$$

where $q = \max \{k, l\} + 1$. All the thresholds are set to $-1/2$.

We use this weighting scheme in our implementations described below to train BAMs with concepts lattices.

A Training Set for a BAM to learn a Concept Lattice

A training set T consists of a set of concepts in the form (A, B) , where elements of A comes from the set of objects G and elements of B from the set of attributes M of the context (G, M, I) . Here the set G contains all the objects necessary to define the extents (A) of the concepts (A, B) in the training set and M contains all the attributes necessary to define the intents in the training set. This means that the elements in T are defined on (G, M, I) . A conceptually consistent training set however needs its training patterns to obey the fundamental rule (completeness constraint) of the formal concept: $A' = B$ & $B' = A$ which ensures that the set of formal concepts in the training set are storable in the BAM.

Any given set of arbitrary concepts (a Training set T) which satisfies the above condition defines a concept lattice on the context (G, M, I) formed by all the objects and attributes of the concepts in T . This training set is a subset of $B(G, M, I)$ (set of all the concepts of the context (G, M, I)). It also is a subset of the concept lattice $\underline{B}(G, M, I)$ of the context (G, M, I) .

What Actually a BAM Learns and Returns?

Given a set of training pairs of objects and attributes that satisfy the completeness constraint (i.e. they are formal concepts in a given context), a BAM learns the underlying concept lattice. As mentioned before, join and meet concepts that are inferred by the patterns in the training set are automatically detected and learnt by the BAM. For example, given four training patterns, one for each object: dog, cat, frog and fish (with all of their corresponding attributes) the concept lattice given in fig.1 is derived.

Unlike purely feed-forward neural network architectures, BAMs can accept input patterns from either layer. We can present a pattern with objects to the first layer (referred to as the object layer) or a pattern with attributes to the second layer (referred to as the attribute layer). The BAM returns the most specific concept containing all the objects of the input pattern in the first case and the most generic concept containing all the attributes of the input pattern in the second case. We use this interesting property during the extraction of candidate concepts from query and documents BAMs to match between concepts (as described in section 5.2).

Stable Points

An arbitrary pair consisting of a set of objects and a set of attributes extracted from text is not necessarily a stable point in a BAM. Also an actual training pattern used for training a BAM with a concept lattice does not necessarily become a stable point in the BAM. This is because:

1. a set of objects and set of attributes initially extracted from text might not be a formal concept as it might not comply to the completeness constraint (1).
2. A given input to a BAM may lead to retrieving an inferred *Join* or *Meet* concept instead of a concept(s) in the training set that contributed to infer the join or the meet concept.

4. Towards Concepts/Ideas Expressed in Natural Language

A number of questions have to be addressed before employing these techniques into IR tasks. These include: what is an object; what is an attribute; what is a concept/idea; how can we extract objects, attributes and concepts from textual material; what is the analogy between the order relationship defined for formal concepts and the ideas/concepts extracted from textual documents; what is the role of join/meet concepts in human understanding of natural language text etc. This section attempts to answer these questions.

4.1 Abstracting Ideas/Concepts in Human Understanding

The theory of concept lattices has been founded [11,19] based on a traditional understanding of concepts by which a concept is determined by its extent and intent. The extent of a concept (e.g. *DOG*) is the collection of all objects covered by the concept (the collection of all *dogs*), while the intent is the collection of all attributes (e.g. *to bark*, *to be a mammal*) covered by the concept. This interpretation of a concept can be directly employed in representing concepts expressed in natural language. The formation of an idea or a concept in the human mind during the understanding of natural language text is initially triggered by the objects (physical or conceptual) and attributes (properties of objects) in the text followed by the overall context of the subject being read and the reader's background knowledge of the subject.

The Two Entities: *Objects* and *Attributes*

In general, an object corresponds to the subject of the context, and attributes modify the meaning of the object to express the context in which the object is being used. For instance, a particular set of attributes of the object *DOG* may support the context of say *eating patterns of dogs*, while certain other set of attributes may support say the *sleeping patterns of dogs*. FCA captures these two important aspects/features, the *subject* and the *context* in its two entities *objects* and *attributes* to formulate an idea/concept. This makes FCA suitable for formulating and manipulating “human thoughts” (concepts) within computer systems.

4.2 Similarity of super-sub order relationship in formal concepts and natural ideas/concepts

A sub-concept in FCA, defined as a concept with less objects and more attributes than its superconcept(s) means we need more attributes to define something specific compared to the less attributes needed to define something generic. Ideas/concepts stored in human mind may be structured in a similar manner, whereby we need more detail to learn a more specific idea/concept. For instance to define a *bird* we need to say it *can fly* and it *has a beak*, in addition to the information necessary to say that its an *animal*. However the frequently used generic attributes (in this case the attributes to specify bird is a living animal) are not usually used to express an idea during normal human conversations and writing. They are implicit. In general human brain has gained the necessary background knowledge in understanding frequently used common ideas when expressed just by the main object(s) of the subject. For instance, we never try to define what an animal is during conversations, instead we simply use the term “*animal*”. At some point during our learning process (implicit or explicit), we have absorbed all the necessary attributes to understand what an animal is. It is obvious however that encoding concepts/ideas in a computer requires all the information necessary to distinctly identify a particular idea/concept to be explicitly specified. These background general ideas/concepts are analogous to the superconcepts and the specific sub-categories/concepts of them are analogous to the subconcepts defined in the FCA formalization.

4.3 The role of Join & Meet concepts in Human understanding

Categorising common objects (or ideas) together is a natural phenomenon in the human understanding. If you are asked to name few animals you can give a vast number of different animals as examples for animals. If you are then asked to name few animals which are carnivorous, you certainly have no problem of naming a set of animals who eat meat. You may have given names of some of these carnivorous animals as examples for animals for the first question as well. This means your brain knows how to categorise the same set of objects depending on the context (depending on the attributes that each object possesses). It is the same phenomenon that the *meet* and *join* concepts model in FCA.

These similarities between the formalization of concepts in FCA and ideas/concepts of human understanding/thought process mentioned above suggest that the way the humans formulate concepts, structure them and use them in the process of understanding and expressing ideas is analogous to the way concepts are formulated in FCM and are structured in a Concept Lattice.

5. The Model

The proposed model has 3 major components:

1. Preprocessor
2. Matcher
3. User feedback processor

Structure of the model is illustrated in the following diagram.

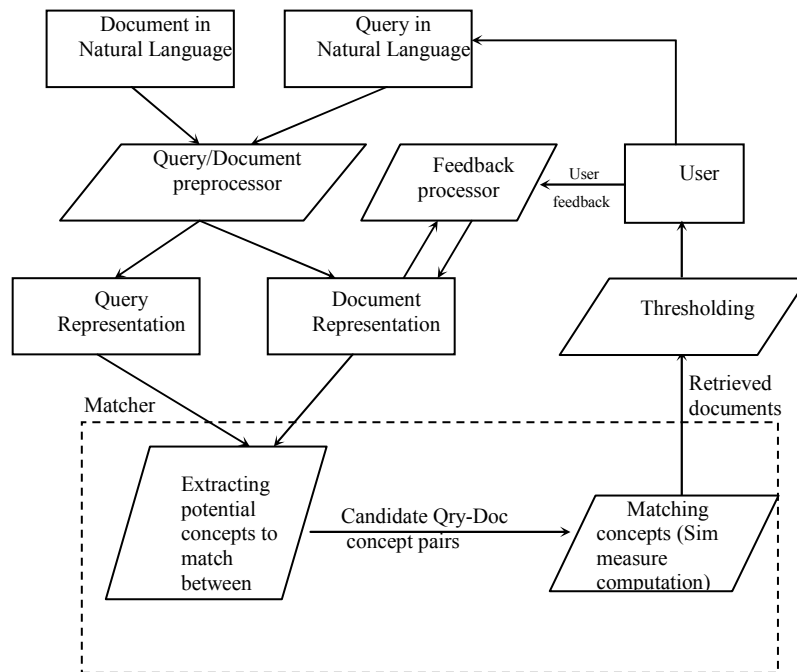


Fig. 3. Block diagram of the model

5.1 The Preprocessor

The most important component of all is the preprocessor. It is responsible for extracting concepts (objects, attributes & their relationships) from natural language text material (queries or documents), constructing a training set of concepts/patterns for the BAM, transforming the initial set of training concepts into a consistent set of formal concepts that can be represented in a concept lattice structure, and encoding the underlying concept lattice structure in a BAM.

Extracting concepts

Extracting objects and their attributes from natural language text is partly a natural language understanding problem that has not yet been very successful, especially for unrestricted text. Even though the large amount of unrestricted text makes extracting desired features more difficult for IR, the fact that a deep and complete understanding of the text may not be necessary for IR makes a shallow and partial representation of the content of text sufficient [11]. We use a set of conventional natural language processing techniques such as part-of-speech tagging (POS) and syntactic chunking (of sentences into noun and verb phrases). The results of tagging and chunking together with a selected set of prepositions (in English language) such as *of*, *in*, *at* etc that appears as connecting words between noun groups are used to extract a shallow, but sufficiently representative set of objects and (their) attributes from documents (and also from queries) to form concepts to represent the text.

Noun-Phrase analysis has been used by a number of researchers for extracting features for document indexing for IR and has proved to be superior to full text indexing [11]. The way we use noun phrases, however is different to the way they are used in phrase-based document indexing. Each noun and verb group detected is assigned an identification label and a string representing the syntactic structure is constructed for each sentence using these labels. Note that text that does not belong to a noun or verb group is left unlabelled and used as it is. For example, the syntactic structure of the sentence “[*The dog*] (*chased*) [*the cat*] *away*” would look like: $NG_1|VB_1|NG_2|away$, where NG_1 is the label for the first noun group (in this case [*The dog*]), NG_2 is the label for the second noun group and VB_1 is the label for the first verb group. | is the separator character.

The syntactic structures of the sentences thus constructed and their individual components are then further analysed for the identification of objects and attributes. We use a few rules that were developed after analysing the empirical results of POS tagging and chunking of textual material for this purpose.

Prepositions (connectors) between two chunks

Use of Noun phrases for document indexing in general do not capture the relationships between noun groups. We capture the relationships between those noun groups that are connected by the prepositional connectors *in*, *on*, *of*, *with*, *to*, *into* and *from*. The relationships between noun groups inferred by these connectors can be interpreted as *object-attribute* relationships. It should be noted here that unlike in

conceptual graphs, concept lattices do not keep relationship types (e.g. is_a, part-of etc). It only needs to know whether there is a object-attribute relationship between two terms/phrases. The set of connectors/prepositions mentioned were selected after analysing experimental results on the syntactic structure of sentences. This list is not complete by any means, but consists of the most frequent and useful connectors.

Syntactic structure of Noun Groups

In most of the cases, noun groups contain several noun words and adjectives/modifiers within them. These words within a noun group have important relationships between them and therefore will be useful if extracted to form meaningful concepts. The POS tags attached to each word is helpful to analyse the syntactic structure within each noun group in order to detect useful object-attribute relationships between the words within the group. We make use of a number of rules developed after analysing experimental results on the syntactic structures of noun groups for this purpose (that are not detailed here).

For example, if the syntactic structure of a noun group is DT|JJ|NN or JJ|NN (e.g. “*The_DT fat_JJ man_NN*”) then a concept is formed as $NN \rightarrow JJ$ (i.e. *man \rightarrow fat*). Here “*fat*” is an attribute of the object “*man*”. Here the tag *_DT* stands for a determinants, *_JJ* for an adjective and *_NN* for a noun.

Some noun groups were found to contain only noun words (no adjectives). (E.g. [*compute_NN keyboard_NN*]). In such cases it is difficult to determine which word(s) to be taken as the attribute and which word(s) as object(s). For such noun groups, concepts are formed taking each noun word in the group as an object as well as an attribute. For the example [*computer_NN keyboard_NN*], two concepts are formed one as *computer \rightarrow keyboard* and the other as *keyboard \rightarrow computer*. In the first case *computer* is regarded as the main subject in which case *keyboard* is an attribute of the computer. In the second case, *keyboard* is considered as the main topic in which case *computer* is an attribute of the keyboard. This specifies that the context of the topic "keyboards" is computers (not typewriter keyboards).

In addition to the above, possessive relationships between two noun terms within noun groups denoted by a trailing 's or 's are also detected and used. These are correctly identified by taggers and tagged by most taggers (e.g. LtChunk tags them with the tag *_POS*), e.g. [*The_DT man_NN's_POS hair_NN*]. These possessive relationships are also captured and processed separately to form concepts containing all the words in the noun group up to “ ‘ “ as the object and the words after the tag *POS* as the attribute. i.e. *Man \rightarrow hair* for the above example. Note that determinants (like *A, The*) are always ignored.

The above-mentioned methods/rules, though not perfect, have shown to extract a reasonably well representative set of concepts from a given text. Additionally, they happen to deal with certain verbal groups as well, e.g. both “*Man's hair*” and the “*Hair of the man*” gives the same concept *Man \rightarrow hair* (can be interpreted as “*man has hair*”)

The set of objects and their attributes thus extracted makes up the initial training set for training a BAM. A consistent set of storable training patterns is then constructed

from the extracted set of concepts (by applying the completeness constraint $A' = B$ and $B' = A$) and a BAM is trained (weights are set according to the equation (3) given in section 3.2). BAMs formed as described above for each document/query represents that document/query in our model in its subsequent stages of the IR process.

Importance of concepts and keywords

A given document generally consists of more than one simple concept (of the kind we extract from text). Some of these may be more generic concepts that might not be very important to distinctly identify the document (by a query) while some others may be much more important. This importance of a concept in a document is modelled by means of a weight. More important concepts are expected to gain bigger weights compared to the less important ones as a result of the online learning process describe below. In the proposed model, we use weights for:

1. each related object-attribute pair present in each document representation (we call such a pair a *unit concept*) and
2. each keyword/keyphrase in each document

The weights of unit concepts model the importance of their object-attribute pairs with respect to the individual documents, within which the concepts are present, while the weights of the keywords/keyphrases model the importance of keywords/keyphrases in the documents. Note that we distinguish between unit concepts and keywords as we are more interested in concept matching rather than keyword matching. For us a unit concept (i.e. an object and a related attribute) makes more sense than a single keyword/keyphrase, as it describes a topic and its context.

We also consider it important to maintain the importance of concepts/keywords with respect to each individual document. A concept which is important to identify and retrieve one document may not be important to identify another document, which also contains the same concept. The presence of the said concept in the second document should not miss-recognize the second document as relevant to a query that contains the said unit concept. Most centralised document indexing schemes for IR have this drawback as they assign a single weight for each index term with respect to the whole document collection. Instead maintaining weights separately with respect to each individual document allows the same object-attribute pair (unit concept) or the same keyword/keyphrase to have different importance weights in different documents. It should be noted here that we do not weight concepts or keywords in the query.

The weights of all the unit concepts and keywords of document representations are initialised to a pre-decided value and are then continuously updated during the subsequent query sessions depending on the user feedback. It is these updated document representations that represent the documents at the subsequent query sessions.

5.2 The Matcher

Once queries and documents are represented in concept lattices (encoded in BAMs), the next step is to match a given query with the documents and compute similarity measures (known as the RSV: Retrieval Status Value) for each query-document pair. These RSV values are then subject to threshold in order to decide what documents should be presented to the user. This whole process involves three subtasks.

Obtaining candidate concepts to match between (from the document and the query representations)

Not all the concepts in a given pair of a query and a document lattices match each other. Therefore attempting to compare each and every concept in the query lattice with each and every concept in the document lattice is not worth the effort. In addition such a matching strategy fails to make use of the important knowledge of the order relationship structure, which has already been captured and encoded in the document/query representations (in BAMs). This is where the importance of using BAMs matters. Instead of using conventional traversing and searching, we make use of the BAM's properties to get most specific (and most generic) concept for a given set of objects (and attributes) respectively. Our strategy is based on the *object concepts* and *attribute concepts* defined by each object and attribute in the query. An *object concept* (*attribute concept*) is the concept returned by a BAM when a pattern containing only one object (attribute) is presented to its object (attribute) layer. The goal here is to extract the most specific concepts (wherever possible) to match between, as we are interested in matching the most specific ideas/concepts between queries and documents whenever such information is available.

For instance, the word *dog* may appear in many documents in different contexts. A given user may be interested in (say) *dog races*. In this case, we need to look for the more specific context (*dog races*) of the object *dog*.

For obtaining candidate concepts to match between a query and a document, we first look for the presence of query objects and attributes in the document representation. For each object and attribute common to the document and the Query, Object and attribute concepts (respectively) are extracted from both the query and the document BAMs. Such object and attribute concept pairs are the candidate concept pairs to match between the query and the document. During this process, we make sure to extract the most specific concepts wherever possible and also not to extract the same concept pair more than once. Also we avoid extracting document (query) concepts that are general (in the general-specific hierarchy in the concept lattice) to any of the already extracted document (query) concepts to match with the same query (document) concept.

Notice that, in case of an object or attribute in the query appearing as both object and attribute in the document representation, we check whether there is any order relation (in the concept hierarchy) between them in order to avoid matching two related document concepts with the same query concept. In case of such related document concepts, only the most specific concept is considered for matching. In some cases we find the same term (word or phrase) appears both as an object and as an attribute

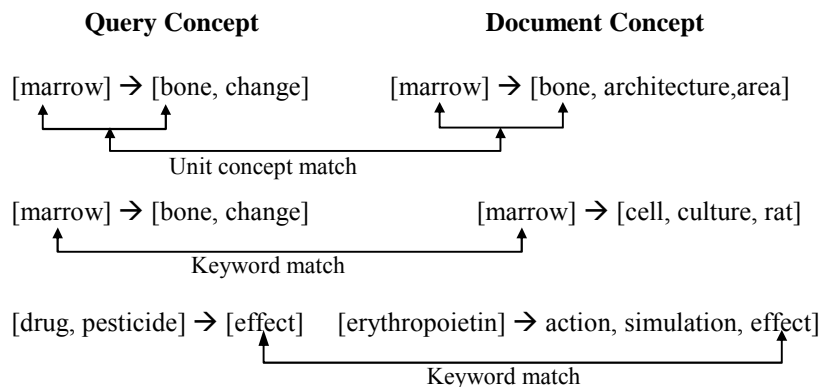
in document representations, but they represent two different ideas/concepts (i.e. they are not related in the concept hierarchy). In this case, the attribute concept given by the query BAM is also taken into account as a candidate concept to be matched with the object concept given by the query BAM, in addition to the object concept given by the document BAM.

The following are the eight different cases identified and taken into account in the development of the algorithm that extracts candidate concepts to match between queries and documents in our implementation.

1. Query object is present ONLY in the document object set
2. Query object is present ONLY in the document attribute set
3. Query object is present in both (document object & attribute) sets and they are related (i.e. the query object plays both object and attribute roles in the document and they are present in the same concept or in different concepts of which one is a super/sub concept of the other.)
4. Query object is present in both (document object & attribute) sets and they are not related
5. Query attribute is present ONLY in the document object set
6. Query attribute is present ONLY in the document attribute set
7. Query attribute is present in both (document object & attribute) sets and they are related
8. Query attribute is present in both (document object & attribute) sets and they are not related

Computing a similarity measure (RSV value)

Each candidate query and document concept pair extracted to match between is then examined for the presence of common unit concepts and keyword/keyphrases. Presence of matching unit concepts between a candidate query and a document concept pair leads to a concept match. The more unit concept matches is the better, as it means the two concepts are more similar. The presence of a common object or an attribute that do not participate in a matching unit concept leads to a keyword matching (see the following illustration).



A similarity measure for each candidate query-document concept pair considered is computed as the sum of the weights of matching unit concepts and keywords. The final RSV value for a query-document pair is the sum of the similarity measures of all the candidate query-document concept pairs considered between the query and the document.

Thresholding

The RSV values computed for each query-document pair are then subject to thresholding in order to decide which documents are to be presented to the user as the retrieved set of documents. We use a kind of dynamic thresholding strategy by taking into account the total number of unit concepts available in all the candidate query concepts considered for comparing with document concepts (i.e. the total number of unit concepts we are looking for in the candidate document concepts) to compensate the varied sizes (size of a query is determined by the number of unit concepts in its representation) of queries. This value is multiplied by a predefined base threshold value. Use of a base threshold value allows us to experiment on the best thresholding value to be used by varying the base threshold.

$$(\text{Base Threshold}) \times \left(\frac{\text{No. of unit concepts in all the candidate query concepts considered for matching between a given query-document pair}}{\text{No. of unit concepts in all the candidate query concepts considered for matching between a given query-document pair}} \right)$$

5.3 Feedback Processor

The task of feedback processor is to accept the user feedback in the form of *yes* or *no* (i.e. accepts a document as relevant or reject it as irrelevant) and accordingly modify the document representation.

Unlike most other models, we use the user feedback to modify the document representation rather than the query representation. The traditional approach, which uses user feedback for query reformulation, though it has shown about 20% improvements on recall and precision, does not lead however to a learning system, because the important user decisions are used only within one query session for searching for one information need. The result gained by relevance feedback at one query session is usually not available for the subsequent query sessions, as they are not learned by the system. A separate learning mechanism is required to make the systems adaptive.

In our model, the document representations are updated instead, and the modifications made are retained for later use. We expect the document representations to converge to a well representative set of concepts (for each document) over a period of time. Such a set of concepts, in fact, will become more customized to the vocabulary and the writing style of the end user, as it is the concepts of the user formulated queries that are appended into relevant document representations. Our reinforcement learning process works as follows:

If user says a particular (retrieved) document is relevant to a given query, all the unit concepts of the query are amended to the document representation. In case that a

particular unit concept of the query is already present in the document, we consider it as an important unit concept (because it has made some contribution for the document's retrieval in the first place) and therefore its weight is increased by a pre-decided amount. Unit query concepts not present in the document are simply added to the document representation with an initial weight value. This may result in unnecessary unit concepts getting into the document's representation, but we expect such concepts to end up with low weights in the long run. It is important to note that we strictly maintain the roles of the terms during this document updating process. Also, we apply the completeness constraint after adding new unit concepts to a document representation in order to combine the new unit concept(s) with appropriate concepts and place them at the appropriate positions in the concept hierarchy.

On the other hand, if a user says a particular (retrieved) document is not relevant to the query then we examine the unit concepts and keywords that are common to the query-document pair (i.e. those matching unit concepts and keywords that contributed for the document's retrieval) and their weights are decreased (by a pre-decided value) to say that those unit concepts and keywords, though common to both the query and the document, are not very important to decide the relevancy of the document to the query.

6. Related Work

Concept lattices have been used for information retrieval by a number of researchers including Godin et al., [12,13,5]; Carpineto and Romano, [3,4]; Cole and Eklund [7]; Cole et al [8]. Most of these researchers have employed FCA to support user interface design, to help the user navigate through the concept lattice to locate desired documents. Almost all of them formulate a concept as a set of documents as the objects and their keywords as the attributes, and represent the entire document collection in a single large concept lattice. In contrast, we attempt to capture concepts as an analogue of how the human cognitive brain process might work. The most related work to that which we describe here is the Lattice-based data structure proposed by Merwe and Kourie [18]. Their data structure is very much similar to our representation scheme in the sense that we both define "a concept" in the same way and embed lattice structures in two layered data structures. However, Merwe's paper lacks a retrieval process to compare with ours.

7. Discussion

We expect that in the long run, when used on a given collection of documents with a good representative set of natural language queries, the system should stabilise on a better set of representative concepts for each document in the collection, through which a better recall and precision can be achieved. We expect our system to perform better than the traditional models in terms of effectiveness, given representative query and document concept lattices.

Several advantages and disadvantages of the proposed model have been identified in comparison to the well known IR models.

Advantages

1. Performs explicit concept matching
2. Adaptive - the system continuously learns as it is used making it suitable for more customised/personalised type of IR applications. However, it is possible to provide an option to remove the knowledge learnt by the system over its lifetime, leaving it with only the initial knowledge/original concepts extracted initially from the documents. This would reset the system back to its starting point.
3. Improves recall through use. Documents that were not picked up for a particular query at an early attempt will be picked up later (for the same or similar query) as a result of upgrading the representations of those documents by other queries. This is very useful and one of the most important features of our model. A collection of documents and queries that has more than one query relevant to each document is required for exploiting the potential of this feature.
4. Improves precision through use. Because of the degrading of weights of matching unit concepts and keywords between a query and documents that are retrieved but judged as not relevant, the RSV values of such documents for the same or similar query at later trials will be smaller, and as such the likelihood of retrieving them at later trials for similar queries is less.
5. The number of terms used/stored may be less (Inverted index indexes all the terms)
6. Nothing is central (each document is represented independently) and so is suitable for an agent-based retrieval on documents distributed over a computer network.
7. Used only information local to a given document and therefore can be applied on collections, which are not restricted to a particular domain.

Disadvantages

1. The document representation process may be computationally expensive especially because the system needs to build BAMs for each document at each query session.
2. Extracting concepts from queries and documents to match between may also be expensive especially when the documents are lengthy (and so have more concepts in the representation). This process has two stages (1) to check the presence of each object/attribute of the query representation in the document representation and (2) presenting the object/attribute concepts made up of the matching object/attribute into the corresponding layers of the query and Document BAMs to extract the concepts to match between.
3. Only local information is used. Therefore the matching process does not have any idea of the global document collection (the problem space).
4. At present, no query reformulation mechanism is used. Short queries are adversely affected.
5. At present the implemented model does not support pure keyword-queries, i.e. queries formulated with few isolated keywords.

6. The concept extraction process is not perfect in the sense that some important concepts will not be extracted, especially because of the distant special locations the participating components appear in the sentences. Also, some useless concepts get into the representations, leading to the retrieval of irrelevant documents. A better object and attribute extraction for the document/query representation will improve the performance of the model.

At present, we are at the stage of conducting preliminary testing for improving our model. Due to the long training process, we have been working on a small document collection with 236 documents and 177 queries. This is a subset of the Cranfield collection. Since our model works best if each document is relevant to many queries (this allows the document representations to learn), we have created the above mentioned document collection by choosing documents/queries from the Cranfield collection which are judged as relevant to three or more queries. Preliminary results are promising with 5.1 average precision over all 177 queries. Following are the precisions at various points.

Retrieval points	Avg. Precision
5 documents	0.37
10 documents	0.270
20 documents	0.168

It should be noted that, due to less number of relevant documents for queries in the document collection used, number of (actual) recall points for queries are less. This might have affected the value of the average precision to have a high value. Also the less number of relevant documents for queries makes the precision values shown in the table (above) smaller down the retrieval points. However, the value of precision at 5 seems an interesting figure that shows how good the performance (precision) is within the top 5 documents retrieved.

We have no information (at present) to comment on the storage requirements over a long period of time as the document representations tend to grow over the time. But, we expect the system to converge to a fixed document representation. Keeping track of the less-used or never-used unit concepts, and thus perceived unimportant concepts in document representations and removing such less useful concepts from the documents will help making the model efficient in terms of retrieval speed, and also reduce the storage requirements of the system.

An important future enhancement to the model for better performance will be to incorporate a query enhancement/reformulation mechanism. At present, the model does not have a query reformulation component to enhance the initial user queries with additional related concepts/keywords.

References

1. Amari, S. (1972). Learning patterns and pattern sequences by self-organizing nets of thresholding elements. IEEE Trans. On Computers, 21(11), 461-482.

2. Belohlávek, R (2000): Representation of Concept Lattices by Bidirectional Associative Memories, *Neural Computation* Vol 12 N.10 October 2000. Pp 2279-2290
3. Carpineto, C., & Romano, G. (1996): A Lattice Conceptual Clustering System and its Application to Browsing Retrieval. *Machine Learning*, 24, 1-28.
4. Carpineto, C., & Romano, G. (1996) : Information retrieval through hybrid navigation of lattice representations. *International Journal of Human-Computer Studies*, 45, 553-578.
5. Carpineto, C. & Romano, G. (2000): Order-theoretical ranking, *JASIS* vol51 No. 7 587-601 (2000).
6. Cole, R and Eklund, P.W. (1993): Scalability of Formal Concept Analysis, *Computational Intelligence*, Vol 2, No. 5, 1993 <http://www.int.gu.edu.au/kvo/papers/>
7. Cole, R.J. and Eklund, P.K. (1996): Application of Formal Concept Analysis to Information Retrieval using a Hierarchically Structured Thesaurus, <http://www.int.gu.edu.au/kvo/papers/> International Conference on Conceptual Graphs, ICCS '96, Sydney, 1996, pp. 1-12, University of New South Wales, 1996.
8. Cole, R.J. Eklund, P.K. Stumme, G.: CEM-A Program for Visualization and Discovery in Email, <http://www.int.gu.edu.au/kvo/papers/> In D.A. Zighed, J. Komorowski, J. Zytkow (Eds), *Proc. of PKDD 2000*, LNAI 1910, pp. 367-374, Springer-Verlag, Berlin, 2000.
9. Darmstadt University of Technology: Formal Concept Analysis S/W, http://www.mathematik.tu-darmstadt.de/ags/ag1/Software/software_en.html
10. Evans, D.A. and Zhai, C. (1996): Noun-Phrase Analysis in Unrestricted Text for Information Retrieval, *Proceedings, 34th Annual Meeting of the Association for Computational Linguistics (ACL'96)*, pages 17--24, Santa Cruz, CA.
11. Ganter, B. Wille, R. (1999) : Formal Concept Analysis : Mathematical Foundations, ISBN 3-540-627771-5 Springer-Verlag Berlin Heidelberg 1999
12. Godin, R., Gecsei, J., & Pichet, C. (1989): Design of a browsing interface for information retrieval. *Proceedings of the 12th International Conference on Research and Development in Information Retrieval (ACM SIGIR '89)*, pp.32-39. Cambridge, MA, ACM.
13. Godin, R., Missaoui, R., April, A. (1993): Experimental comparison of navigation in a Galois lattice with conventional information retrieval methods. *International Journal of Man-machine Studies*, 38, 747-767.
14. Hopfield, J.J.(1984) : Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Natl. Acad. Sci. U.S.A.*, 81, 3088-3092.
15. Kosko, B. (1987) : Adaptive bidirectional associative memory. *Applied Optics*, 26(23), 4947-4960.
16. Kosko, B. (1988) : Bidirectional associative memory. *IEEE Trans. Systems, Man and Cybernetics*, 18(1), 49-60.
17. Lewis, D. (1992): Feature Selection and Feature Extraction for Text Categorization, *Proceedings of Speech and Natural Language Workshop*, 1992
18. van der Merwe, F.J. & Kourie, D.G. (2001): A Lattice-Based Data Structure for Information Retrieval and Machine Learning, *ICCS'01 International workshop on Concept Lattices-based KDD*.
19. Wille. R. (1997): Conceptual Graphs and Formal Concept Analysis. In: Lukose, D. et. al. (eds.): *Conceptual Structures: Fulfilling Peirce's Dream*, *Proceedings of the ICCS'97*. Springer, Berlin--New York (1997) 290--303 16

This paper was presented at the 24th BCS-IRSG European Colloquium on IR Research (ECIR'02) held in March 25-27, 2002 in Glasgow, UK, and appeared in the "Lecture Notes in Computer Science" series of Springer titled "Advances in Information Retrieval – 24th BCS-IRSG European Colloquium on IR Research, Glasgow, UK, March 2002", Eds. Crestani, F., Girolami, M. and Rijsbergen C. J., ISSN 0302-9743