Encoding Classifications into Lightweight Ontologies

Fausto Giunchiglia, Maurizio Marchese, and Ilya Zaihrayeu

Department of Information and Communication Technology
University of Trento, Italy
{fausto, marchese, ilya}@dit.unitn.it

Abstract. Classifications have been used for centuries with the goal of cataloguing and searching large sets of objects. In the early days it was mainly books; lately it has also become Web pages, pictures and any kind of electronic information items. Classifications describe their contents using natural language labels, which has proved very effective in manual classification. However natural language labels show their limitations when one tries to automate the process, as they make it very hard to reason about classifications and their contents. In this paper we introduce the novel notion of *Formal Classification*, as a graph structure where labels are written in a propositional concept language. Formal Classifications turn out to be some form of lightweight ontologies. This, in turn, allows us to reason about them, to associate to each node a normal form formula which univocally describes its contents, and to reduce document classification to reasoning about subsumption.

1 Introduction

In today's information society, as the amount of information grows larger, it becomes essential to develop efficient ways to summarize and navigate information from large, multivariate data sets. The field of classification supports these tasks, as it investigates how sets of "objects" can be summarized into a small number of classes, and it also provides methods to assist the search of such "objects" [8]. In the past centuries, classification has been the domain of librarians and archivists. Lately a lot of interest has focused also on the management of the information present in the web: see for instance the WWW Virtual Library project [1], or directories of search engines like Google, or Yahoo!.

Web directories are often called *lightweight ontologies* [23]. However, they lack at least one important property attributable to the notion of ontology. Namely, that an ontology must be represented in a *formal language*, which can then be used for *automating reasoning* [16]. None of the existing human crafted classifications possesses this property. Because classification hierarchies are written in natural language, it is very hard to automate the classification task, and, as a consequence, standard classification approaches amount to *manually* classifying objects into classes. Examples include DMOZ, a human edited web directory, which "*powers the core directory services for the most popular portals and search*

Y. Sure and J. Domingue (Eds.): ESWC 2006, LNCS 4011, pp. 80–94, 2006.

engines on the Web, including AOL Search, Netscape Search, Google, Lycos, DirectHit, and HotBot, and hundreds of others" [22]; and the Dewey Decimal Classification System (DDC) [5]. Although they are based on well-founded classification methodologies, all these classifications have a number of limitations:

- the semantics of a given category is implicitly codified in a natural language label, which may be ambiguous and will therefore be interpreted differently by different classifiers;
- a link, connecting two nodes, may also be ambiguous in the sense that it may be considered to specify the meaning of the child node, of the parent node, or of both. For instance, a link connecting the parent node *"programming"* with its child node *"Java"* may, or may not mean that (a) the parent node means "computer programming" (and not, for example, "events scheduling"); (b) that the child node means "Java, the programming language" (and not "Java, the island"); and (c) that the parent node's meaning excludes the meaning of the child node, i.e., it is "programming and *not* Java";
- as a consequence of the previous two items, the classification task also becomes ambiguous in the sense that different classifiers may classify the same objects differently, based on their *subjective* opinion.

In the present paper we propose an approach to converting classifications into lightweight ontologies, thus eliminating the three ambiguities discussed above. This in turn allows us to automate, through propositional reasoning, the essential task of document classification. Concretely, we propose a three step approach:

- first, we convert a classification into a new structure, which we call *Formal Classification* (*FC*), where all the labels are expressed in a propositional Description Logic (DL) language (i.e., a DL language without roles) [3];
- second, we convert a FC into a *Normalized Formal Classification* (*NFC*). In NFCs each node's label is a propositional DL formula, which univocally codifies the meaning of the node in the corresponding classification, taking into account both the label of the node and its position within the classification;
- third, we encode document classification in NFCs as a propositional satisfiability (SAT) problem, and solve it using a sound and complete SAT engine.

NFCs are *full-fledged* lightweight ontologies, and have many nice properties. Among them:

- nodes' labels univocally codify the set of documents, which can be classified in these nodes;
- NFCs are *taxonomies* in the sense that, from the root down to the leaves, labels of child nodes are subsumed by the labels of their parent nodes;
- as nodes' labels codify the position of the nodes in the hierarchy, document classification can be done simply by analyzing the set of labels. There is no need to inspect the edge structure of the NFC.

The remainder of the paper is organized as follows. In Section 2 we introduce classifications and discuss how they are used. In Section 3 we motivate a formal

approach to dealing with classifications. In Section 4 we introduce the notion of FC as a way to disambiguate labels in classifications. In Section 5 we discuss how we disambiguate links in classifications by introducing the notion of NFC. In Section 6 we show how the essential task of document classification can be fully automated in NFCs by means of propositional reasoning. In Section 7 we discuss related work. Section 8 summarizes the results and concludes the paper.

2 Classifications

Classifications are rooted trees, where each node is assigned a *natural language* label. Classifications are used for the categorization of various kinds of objects, such as books, office documents, web pages, and multimedia files into a set of *categories*. Classifications are also used for searching for objects by browsing the categories and looking inside those, where the objects are likely to be located.

We define the notion of classification as *a rooted tree* $C = \langle N, E, L \rangle$ *where* N *is a finite set of nodes,* E *is a set of edges on* N*, and* L *is is a finite set of labels expressed in natural language, such that for any node* $n_i \in N$*, there is one and only one label* $l_i \in L$*.*

Labels describe real world entities or individual objects, and the meaning of a label in a classification is the set of documents, that are *about* the entities (or individual objects) described by the label. We call this meaning of labels, the *classification semantics of labels*. Note, that a label can be about a document, e.g., a book; and, in this case, the classification semantics of this label is the set of documents, which are about the book, e.g., book reviews.

There are many methodologies for how to classify objects into classification hierarchies. These methodologies range from the many rigorous rules of DDC [5], "polished" by librarians during more than one hundred years; to less strict, but still powerful rules of classification in a web directory[1]. In all the different cases, a human classifier needs to follow a common pattern, which we summarize in four steps. We discuss the steps below, and we elucidate them on the example of a part of the DMOZ web directory shown in Figure 1.

1. Disambiguating labels. The challenge here is to disambiguate natural language words and labels. For example, the classifier has to understand that in the label of node n_7, the word "*Java*" has at least three senses, which are: an island in Indonesia; a coffee beverage; and an object-oriented programming language;
2. Disambiguating links. At this step the classifier has to interpret links between nodes. Namely, the classifier needs to consider the fact that each non-root node is "viewed" in the *context* of its parent node; and then specify the meanings of the nodes' labels. For instance, the meaning of the label of node n_8, *computers*, is bounded by the meaning of node n_6, *business books publishing*;
3. Understanding classification alternatives. Given an object, the classifier has to understand what classification alternatives for this object are. For instance, the book *"Java Enterprise in a Nutshell, Second Edition"* might potentially be put

[1] See, for instance, the DMOZ classification rules at http://dmoz.org/guidelines/.

in all the nodes of the hierarchy shown in Figure 1. The reason for this is that the book is related to both business and technology branches;

4. Making choices. Given the set of classification alternatives, the classifier has to decide, based on a predefined system of rules, where to put the given object. The system of rules may differ from classification to classification, but one rule is commonly followed: the *get-specific* rule. The rule states that any object must be classified in a category (or in several categories), which most *specifically* describes the object. In order to follow this rule, one needs to "dig" deep into the classification schema and find a category, which is located as low as possible in the classification tree, and which is still more general than what the object is about. Note, that there may be more than one such category. For instance, if the get-specific rule was used, then one would classify the above mentioned book into nodes n_7 and n_8, as they most specifically characterize it.

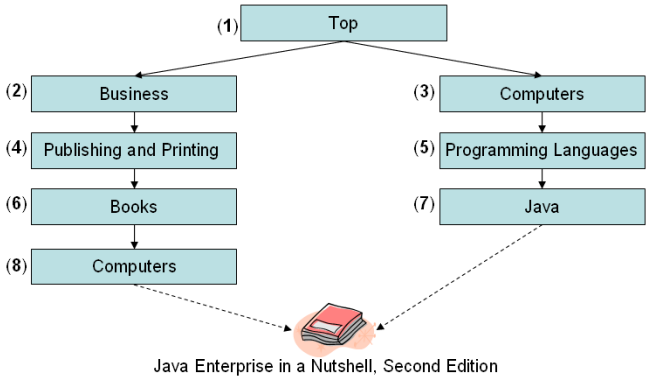

Fig. 1. A part of the DMOZ web directory

Note, that the four steps above are also followed when one is searching for an object by means of classification browsing. The only difference is in that now the categories are searched for where to find the object, and not where to put it.

3 Why Formal Classifications?

Let us exemplify our arguments in favour of a formal approach to classification on the English part of the DMOZ web directory[2]. We report a summary of the statistical analysis we performed on it in Table 1.

Humans have proven to be very effective at performing steps 1 and 2 described in Section 2. However, there are still some challenges to be addressed. The main challenge in step 1 is dealing with the ambiguities introduced by multiple possibilities in meaning. One source of this is in that labels contain many

[2] We excluded branches leading to non-English labels, such as "Top/World/" or "Top/Kids_and_Teens/International/".

Table 1. DMOZ statistics

Statistics category	Value
Total English labels	477,786
Tokens per label, avg.	1.81
Total links classified in English labels	3,047,643
Duplicate links, % from the total	10.70%
Nouns and adjectives polysemy, avg.	3.72
"and"'s and *"or"*'s per label, avg.	0.23
Total disjunctions per label, avg.	3.79
Root-to-leaf path length, avg.	7.09
Branching factor, avg.	4.00

conjunctions "and"'s and "or"'s, whereas they actually mean inclusive disjunction, i.e., either the first conjunct, or the second, or both. For instance, the phrase "wine and cheese" means either wine, or cheese, or both. Apart from the conjunctions, multiple possibilities are also introduced by punctuation marks denoting enumeration (e.g., the comma), and by words' senses (recall the "Java" example from the previous section). It has been shown, that cognitive reasoning with the presence of multiple possibilities in meaning is an error-prone task for humans [10]. For instance, even if DMOZ labels are short phrases, consisting, on average, of 1.81 tokens, they contain 0.23 conjunctions per label; and average polysemy for nouns and adjectives is 3.72 per word.

The challenge of step 2 is that the classifier may need to follow a long path of nodes in order to figure out a node's meaning. It has two consequences: first, the classifier needs to deal with the growing complexity in ambiguity introduced by each new label in the path; and, second, the classifier has to consider each new label in the context of the labels of the ancestor nodes, and, thus, partly resolve the ambiguity. Note, that the average length of a path from the root to a leaf node in DMOZ is rather high and it constitutes 7.09 nodes.

Steps 3 and 4 are where the real problems for humans begin. Even with classifications of an average size, it is not easy to find all the classification alternatives. With large classifications this task becomes practically unfeasible. For instance, think about possible classification alternatives in DMOZ, which has 477,786 English categories. Thus, at step 3, a human classifier may not be able to enumerate all the possible classification alternatives for an object.

Step 4 requires abundant expertise and profound methodological skills on the side of the classifier. However, even an expert makes subjective decisions, what leads, when a classification is populated by several classifiers, to nonuniform, duplicate, and error-prone classification. If the get-specific rule is used, then the classifier has to parse the classification tree in a top-down fashion, considering at each parent node, which of its child nodes are appropriate for the classification. Note, that even if DMOZ encourages the classification of a Web page in a single category, among 3,047,643 links (classified in English labels), about 10.70% are classified in more than one node[3]. And, about 91.36% of these are classified in

[3] We identified duplicate links by exact equivalence of their URLs.

two different nodes. This is not surprising given that DMOZ is populated by more than 70,000 classifiers, and that it has average branching factor of 4.00.

Given all the above described complexity, humans still outperform machines in natural language understanding tasks [20], which are the core of steps 1 and 2. Still, the availability of electronic repositories that encode world knowledge (e.g., [12, 14]), and powerful natural language processing tools (e.g., [17, 12]) allows the machines to perform these steps reasonably well. Moreover, machines can be much more efficient and effective at steps 3 and 4, if the problem is encoded in a formal language, which is what we propose to do in our approach.

4 Disambiguating Labels

Formal Classifications (FCs) are rooted trees, where each node is assigned a *formal language* label. FCs and classifications are related in the sense that a FC is a *formalized* copy of a classification. In other words, a FC has the same structure as the classification, but it encodes the classification's labels in a formal language, capable of encapsulating, at the best possible level of approximation, their classification semantics. In this respect, classifications's labels have at least one nice property. Namely, since labels are meant to describe real world entities, and *not* actions, performed on or by entities, and relations between entities, the labels are mainly constituted of noun phrases; and, therefore, there are very few words which are verbs. This makes it very suitable to use a Description Logic (DL) language as the formal language, as DLs are a precise notation for representing noun phrases [3].

We define the notion of Formal Classification as *a rooted tree* $FC = \langle N, E, L^F \rangle$ *where* N *is a finite set of nodes,* E *is a set of edges on* N*, and* L^F *is a finite set of labels expressed in Propositional Description Logic language* L^C*, such that for any node* $n_i \in N$*, there is one and only one label* $l_i^F \in L^F$.

Converting classifications into FCs automates step 1, as described in Section 2. In our approach we build on the work of Magnini et. al. [13]. We translate a natural language label into an expression in L^C by means of mapping different parts of speech (POSs), their mutual syntactic relation, and punctuation to the classification semantics of labels. We proceed in two steps, as discussed below:

1. Build atomic concepts. Senses of (multi-word) common nouns and adjectives become atomic concepts of L^C, whose interpretation is the set of documents about the entities, which are denoted by the nouns, or which possess the qualities denoted by the adjectives[4]. We enumerate word senses using WordNet [14], and we write [`x#i`] to denote an atomic concept corresponding to the i^{th} sense of the word `x` in WordNet. For instance, [`programming#2`] is an atomic concept, whose interpretation is the set of documents which are about computer programming;

[4] Because of their negligibly small presence, we do not consider verbs. We neither consider articles, numerals, pronouns and adverbs. However, their share in the labels of actual classifications is reasonably small. When such words are found, they are just omitted from the label.

and the atomic concept [red#1] denotes the set of documents which are about red entities, e.g., red cats or red hats. Proper nouns become atomic concepts of L^C, whose interpretation is the set of documents about the individual objects, denoted by these nouns. They may be long expressions, denoting names of people, movies, music bands, and so on. Some examples are the movie *"Gone with the Wind"*, and the music band *"The Rolling Stones"*.

2. Build complex concepts. Complex concepts are built from atomic concepts as follows: first, we build formulas for words as the logical disjunction ($\sqcup$) of atomic concepts corresponding to the words' senses, and we write [x$*$] to denote the disjunction of the senses of word x. For instance, the noun *"Programming"* becomes the concept [programming#1 $\sqcup$ programming#2], whose interpretation is the set of documents which are about event scheduling and/or about computer programming. Second, labels are chunked, i.e., divided into sequences of syntactically correlated parts of words. We then translate syntactic relations of words within chunks to the logical connectives of L^C following a precise pattern. Let us consider few examples.

A set of adjectives followed by a noun group is translated into logical conjunction ($\sqcap$) of the formulas corresponding to the adjectives and the nouns. The interpretation of the resulting concept is the set of documents which are about the real world entities denoted by all the nouns, and which possess qualities, denoted by all the adjectives. For instance, the phrase *"long cold winter blizzard"* is translated into the concept [long$* \sqcap$ cold$* \sqcap$ winter$* \sqcap$ blizzard$*$]. Prepositions are also translated into the conjunction. The intuition is that prepositions denote some commonality between the objects they relate; and, in terms of the classification semantics, this "commonality" can be approximated to the set of documents which are about the both objects. For instance, the following phrases: *"books of magic"*, *"science in society"*, and *"software for engineering"*, they all denote what the two words, connected by the prepositions, have in common.

Coordinating conjunctions "and" and "or" are translated into the logical disjunction. For instance, *"flights or trains"* and *"animals and plants"* become [flight$* \sqcup$ train$*$] and [animal$* \sqcup$ plant$*$] respectively. Punctuation marks such as the period (.), the coma (,) and the semicolon (;) are also translated into logical disjunction. For instance, the phrase *"metro, bus, and trolley"* is converted into the concept [metro$* \sqcup$ bus$* \sqcup$ trolley$*$].

Words and phrases denoting exclusions, such as "excluding", "except", "but not", are translated into the logical negation ($\neg$). For instance, the label *"runners excluding sprinters"* becomes the concept [runner$* \sqcap \neg$sprinter$*$]. However, since they are meant to describe what "there is" in the world, and not what "there isn't", labels contain very few such phrases, if at all.

The use of logical connectives, as described above but with the exception of prepositions, allows it to *explicitly* encode the classification semantics of labels. In other words, the interpretation of the resulting formulas explicitly represents the set of documents which are about the natural language labels. The translation of prepositions is an approximation, as they may encode meaning, which only partly can be captured by means of logical conjunction. For example, *"life in*

war" and *"life after war"* will collapse into the same logical formula, whereas the classification semantics of the two labels is different. In this respect we are different from [18], where DL roles are used to encode the meaning of labels. The advantage of our approach is in that, while using a simpler subset of DLs, we are able to explicitly capture the semantics of a large portion of the label data in a real classification.

In order to estimate how much of the information encoded into the labels of a real classification can be captured using our approach, we have conducted a grammatical analysis of the DMOZ classification. For doing this, we have used the OpenNLP Tools tokenization and POS-tagging library [17], which reports to achieve more than 96% accuracy on unseen data. In Table 2 we show POS statistics of tokens. Note, that about 77.59% of the tokens (nouns and adjectives) become concepts, and about 14.69% (conjunctions and prepositions) become logical connectives of L^C. WordNet coverage for common nouns and adjectives is quite high, and constitutes 93.12% and 95.01% respectively. Detailed analysis of conjunctions and prepositions shows that about 85.26% of them are conjunctions "and", and about 0.10% are conjunctions "or". In our analysis we found no words or phrases which would result into the logical negation. Only about 4.56% of the tokens are verbs and adverbs in all their forms.

Table 2. DMOZ token statistics

POS	**Share**
Common nouns	71.22%
Proper nouns	0.18%
Adjectives	6.19%
Conjunctions and prepositions	14.69%
Verbs, adverbs	4.56%
Other POSs	3.16%

Note, that the propositional nature of L^C allows us to *explicitly* encode about 90.13% of label data in DMOZ (i.e., nouns, adjectives, conjunctions "and" and "or"). Still, this is a rough understated estimation, as we did not take into account multi-word nouns. In fact, manual analysis of the longest labels, as well as of the ones with verbs, shows that the majority of these labels represents proper names of movies, games, institutions, music bands, etc.

5 Disambiguating Edges

As discussed in Section 2, the classification semantics of links codifies the fact that child nodes' labels are always considered in the context of their parent nodes. This means that the meaning of a non-root node is the set of documents, which are about its label, and which are *also* about its parent node. We encode the classification semantics of links as a property of nodes in FCs, which we call the *concept at a node*. We write C_i to refer to the concept at node n_i, and we define this notion as:

$$C_i = \begin{cases} l_i^F & \text{if } n_i \text{ is the root of } FC \\ l_i^F \sqcap C_j & \text{if } n_i \text{ is not the root of } FC, \text{ where } n_j \text{ is the parent of } n_i \end{cases} \quad (1)$$

There may be two meaningful relations between the concept of a parent node, and the label of its child node, as represented in Figure 2:

- in case (a) the label of the child node is about the parent node, but it is also about something else. In this case the parent node *specializes* the meaning of the child node by bounding the interpretation of the child node's label with the interpretation of the concept of the parent node. For instance, think about a classification where the root node is labeled "Italy" and its sole child node is labeled "Pictures" (see Figure 2-a). A human can understand that the meaning of the child node is "pictures of Italy" and not "pictures of Germany", for example. In the corresponding FC this knowledge is encoded into the concept at node $C_2 = [\texttt{italy} * \sqcap \texttt{picture}*]$;
- in case (b) the child node represents a *specification* of the parent node, and their relation can be, for instance, an "is-a" or a "part-of" relation. Note, that in this case, differently from case (a), the parent node does not influence the meaning of the child node. Suppose that in the previous example the child node's label is "Liguria" (see Figure 2-b). A human can understand that the meaning of this node is the same as of its label. In the corresponding FC this knowledge is encoded into the concept at node $C_2 = [\texttt{italy} * \sqcap \texttt{liguria}*]$, which can be simplified to $C_2 = [\texttt{liguria\#1}]$, taking into account that both words "Italy" and "Liguria" have only one sense in WordNet, and given that the corresponding axiom $[\texttt{liguria\#1} \sqsubseteq \texttt{italy\#1}]$ is memorized in some background knowledge base.

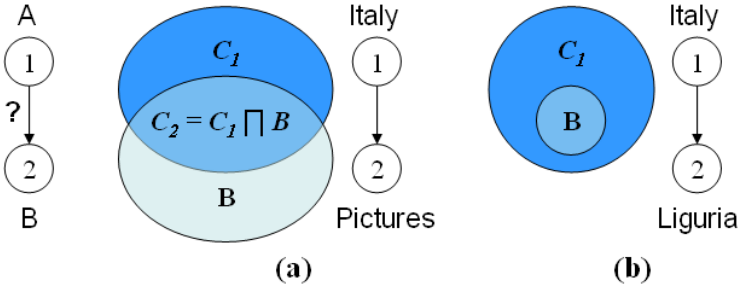

Fig. 2. Edge semantics in FCs

Note, that applying Equation 1 recursively, we can compute the concept at any non-root node n_i as the conjunction of the labels of all the nodes on the path from the root of the FC, n_1, to n_i. This corresponds to how the notion of concept at a node is defined in [7], namely:

$$C_i = l_1^F \sqcap l_2^F \sqcap \ldots \sqcap l_i^F \quad (2)$$

The concept at a node encodes, but only to a certain extent, the path from the root to the node. In fact, there may be more than one way to reconstruct a path

from a concept. Atomic concepts in a concept at a node may be "distributed" differently among different number of nodes, which, in turn, may have a different order in the path. The number of nodes may range from one, when the concept at the node is equivalent to the node's label, to the number of clauses in the DNF equivalent of the concept. However, all the possible paths converge to the same semantically equivalent concept. Consider, for instance, node n_8 in the classification shown in Figure 1. All the following paths will converge to the same concept for this node[5]: *"top:publishing_and_printing:business_books:computers"*, *"top:business:publishing_and_printing:computer_books"*.

We use the notion of concept at a node to define a further new structure which we call *Normalized Formal Classification* (NFC). A NFC is *a rooted tree* $NFC = \langle N, E, L^N \rangle$ *where* N *is a finite set of nodes,* E *is a set of edges on* N*, and* L^N *is is a finite set of labels expressed in* L^C*, such that for any node* $n_i \in N$*, there is one and only one label* $l_i^N \in L^N$ *and* $l_i^N \equiv C_i$.

Note, that the main characteristic of NFCs, that distinguishes them from FCs, is the fact that labels of child nodes are always more specific than the labels of their parent nodes. Particularly, if a taxonomic classification, i.e., a classification with only "is-a" and "part-of" links, is converted into a FC, then the latter is also a NFC. Apart from this, NFCs have a number of important properties relevant to classifications, discussed below:

- the classification semantics of the labels of nodes is the set of documents which *can* be classified in these nodes. We underline the "can" since, as we discuss in the next section, documents which *are* actually classified in the nodes are often a subset of the classification semantics of the labels;
- two nodes, representing in a classification the same real world entities, will have semantically equivalent labels in the NFC. This fact can be exploited for automatic location and/or prevention of adding of such "duplicate" nodes. As an example, consider the different paths that lead to the same concept at a node as described earlier in this section;
- NFCs are full-fledged lightweight ontologies, suitable for the automation of the core classification tasks, such as document classification, as it is discussed in the following section.

6 Document Classification

Before some document d can be classified, it has to be assigned an expression in L^C, which we call the *document concept*, written C^d. The assignment of concepts to documents is done in two steps: first, a set of keywords is retrieved from the document using text mining techniques (see, for example, [19]); the keywords are then converted into a concept using similar techniques to those used in the translation of natural language labels into labels in FCs (see Section 4).

We say that node n_i is a *classification alternative* for the classification of some document d with concept C^d, if $C^d \sqsubseteq l_i^N$. In fact, if this relation holds, then the

[5] For sake of presentation we give these examples in natural language.

document is about the classification node, whose semantics is encoded in the label of the NFC. For any given document d and a NFC, we compute the set of classification alternatives for d in the NFC as follows:

$$A(C^d) = \{n_i | C^d \sqsubseteq l_i^N\} \tag{3}$$

By computing Equation 3, we can automate step 3 described in Section 2. The automation of step 4, i.e., making classification choices, depends on what classification algorithm is used. Below we show how it can be automated for some set A of classification alternatives if the get-specific rule (see Section 2) is used:

$$C(A) = \{n_i \in A | \nexists n_j \in A \ (i \neq j), \ \textit{such that} \ l_j^N \sqsubseteq l_i^N\} \tag{4}$$

The set $C(A)$ includes all the nodes in the NFC, whose labels are more general than the document concept, and more specific among all such labels. As labels of child nodes in NFCs are more specific than the labels of their parent nodes, $C(A)$ always consists of nodes which lie as low in the CNF tree as possible, and which are still classification alternatives for the document. Note, that the get-specific rule applies not only to nodes located on the same path from the root, but also to nodes located in different branches. For instance, a document about computer graphics will *not* be classified in the node *"top:computers"* if the more specific node *"top:arts:computers"* exists.

Formula 4 stipulates that the set of documents classified in some node n_i may (and, in most cases will) be a subset of the interpretation of its label l_i^N. In fact, the set of documents which are *actually* classified in n_i excludes those, which belong to the interpretation of labels, which are more specific than l_i^N. We encode this set in the concept D_i which univocally identifies the set of documents classified in node n_i, and, therefore, defines the classification semantics of n_i in the NFC. We compute D_i as follows:

$$D_i = l_i^N \sqcap \neg \bigsqcup (l_j^N | j \neq i, l_j^N \sqsubseteq l_i^N) \tag{5}$$

Computation of Equations 3, 4 and 5 requires verifying whether subsumption holds between two formulas in L^C. As discussed in [6], a problem, expressed in a propositional DL language, can be translated into an equivalent propositional satisfiability problem, and can therefore be solved using sound and complete reasoning procedures of a SAT decider. The translation rules from L^C to a propositional language transform atomic concepts into propositions, less generality into implication (i.e., $[A \sqsubseteq B] \Rightarrow [A \rightarrow B]$), disjunctions and conjunctions into logical or's and and's respectively (e.g., $[A \sqcup B] \Rightarrow [A \vee B]$), and so on. Interested readers are referred to [6] for details. Thus, if we need to check whether a certain relation rel holds between two concepts A and B, given some knowledge base $\mathcal{KB}$, which represents our a priori knowledge, we construct a propositional formula according to the pattern shown in Equation 6, and check it for validity:

$$\mathcal{KB} \rightarrow rel(A, B) \tag{6}$$

The intuition is that $\mathcal{KB}$ encodes what we know about concepts A and B, and $rel(A, B)$ holds only if it follows from what we know. In our approach $\mathcal{KB}$ is built as a set of axioms which encode the relations that hold between *atomic* concepts in A and B. As discussed in Section 4, atomic concepts in L^C are mapped to the corresponding natural language words' senses. These senses may be lexically related through the synonymy, antonymy, hypernymy (i.e., the "kind-of" relation, e.g., *car* is a kind of *vehicle*), or holonymy (i.e., the "part-of" relation, e.g., *room* is a part of *building*) relations. These relations can be translated into axioms, which *explicitly* capture the classification semantics of the relation that holds between the two senses. Thus, for instance, the set of documents which are about *cars* is a subset of the set of documents which are about a hypernym of the word "car", *vehicle*. The idea, therefore, is to find the lexical relations using WordNet, and to translate synonymy into logical equivalence, antonymy into disjointness, hypernymy and holonymy into subsumption in L^C.

Let us consider an example. Recall the classification in Figure 1, and suppose that we need to classify the following book: *"Java Enterprise in a Nutshell, Second Edition"*, whose concept is $[\texttt{java\#3} \sqcap \texttt{enterprise\#2} \sqcap \texttt{book\#1}]$. It can be shown, by means of propositional reasoning, that the set of classification alternatives includes all the nodes of the corresponding NFC. For sake of space we provide concrete formulas only for nodes n_7 and n_8, whose labels are $l_7^N = [\texttt{computer}* \sqcap \texttt{programming}* \sqcap \texttt{language}* \sqcap \texttt{java}*]$, and $l_8^N = [\texttt{business}* \sqcap [\texttt{publishing}* \sqcup \texttt{printing}*] \sqcap \texttt{publishing}* \sqcap \texttt{books}* \sqcap \texttt{computer}*]$. We can extract the following knowledge from WordNet: the programming language Java is a kind of programming languages, and it is a more specific concept than computer is; books are related to publishing; and enterprise is a more specific concept than business is. We encode this knowledge in the following axioms:

$$a_1 = [\texttt{java\#3} \sqsubseteq \texttt{pr\_language\#1}]; \quad a_3 = [\texttt{book\#1} \sqsubseteq \texttt{publishing\#1}];$$
$$a_2 = [\texttt{java\#3} \sqsubseteq \texttt{computer\#1}]; \quad a_4 = [\texttt{enterprise\#1} \sqsubseteq \texttt{business\#2}].$$

We then translate the axioms and the labels into the propositional logic language, and we verify if the condition in Formula 3 holds for the two labels by constructing two formulas, following the pattern of Equation 6, as shown below:

$$(a_2 \wedge a_3 \wedge a_4) \rightarrow (C^d \rightarrow l_8^N); \qquad (a_1 \wedge a_2) \rightarrow (C^d \rightarrow l_7^N).$$

Then, we run a SAT solver on the above formulas, which shows that they are tautologies. It means that both nodes n_7 and n_8 are classification alternatives for the classification of the book. Among all the classification alternatives, only these two nodes conform to the get-specific rule, and, therefore, are final classification choices for the classification of the book. The latter can be shown by computing Equation 4 by means of propositional reasoning.

Note, that the edges of the NFC are *not* considered in document classification. In fact, the edges of the NFC become redundant, as their information is implicitly encoded in the labels. As from Section 5, there may be several paths to the same concept. Analogously, given a set of labels, there may be several ways to reconstruct the set of edges of a NFC. However, from the classification point of view, all these NFCs are equivalent, as they classify documents *identically*.

7 Related Work

In our work we adopt the notion of concept at a node as first introduced in [6] and further elaborated in [7]. Moreover, the notion of label of a node in a FC, semantically corresponds to the notion of concept of a label introduced in [7]. In [7] these notions play a key role in the identification of semantic mappings between nodes of two schemas. In this paper, these are the key notions needed to define NFCs.

This work as well as the work in [6, 7] mentioned above is crucially related and depends on the work described in [4, 13]. In particular, in [4], the authors introduce the idea that in classifications, natural language labels can be translated in logical formulas, while, in [13], the authors provide a detailed account of how to perform this translation process. The work in [6, 7] improves on the work in [4, 13] by understanding the crucial role that concepts at nodes have in matching heterogeneous classifications and how this leads to a completely new way to do matching. This paper, for the first time, recognizes the crucial role that the ideas introduced in [4, 6, 7, 13] have in the construction of a new theory of classification, and in introducing the key notion of FC.

A lot of work in information theory, and more precisely on formal concept analysis (see, for instance, [24]) has concentrated on the study of concept hierarchies. NFCs are what in formal concept analysis are called concept hierarchies with no attributes. The work in this paper can be considered as a first step towards providing a computational theory of how to transform the "usual" natural language classifications into concept hierarchies.

The classification algorithm, proposed in this paper, is similar to what in the DL community is called *realization*. Essentially, realization is the task of finding the most specific concept(s) an individual object is an instance of given a hierarchy of concepts [3]. The fundamental difference between the two approaches is in that in DL the concept hierarchy is *not* predefined by the user, but is built bottom-up from atomic concepts by computing the partial ordering of the subsumption relation. In our case, the underlying classification structure is defined solely by the user.

In Information Retrieval, the term *classification* is seen as the *process* of arranging a set of objects (e.g., documents) into *categories* or *classes*. There exist a number of different approaches which try to build classifications *bottom-up*, by analyzing the contents of documents. These approaches can be grouped in two main categories: *supervised classification*, and *unsupervised classification*. In the former case, a small set of training examples needs to be pre-populated into the categories in order to allow the system to automatically classify a larger set of objects (see, for example, [2, 15]). The latter approach uses various machine learning techniques to classify objects, for instance, data clustering [9]. There exist some approaches that apply (mostly) supervised classification techniques to the problem of documents classification into hierarchies [11, 21]. The classifications built following our approach are better and more natural than those built following these approaches. In fact, they are constructed *top-down*, as chosen by the user and not constructed bottom-up, as they come out of the document

analysis. Our approach has the potential, in principle, to allow for the automatic classification of (say) the Yahoo! documents into the Yahoo! web directory.

8 Conclusions

In this paper we have introduced the notion of Formal Classification, namely of a classification where labels are written in a propositional concept language. Formal Classifications have many advantages over standard classifications all deriving from the fact that formal language formulas can be reasoned about far more easily than natural language sentences. In this paper we have highlighted how this can be done to perform document classification. However much more can be done. Our future work includes testing the feasibility of our approach with very large sets of documents, such as those classified in the DMOZ directory, as well as the development of a sound and complete query answering algorithm.

Acknowledgements

This work is partially supported by the FP6 project Knowledge Web[6]. We also thank all the members of the KnowDive group, and, especially, Mikalai Yatskevich, for the many useful discussions about earlier versions of this paper.

References

1. The WWW Virtual Library project. see http://vlib.org/.
2. G. Adami, P. Avesani, and D. Sona. Clustering documents in a web directory. *In Proceedings of Workshop on Internet Data management (WIDM-03)*, 2003.
3. Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook : Theory, Implementation and Applications*. Cambridge University Press, 2003.
4. P. Bouquet, L. Serafini, and S. Zanobini. Semantic coordination: a new approach and an application. *In Proc. of the 2nd International Semantic Web Conference (ISWO'03). Sanibel Islands, Florida, USA*, October 2003.
5. Lois Mai Chan and J.S. Mitchell. *Dewey Decimal Classification: A Practical Guide*. Forest P.,U.S., December 1996.
6. F. Giunchiglia and P. Shvaiko. Semantic matching. *workshop on Ontologies and Distributed Systems, IJCAI*, 2003.
7. F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-match: An algorithm and an implementation of semantic matching. *In Proceedings of ESWS'04*, 2004.
8. A.D. Gordon. *Classification*. Monographs on Statistics and Applied Probability. Chapman-Hall/CRC, Second edition, 1999.
9. A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.
10. Johnson-Laird. *Mental Models*. Harvard University Press, 1983.

[6] The Knowledge Web project. See http://knowledgeweb.semanticweb.org/.

11. Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 170–178, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.
12. Douglas B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.
13. Bernardo Magnini, Luciano Serafini, and Manuela Speranza. Making explicit the semantics hidden in schema models. *In: Proceedings of the Workshop on Human Language Technology for the Semantic Web and Web Services, held at ISWC-2003, Sanibel Island, Florida*, October 2003.
14. George Miller. *WordNet: An electronic Lexical Database*. MIT Press, 1998.
15. Kamal Nigam, Andrew K. McCallum, Sebastian Thrun, and Tom M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.
16. Natalya F. Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.*, 33(4):65–70, 2004.
17. The OpenNLP project. See http://opennlp.sourceforge.net/.
18. S. Sceffer, L. Serafini, and S. Zanobini. Semantic coordination of hierarchical classifications with attributes. Technical Report 706, University of Trento, Italy, December 2004.
19. Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
20. J. F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.
21. Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. In *ICDM*, pages 521–528, 2001.
22. DMOZ: the Open Directory Project. See http://dmoz.org/.
23. Michael Uschold and Michael Gruninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33(4):58–64, 2004.
24. Rudolf Wille. Concept lattices and conceptual knowledge systems. *Computers and Mathematics with Applications*, 23:493–515, 1992.