Abstract
Indexing by means of a hierarchical classification is a common method of organising information for subsequent ease of access. The semantic web makes use of this approach by allowing entities to be grouped into classes, with a sub-class / super-class structure. Frequently, different data sources use different hierarchical classification structures, which complicates the task of integrating information from multiple sources. In this paper, we outline a method for automatically determining a soft mapping between the classes in different hierarchies, using a previously described method for identifying equivalence instances. In turn, the identification of the correct category can improve the instance matching performance. A trial application mapping between genres in large movie databases is presented.

Keywords: Soft semantic web, intelligent information management.

1 Introduction
According to recent estimates [1], the amount of new information stored on paper, film, magnetic, and optical media increased by approximately 30% per annum in the period 1999-2002. At that time, the “surface” web was estimated to contain about 170 terabytes of data. Since then, it appears that the volume of information has continued to increase at a similar rate. The problem of information location is fundamental to the successful use of the world’s information resources, and the semantic web is intended to address this issue by enabling the use of meta-data to classify and describe information, services, etc. The ability to create hierarchical classification structures is a core part of the semantic web knowledge representation (and indeed, a core part of many approaches to indexing and organising information). However, there is no universal hierarchy of knowledge – and hence different information sources will almost inevitably differ in the hierarchical classification scheme they adopt. To take a very simple example, the set of tracks or albums classified in one online music store as

```
music > rock > classic rock > 70’s classics
```

may correspond to another’s

```
music > rock&pop oldies
```

This is the basis of the “ontology alignment” or “schema matching” problem, when different sources classify the same set of objects according to two different hierarchies.

Music and film genres are an obvious application for soft category mapping - Aucouturier and Pachet [2] state that

```
music genre is an ill-defined notion, that is not founded on any intrinsic property of the music, but rather depends on cultural extrinsic habits.
```

They also comment that the intensional and extensional definitions of music genres...
frequently do not coincide, and they review various approaches to the representation and assignment of musical genre. They divide approaches into manual and automatic, with the latter further split into approaches which use various features of the music, typically extracted from signal processing, to decide class membership, and approaches which are based on measuring similarity to other tracks in a class.

Downie[3]investigates automatic association of genre tags with digital music, on the basis of sound samples, and [4] looked at the automatic creation of document genre in text libraries, also on the basis of measurable properties of the document. We take the view that genre (and in our view, most other useful hierarchical classifications) are subjective; at the same time, they are a very useful tool for organising and retrieving instances in a collection. Although it is probably impossible to rigorously derive genre on the basis of simple features, the work reported here shows that a useful approximate classification method can be created.

In [5, 6] we have used “instance-matching” to determine that objects from different sources are the same - for example, to deduce with a reasonable degree of certainty that an author known in one database as “Lewis Carroll” represents the same individual as the author known in a second database as “C L Dodgson”.

In this paper we briefly outline the SOFT method for instance matching and show its use in identifying when different news reports are concerned with the same underlying story. We go on to show how this leads to a method of finding correspondences between hierarchies, when instances are classified according to different categorisations (described by different meta-data schemata). We use the equivalence of instances from different sources to learn a soft mapping between categories in these hierarchies, allowing us to compare the hierarchical classification of instances as well as their attributes.

Such correspondences may in turn be used to improve the identification of equivalent instances. Some successful initial tests of the method on large movie databases are reported.

## 2 Instance Matching

We assume two sets of objects (also referred to as instances) \( A = \{a_1 \ldots a_n\} \) and \( B = \{b_1 \ldots b_m\} \), where we wish to establish an approximate relation

\[
h : A \rightarrow B
\]

The SOFT method [5] determines which instances are equivalent by comparing their attributes. For example, if sets \( A \) and \( B \) refer to films, attributes could be \textit{title, director, year} etc.

Let the objects in \( A \) and \( B \) have attribute values taken from \( C_1, C_2, \ldots D_1, D_2, \ldots \) with relations defined as

\[
R_i : A \rightarrow C_i \quad i=1\ldots n_A
\]

\[
S_j : B \rightarrow D_j \quad j=1\ldots n_B
\]

We do not assume that the information about \( A \) and \( B \) in relations \( R_i, S_j \) is identical or completely consistent, but we do assume that some of these relations reflect similar or identical properties of the objects in \( A \) and \( B \). Thus for some choices of pairs of codomains \((C, D)\) we assume an exact or approximate matching function \( h_{ij} \) which for each element of \( C_i \) returns a (possibly fuzzy) subset of \( D_j \). As shown in [5, 6], this can be converted to a mass assignment giving an estimate of the probability that the element corresponding to some \( C_i \) lies in a subset \( \{d_1, \ldots d_l\} \subseteq D_j \).

If \( R_i(a_k) = C_{ik} \)

and \( h_{ij}(C_{ik}) = \tilde{D}_{jk} \) where \( \tilde{D}_{jk} \) denotes a fuzzy subset of \( D_j \)

and \( S_j(\tilde{B}_k) = \tilde{D}_{jk} \) using the (inverse) extension principle

then \( h(a_k) = \tilde{B}_k \)

i.e. \( a_k \) corresponds to a fuzzy subset \( \tilde{B}_k \). We consider each \( h_{ij} \) to give a different “observation” of the true value\(^1\), and seek the fuzzy set most likely to give these observations.

---

\(^1\) In an analogous way, to determine the bias of an unfair coin we could observe the results of several sequences of coin tosses and choose a bias to maximise the likelihood of the observations.
Let $M_n$ be the mass assignment on $B$ that makes the observed values most likely after $n$ observations, i.e. choose the masses to maximize

$$Pr(M_n | o_1, o_2, \ldots, o_n)$$

This gives a way of updating $M$ after each observation. Using a naive bayes assumption

$$Pr(M_n | o_1, o_2, \ldots, o_n) = \frac{Pr(o_1, o_2, \ldots, o_n | M_n) \times Pr(M_n)}{Pr(o_1, o_2, \ldots, o_n)}$$

$$Pr(o_1, o_2, \ldots, o_n | M_n) = Pr(o_1 | M_n) \times Pr(o_2 | M_n) \times \ldots \times Pr(o_n | M_n)$$

Assuming each possible mass assignment $M_n$ is equally likely,

$$M_n(B_j) = \frac{N_n(B_j)}{\sum_{X \subseteq B} N_n(X)}$$

where $N_n(X)$ is number of times the subset $X$ has been observed.

2.1 Example – News Stories

The SOFT algorithm was applied to 1,701 stories from the BBC news archive (www.solutionsseven.co.uk/bbc/ from May 19 - 31, 2005 incl) and 552 stories from the Sky news archive (www.sky.com/skynews/archive, same period). A manually produced ground truth established 353 true matches - this figure is low due to different editorial styles and content. An example is shown below, with matching attribute pairs as implied i.e. Headline-Title, etc. The similarity between longer text sections (summary, content etc) was computed as

$$\sum_{a \in doc1b \in doc2} \sum_{a,b} f_a \frac{Sim(a,b) \times f_b}{f}$$

where $f$ is the relative frequency of a word in a document and $Sim$ is the similarity of words measured using the word similarity matrix[7, 8]. The similarity between short text sections is on the basis of the proportion of common words , with fuzzy matching for dates, currency, etc. Using the SOFT algorithm, we find 380 pairs of matching news stories when the similarity threshold between two stories is set to 0.15.

Figs 1 and 2 illustrate the SOFT result, a recall of 100% and an average precision of 92%.

<table>
<thead>
<tr>
<th>BBC</th>
<th>Sky</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Headline</strong></td>
<td>Oldest FA Cup sells for £420,000</td>
</tr>
<tr>
<td><strong>OriginalPublicationDate</strong></td>
<td>2005/05/19 19:51:30</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td>The oldest existing version of the FA Cup becomes the world's most expensive piece of football memorabilia.</td>
</tr>
<tr>
<td><strong>Content</strong></td>
<td>“The oldest existing version of the FA…”</td>
</tr>
</tbody>
</table>

**Fig. 1.** The number of identified matches in the two data sources and the ground truth (per day). There are some false positives where SOFT wrongly identifies stories as the same but no false negatives

**Fig. 2.** The number of identified matches in the two news data sources compared to the total numbers of stories from the two sources

---

2 In common with many uses of naive Bayes, the independence assumption is hard to justify theoretically but appears to be valid in practice.
3 Hierarchy Matching

Given two (or more) hierarchically organised sets of instances, we can use the SOFT method to determine which instances are equivalent by comparing their attributes. Having determined equivalent instances from the two sources, we can look for correspondences between the different classification structures. Repeating the earlier example, online music sources are typically organised hierarchically, but one site’s music > rock > classic rock > 70’s classics section may correspond (or correspond mostly) to another’s music > rock&pop oldies because both contain mostly the same tracks (or albums) where the notion of “same” is determined by SOFT.

The > symbol indicates that all elements in a sub-category belong to the broader “parent” category e.g. anything in classic rock also belongs to rock and hence to the category music. For convenience we define X > X for all X.

In general, we consider two sets of instances A and B with corresponding sets of labels LA and LB each of which has a hierarchical structure i.e. there is a partial order defined on the labels. Note that this does not imply that a hierarchy induces a partition on the instances - it may be that “orthogonal” attributes such as “date” and “type of storytelling” are represented within the same hierarchy.

Each label \( l_i \in LA \) denotes a subset of A i.e. we have a denotation function

\[
\text{den} : L_A \rightarrow A
\]

such that

\[
l_i > l_j \iff \text{den}(l_j) \subseteq \text{den}(l_i)
\]

(and similarly for B)

For example, if A and B are sets of films then LA and LB could be genres such as western, action, thriller, romance, etc.

We use the SOFT method outlined above to derive a soft mapping on the sets of entities A and B

\[
h : A \rightarrow \bar{P}(B)
\]

where \( \bar{P}(B) \) is the set of all fuzzy subsets of B.

It may not be possible to say in all cases that an element of A corresponds to a specific element of B or that it does not correspond to any element of B. This mapping is used to determine a (soft) correspondence between any pair of labels \( l_i \) and \( l_j \) from the label sets LA and LB

\[
g : L_A \rightarrow \bar{P}(L_B)
\]

Given a label \( l_i \in LA \) we consider its denotation \( \text{den}(l_i) \) under the mapping \( h \) and compare it to the denotation of \( l_j \in LB \)

In the ideal case if the two labels are equivalent,

\[
h(\text{den}(l_i)) = \text{den}(l_j)
\]

Given that \( h \) is approximate and that the correspondence between labels may not be exact, we use semantic unification to compare the sets.

\[
\Pr(l_i \rightarrow l_j) = \Pr(h(\text{den}(l_i)) = \text{den}(l_j))
\]

This gives an interval-valued conditional probability which expresses the relation between a pair of labels; we then extract the most likely pair to give a crisp relation

\[
g_c : L_A \rightarrow L_B
\]

Ideally, it should be possible to map such categories into a user’s personal hierarchy – here, we concentrate on extracting rules from the overlap between categories in different classification structures based on a sample and use the derived rules to predict likely categorisations of new examples.

4 Application to Film Databases

The two film websites “rotten tomatoes” and the internet movie data-base (IMDb) are “user-maintained” datasets which aim to catalogue movie information. The databases denoted dbT and dbI below are derived from these sites, respectively containing 94,500 and 94,176 film records, and were used in experiments. Since dbT and dbI are produced by two different movie web sites, there is inevitable “noise” existing in the film data; i.e. different tag sets, different genre names and missing elements.
In order to match attributes, some very simple string matching functions were used as follows:

(i) String S1 is an approximate substring of S2 if S1 is shorter than S2 and most words in S1 are also in S2.

(ii) String S1 is an approximate permutation of S2 if they have a high proportion of common words, i.e. degree of match = proportion of common words, which must be at least two (typical strings are only a few words long).

Both ignore “stop” words such as the, and, etc. We note also that it is possible to obtain better results for people’s names (attributes such as cast, director, etc) using a more structured approach which extracts first name and surname and then matches on that basis.

The average matches between domains are given in Table 1.

### Table 1. Average degree of match between attributes in dbI and dbT

<table>
<thead>
<tr>
<th>dbI attribute</th>
<th>dbT attribute</th>
<th>average match</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>Year</td>
<td>100%</td>
</tr>
<tr>
<td>Title</td>
<td>Title</td>
<td>41%</td>
</tr>
<tr>
<td>Directed_by</td>
<td>Director</td>
<td>27%</td>
</tr>
<tr>
<td>Aka</td>
<td>Title</td>
<td>21%</td>
</tr>
</tbody>
</table>

On the basis of the three best attributes, the system identified movies from dbI dated 1976-1990 which were also in dbT, and compared the genre classification. The similarity threshold between two film records was set to 0.5 giving a total of 14,124 movies which are found to be identical.

The similarity between two genres is relatively hard to decide from text string matching. For example, “animation” is not similar to “children” from the point of view of text matching, but the extension of the sets of films in these two categories shows considerable overlap. Some examples of intuitively reasonable genre mappings are listed in Table 2.

### Table 2. Examples of matching genre pairs determined by the system

<table>
<thead>
<tr>
<th>dbT genre</th>
<th>dbI genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animation</td>
<td>Children</td>
</tr>
<tr>
<td>Comedy</td>
<td>Drama</td>
</tr>
<tr>
<td>Horror</td>
<td>Suspense</td>
</tr>
<tr>
<td>Sci-fi</td>
<td>Fantasy</td>
</tr>
</tbody>
</table>

### 4.1 Results on Unseen Data

The attribute and genre mappings were applied to a new set of 24,839 entries from dbI (calendar years 2000-05), trying to find matches in dbT. For comparison, a manually produced ground truth established 1274 true matches – this figure is low due to the relatively large number of entries for TV series, “foreign” movies etc in dbI which are not included in dbT. Using the SOFT algorithm without genre mapping, we find 861 pairs of matching film entries when the similarity threshold between two films is set to 0.44. With the presence of the ground truth, 261 film matching pairs out of 382 film pairs in 2000 are missing, 102 out of 364 in 2001 are missing, 87 out of 330 in 2002 are missing, 60 out of 142 in 2003 are missing, and 3 out of 8 in 2004 are missing (see figure 3). This represents a recall of 67 % and a precision of 100%. Incorporating the genre mapping as well produces a much better (100%) recall, at the expense of some loss in precision – see figures 3 and 4.
5 Summary

We see the availability of good meta-data as crucial to the development of intelligent information management systems, and the use of soft methods for approximate matching as a vital component in such systems. We have presented methods for determining similarity of content and for finding identifying similar categories within hierarchical classifications.

Initial results are promising. Further testing is necessary to fully establish the robustness of this work, involving a wider variety of datasets. The use of this method with a personal hierarchy would enable the user to adapt source meta-data to his/her own preferences and hence improve the retrieval process.

Fig. 3. The number of correctly identified matches in the two data sources, without and with genre mapping

Acknowledgements

This work was partially supported by BT Intelligent Systems Lab (www.bt.com) and by the FP6ePerSpace project IST 506775 (www.ist-eperspace.org)

We would like to thank the anonymous referees for their helpful comments

Fig. 4. The number of incorrectly identified matches when using genre information (white blocks) and the number of pairs not identified when not using genre information (shaded block)

References