RC 21926 (98585) 5 January 2001

Computer Science

# IBM Research Report

## Distributed Resource Discovery through Exchanges of Examples and Classifiers

**C.-S. Li, L. D. Bergman, V. Castelli, Y.-C. Chang, M. Hill, J. R. Smith**

IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

# Distributed Resource Discovery through Exchanges of Examples and Classifiers

C.-S. Li, L. D. Bergman, V. Castelli, Y.-C. Chang, M. Hill, and J. R. Smith
Data Management Department, IBM T. J. Watson Research Center
P. O. Box 704, Yorktown Heights, NY 10598

## ABSTRACT

Distributed resource discovery is an essential step for information retrieval and providing information services. This step is usually used for determining the location of an information/data repository that has relevant information/data. The most fundamental challenge is the potential lack of semantic interoperability among these repositories. In this paper, we proposed an algorithm to enable distributed resource discovery. In the proposed method, the distributed repositories achieve pair wise semantic interoperability through the exchange of both examples (either in the form of raw data or through a set of descriptors) and the classifiers (which have been trained on the raw data or the descriptors). For each repository, the local classifier is used to classify the examples sent by the remote repository, and the classifier from the remote repository is used to classify the examples from the local repository. The correspondence of the class labels from two repositories can then be established by examining the classification results.

**Keywords:** Taxonomy, resource discovery, interoperability, meta-search.

## 1. INTRODUCTION

Interoperability among database systems, portals, information repositories, online retail stores, and electronic marketplaces is becoming an increasingly important area of research. The goal is to allow a single query to access distributed, often heterogeneous, data sources and search engines. Consequently, mediators in these search engines must be able to integrate heterogeneous information sources with potentially different data representations, taxonomy, ontology, and search capabilities[1,2]. These search engines have to present a unified context for uniform access of information[6]. As a result, they have to translate the query from the original user query in the unified context to the target source for native execution. Examples include those web meta-search engines that can access a variety of other search engines (such as Altavista [www.altavista.com] or Lycos [www.lycos.com]), and return a single set of integrated results. Specifically, a meta-search engine for images, MetaSeek[3], was previously developed at Columbia University to perform meta-search on three image search engines: IBM QBIC, Virage, and SaFe (an image search engine developed at Columbia University).

Resource discovery has been used to describe the process of determining the nature of entities that are contained within an information repository. When a query is generated and sent to multiple heterogeneous repositories, an important step of processing the query is to determine what information is available from each repository. For example, a query that seeks to find areas of deforestation in the Amazon basin between 1995 and the present would need to determine whether a given repository contains appropriate data which might include (in increasing order of specificity):

- determining if the repository is oriented towards earth sciences,
- if it contains deforestation information,
- if the information covers the Amazon basin, or
- if the requested dates are available.

Search interoperability is generally implemented in one of two ways:

- One approach is to define a common set of terms (ontology or taxonomy), and requires repositories to employ the common ontology to remain interoperable. This is feasible in well-established domains such as medicine or particle physics. Examples include Global Change Master Directory (GCMD)[7,8] for earth science data. This directory, GCMD, allows access to a variety of science data distributed at various sites through a single query interface.
- The other approach is to build pair wise translators to create mappings between the terminology and data models used in two repositories. This approach allows the existence of local "dialects" within a repository as long as the mapping between the dialect in the local repository and those used in the query can be established. Recently, the issue of translating Boolean query constraints across heterogeneous information sources[6] such as translate a query from "score>8" to "rating>0.8" or from "format=vhs" to "format = video tape" have been investigated within the Stanford Digital Library Project [Chang & Garcia-Molina[4,5]].

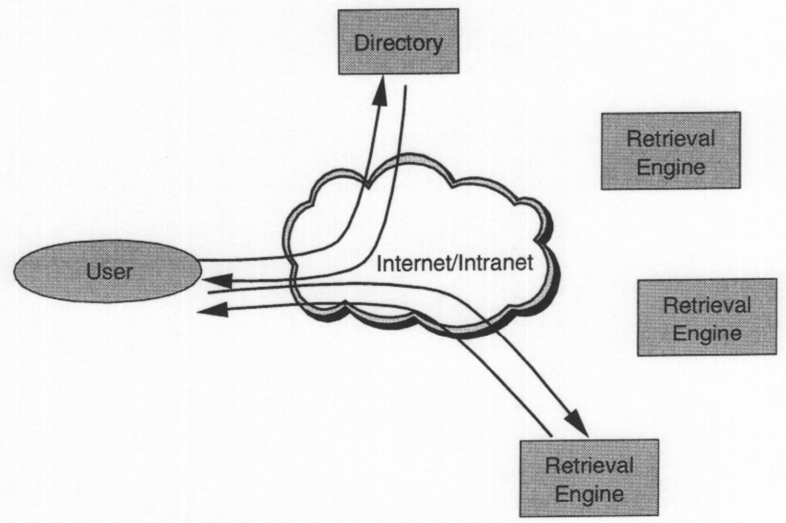


**Figure 1 Distributed Resource Discovery.**

The objective of this paper is to address the challenge in which the translation of the ontology/taxonomy between two repositories or between the local repository and the query is not known in advance. The entities in the repository may have different labels than those used in the query or they may have the same labels, but not the same meaning. For example, a query with the term "deciduous forest" may be used in a repository that has entities labeled "hardwood", yet it may be quite difficult to determine this correspondence. On the other hand, two different repositories may have the term "temperature", but one may be daily maximum temperature, the other may be hourly mean temperature, and thus not correspond. It is important to be able to determine whether entities with different or identical labels actually refer to the same underlying semantic concept.

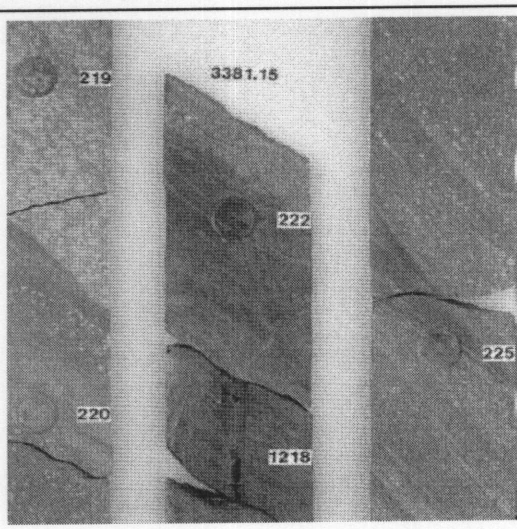


**Figure 2(a) Strata samples from oil well drilling process.**

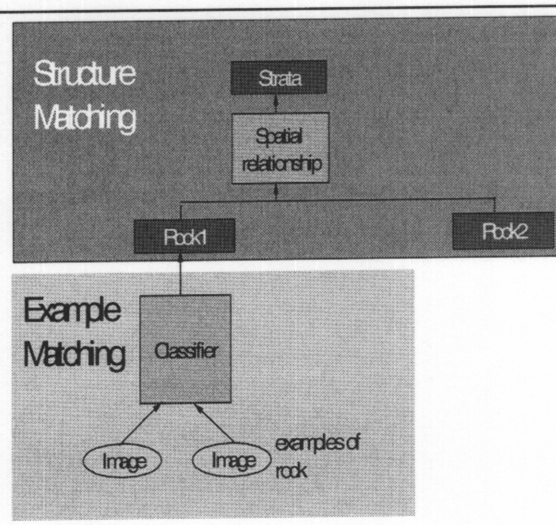


**Figure 2(b) Sample semantic description.**

Specifically, we propose to addresses the mapping discovery between two taxonomies through the construction of a supervised classifier for the taxonomy and a set of examples for each entity in the taxonomy. In many application areas, application data is categorized using classifiers. Examples of categorization include: assigning labels of "fraudulent" and "non-fraudulent" to medical claims records, determining land cover categories such as "forest" or "water" for each region in a satellite image, or assigning a category to a news item for access by a web search engine. As can be seen in the last example, the categorization need not be a simple "flat" scheme - it may be hierarchical, or even overlapping. In the proposed method, the distributed repositories achieve pair wise semantic interoperability through the exchange of both examples and the classifiers simultaneously between these two repositories. For each repository, the local classifier is used to classify the examples sent by the remote repository, and the classifier from the remote repository is used to classify the examples from the local repository. The correspondence of the class labels from two repositories can then be established by examining the classification results.

The rest of this paper is organized as follows: Section 2 gives a preliminary of distributed resource discovery. The proposed method is described in Section 3. An illustrating example is given in Section 4. Additional discussions are presented in Section 5. This paper is concluded in Section 6.

## 2. PRELIMINARY

Figure 1 illustrates a distributed resource discovery scenario. The user requests resources (which can be web pages, images, documents, etc) through Internet/intranet. Usually, the resources are requested in two stages: (1) the available retrieval engines are located from a retrieval engine directory. From this directory, the locations of the retrieval engines are located. This is the retrieval engine discovery process. (2) Once the retrieval engines are located, the data query is sent to those retrieval engines that may have the resources. The retrieved results are then sent back to the user. This process is the data retrieval process.

Resource discovery in a distributed environment is usually difficult if the **vocabulary**, **terminology**, **taxonomy**, **ontology**, or **labels** are not unified and standardized (vocabulary, terminology, taxonomy, ontology, and label are used interchangeable in this paper). Note that it is possible to construct high-level semantic description from low-level semantic description using spatial, temporal and Boolean operators, as shown in Fig. 2. Figure 2(a) shows an example of rock samples extracted from drilling oil/gas wells. Strata from such rock samples usually consist of multiple layers of rocks, with one rock sitting on top of another rock. The spatial relationship between these rocks is “on top of”. Each basic rock type (such as shale, sandstone, or siltstone) needs to be described. Consequently, one process can be used to define the basic rock type, while a separate process can use these basic building blocks in conjunction with spatial operators to describe the high-level construct.

In this paper, we will focus on the method to describe these “elementary” or “basic” semantic entities through a set of examples. In Fig. 2, examples of rocks have been used to define a specific rock. However, the set of examples alone is not adequate. A classifier giving label assignment to examples is usually required in order to allow **unambiguous generalization** from the set of examples and (possibly) counter-examples.

Figure 3(a) illustrates the operation of a typical classifier, in which the classifier is trained by a set of training set. The training set usually contains sets of examples and their corresponding labels. Once the training period ends, the classifier is used on the test set and generates label(s) for each individual example from the test set. A specific example of Fig. 3(a) is shown in Fig. 3(b) for high-resolution satellite images (with 1-meter resolution). In this case, the classifier is trained based on the training set. The vocabulary of the training set is wheat and rye. When the classifier operates on the input data, each of the input data is then assigned a label from the vocabulary defined by the training set, resulting in such results as: (1) oats --> nothing, as none of the labels has more than 1% confidence), (2) winter wheat --> wheat with 90% confidence and to rye with 5% confidence, (3) rye hybrid --> wheat with 20% confidence and rye with 85% confidence.

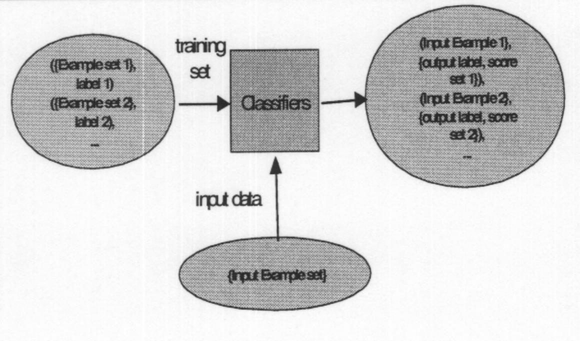


**Figure 3(a) Operation of a typical classifier involving training data and operational data.**

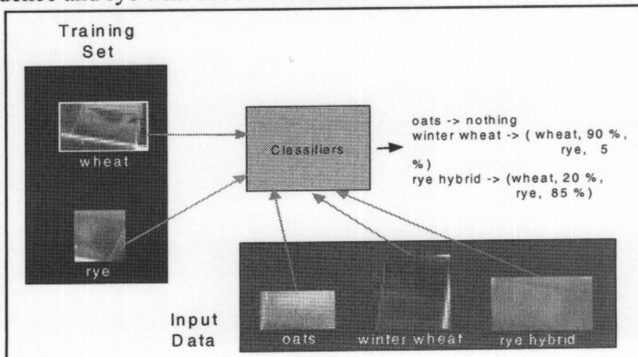


**Figure 3(b) Example of applying classifier to images.**

## 3. PROPOSED ALGORITHM: XCLASSIFY

The drawback of the process described in the previous section is the difficulty to establish the association between two labels. Using the example in Fig. 3 as an example, we can only deduct that "rye hybrid" can be derived from "wheat", but not the other way around. Consequently, we propose a new algorithm, **Xclassify** (Fig. 4), to perform **cross-classification** in order to establish the association between vocabularies. Note that we refer to providing the training data from one set of entities to the classifier of the other as cross-classification.

In the cross-classification process (shown in Fig. 4), repository examples from the remote repository are used as training set for the repository classifier, while the query examples from the local repository are used for the query classifier. The query examples are then served as the input data for the repository classifier, while the repository examples are served as the input data for the query classifier. The output from the classifier will then output to an inference engine - to be described later, and generate association rules.

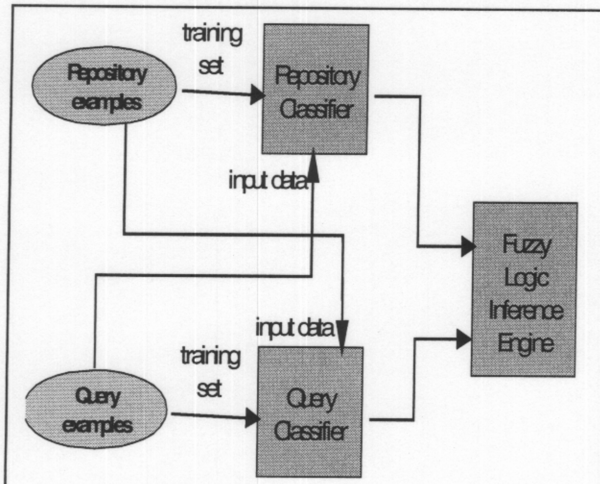


**Figure 4 Xclassify algorithm.**

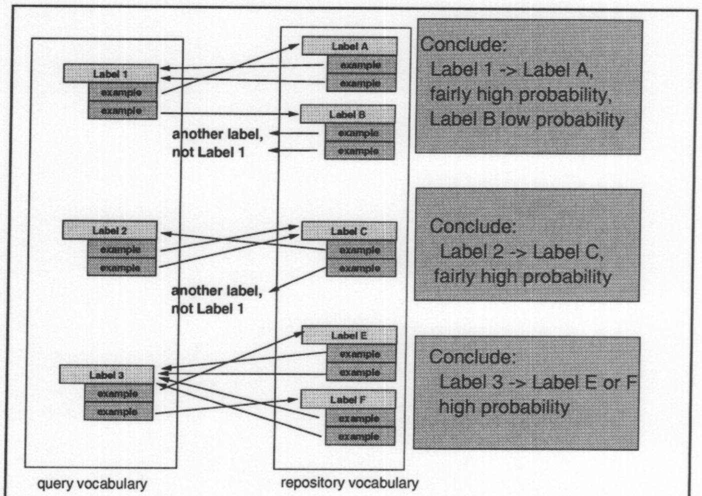


**Figure 5 Sample correspondence generated from the Xclassify algorithm.**

A cross-classification process example is shown in Fig. 5. The labels are divided into query side and repository side. The query side vocabulary consists of three labels: Label 1 has two examples, label 2 has two examples, and label 3 also has two examples. The repository side consists of 5 labels, each of which has two examples. By applying the query classifier on the repository examples, and the repository classifier on the query examples, we can establish the linkage between the query examples and the repository labels, as well as the repository examples and the query labels. For example, both of the repository examples under label A are classified into label 1 using the query classifier. In contrast, the query examples under label 1 are classified into label A and B, respectively, while the examples of label B are classified into other labels in the query vocabulary. Consequently, we can deduct that label 1 from the query vocabulary matches to label A from the repository vocabulary closely with high confidence, but not between label 1 and label B.

The proposed algorithm is intended to extract four elementary types of correspondence between two sets of classes:

- One-to-One: Class "a" from repository 1 uniquely corresponds to class "b" from repository 2. No other classes from repository 1 correspond to class "b", and class "a" does not correspond to any other classes in repository 2.
- One-to-Many: Class "a" from repository 1 corresponds to classes "$b_1,\ldots,b_k$" from repository 2 but no other classes from repository 2. No other classes from repository 1 correspond to classes "$b_1,\ldots,b_k$".
- Many-to-One: Classes "$a_1,\ldots,a_l$" from repository 1 correspond to only class "b" from repository 2. No other classes from repository 1 correspond to class "b".
- Many-to-Many: Classes "$a_1,\ldots,a_l$" from repository 1 correspond to classes "$b_1,\ldots,b_k$" from repository 2. The correspondence between these two class groups is self-contained. In other words, classes "$a_1,\ldots,a_l$" do not correspond to any other classes from repository 2, and no other classes from repository 1 correspond to classes "$b_1,\ldots,b_k$".

Note that the purpose of this algorithm is to discover those irreducible correspondences. An association rule is said to be irreducible if this rule cannot be further broken into multiple one-to-one, one-to-many, many-to-one, or many-to-many rules.

As an example, if there is a rule {a,b} → {1,2}, and also {a} → {2} and {b} → {1}, then the rule {a,b} → {1,2} is reduceable and shall be eliminated. This rule shall exist only if the rule {a} → {2} or {b} → {1} does not exist.

In general, the $m^{th}$ repository in a network of repositories contains $Z_m$ classes. Each of these classes represents a single semantic type or concept such as "forest" or "water". A total of $K_i$ ($1 \leq i \leq Z_m$) examples considered being representative of the $i^{th}$ class is given as the definition for class i. Examples might be sample images, samples text documents, sets of sample parameter values from a dataset, etc. The class definition may also contain a text label used to identify the class.

For two repositories, we assume the following:

- Repository 1 has $\{a_1, \ldots, a_M\}$ defined as classes and examples $\{{}^1e_1, \ldots, {}^{K1}e_1, \ldots, {}^1e_M, \ldots, {}^{KM}e_M\}$ associated with the classes,
- Repository 2 has $\{b_1, \ldots, b_N\}$ defined as classes and examples $\{{}^1f_1, \ldots, {}^{K1}f_1, \ldots, {}^1f_N, \ldots, {}^{KN}f_N\}$ associated with the classes.

The **Xclassify** algorithm for establishing the correspondence between these two repositories is as follows:

1. The examples for each class definition for repository 1 are presented as input to the trained classifier from the repository 2. The $k^{th}$ example from repository 1 produces one or more labels at the output of the classifier for repository 2. The output from the classifier is represented by a vector ${}^k\boldsymbol{\beta}$, where ${}^k\boldsymbol{\beta} = [{}^k\beta_1, \ldots, {}^k\beta_N]^T$ and ${}^k\beta_1 + \ldots + {}^k\beta_N = 1$. Most of the classifiers generate single output, and thus only one of the ${}^k\beta_j$ will be one while the rest of the ${}^k\beta_j$ will be zero. When multiple outputs are generates with varying confidence level, more than one $\beta_j$ can be nonzero. However, the constraint ${}^k\beta_1 + \ldots + {}^k\beta_N = 1$ still holds. After the examples from class $i$ of repository 1 ($i \in \{a_1, \ldots, a_M\}$) are presented to the classifier of repository 2, the correspondence between this class from repository 1 and class $j$ from repository 2 ($j \in \{b_1, \ldots, b_N\}$) is $A_{ij}$. The correspondence is computed from the output vectors generated by each example: $A_{ij} = 1/K_i \sum_{k=1}^{Ki} {}^k\beta_j$
2. The examples for each class definition for repository 2 are presented as input to the trained classifier from the repository 1. The $l^{th}$ example from repository 2 produces one or more labels at the output of the classifier for repository 1. The output from the classifier is represented by a vector ${}^l\boldsymbol{\alpha}$, where ${}^l\boldsymbol{\alpha} = [{}^l\alpha_1, \ldots, {}^l\alpha_M]^T$ and ${}^l\alpha_1 + \ldots + {}^l\alpha_N = 1$. Most of the classifiers generate single output, and thus only one of the ${}^l\alpha_j$ will be one while the rest of the ${}^l\alpha_j$ will be zero. When multiple outputs are generates with varying confidence level, more than one $\alpha_j$ can be nonzero. However, the constraint ${}^l\alpha_1 + \ldots + {}^l\alpha_M = 1$ still holds. After the examples from class $j$ of repository 2 ($j \in \{b_1, \ldots, b_N\}$) are presented to the classifier of repository 1, the correspondence between this class from repository 2 and class $i$ from repository 1 ($i \in \{a_1, \ldots, a_M\}$) is $B_{ji}$. The correspondence is computed from the output vectors generated by each example: $B_{ji} = 1/K_j \sum_{l=1}^{Kj} {}^l\alpha_j$
3. Compute the association score $C_{i,j}$ between $a_i$ and $b_j$ by using the following formulation: $C_{i,j} = A_{i,j} B_{j,i}$, where $i \in \{a_1, \ldots, a_M\}, j \in \{b_1, \ldots, b_N\}$.
4. Generate rules to associate labels between repository 1 and 2 from the association score $D_{i,j}$:
   a. One-to-one (class $i$ in repository 1 corresponds to class $j$ in repostiry 2) : $C_{i,j} > 0$ while $C_{i,m} = 0$ for $m \neq j$ and $C_{n,j} = 0$ for $n \neq i$. In other words, only the intersecting element of the $i^{th}$ row and $j^{th}$ column of the C matrix is nonzero.
   b. One-to-many (class $i$ in repository 1 corresponds to multiple classes in repository 2): For those nonzero elements of the $i^{th}$ row of the C matrix, they are the only nonzero elements in the corresponding column.
   c. Many-to-one (multiple classes in repository 1 correspond to class $j$ in repository 2): For those nonzero elements of the $j^{th}$ column of the C matrix, they are the only nonzero elements in the corresponding row.
   d. Many-to-many (multiple classes as a group in repository 1 correspond to multiple classes as a group in repository 2): Only those elements located at intersecting locations between the rows corresponding to the classes from repository 1 and the columns corresponding to the classes from repository 2 are nonzero.

We need to perform cross-classification in both directions in order to adequately determine whether the set of labels in the query definitions correspond to those in the repository definitions. To see why this is so, consider simply cross-classifying the example sets from repository 2 using the classifier from repository 1. This produces a measure of how well each class from repository 2 corresponds to a given class from repository 1, but produces no measure of specificity for that correspondence. For example, suppose that a set of examples from repository 1 is provided for the rock type "shale". Also suppose that one of the classes produced by the trained classifier from repository 2 is "rock". The "shale" examples when provided as input to the classifier of repository 2 will all be assigned output label "rock" with a high degree of confidence. "Rock" is not a good match for "shale", however, since "rock" can also match "sandstone", "limestone", etc. If we are evaluating different repositories for how well they capture the type of entity we are searching for, we want to assign a much higher score to an entity that really is shale, than to the entity "rock". Thus, simply using a unidirectional cross-classification is inadequate. The way to include a measure of specificity in the scoring procedure is to perform a fully symmetric cross-classification using the examples and the trained classifier from each repository. In the sample problem, the examples for "rock" are likely to contain examples of "sandstone", "limestone", etc., in addition to "shale". When these examples are provided as input to the classifier from repository 1, the output label "shale" will have a fairly low value. By combining the high score for query "shale" → repository "rock" with the low score for repository "rock" → query "shale", we can obtain an intermediate score, which is what we want (note, the symbol "→" is used here to represent cross-classification. It can also be read as "implies")

## 4. ILLUSTRATING EXAMPLE

In this section, we will use an example to illustrate the algorithm proposed in the previous section. We assume the following:

- There are four classes {1, 2, 3, 4} in repository 1,
- Three classes {a, b, c} in repository 2.
- Each class in repository 1 has three examples, and
- Each class in repository 2 has four examples.

The following tables show the outcome when applying the classifier to the examples. The table on the left shows the results when applying the classifier from repository 2 to the examples from the repository 1, while the table on the right shows the results when applying the classifier from repository 1 to the examples from the repository 2.

| Class Label | Example | Outcome |
|---|---|---|
| Class 1 | Example 1 | a |
| Class 1 | Example 2 | a |
| Class 1 | Example 3 | b |
| Class 2 | Example 1 | c |
| Class 2 | Example 2 | c |
| Class 2 | Example 3 | c |
| Class 3 | Example 1 | b |
| Class 3 | Example 2 | b |
| Class 3 | Example 3 | a |
| Class 4 | Example 1 | a |
| Class 4 | Example 2 | a |
| Class 4 | Example 3 | a |

| Class Label | Example | Outcome |
|---|---|---|
| Class a | Example 1 | 1 |
| Class a | Example 2 | 1 |
| Class a | Example 3 | 3 |
| Class a | Example 4 | 4 |
| Class b | Example 1 | 1 |
| Class b | Example 2 | 3 |
| Class b | Example 3 | 3 |
| Class b | Example 4 | 4 |
| Class c | Example 1 | 2 |
| Class c | Example 2 | 2 |
| Class c | Example 3 | 2 |
| Class c | Example 4 | 4 |

The correspondence from repository 1 to repository 2 – the A matrix in the Xclassify algorithm - is calculated as below, using the outcome from classifying the examples from repository 1 with the classifier from repository 2:

| | Class a | Class b | Class c |
|---|---|---|---|
| Class 1 | 2/3 | 1/3 | 0 |
| Class 2 | 0 | 0 | 1 |
| Class 3 | 1/3 | 2/3 | 0 |
| Class 4 | 1 | 0 | 0 |

The correspondence from repository 2 to repository 1 – the B matrix in the Xclassify algorithm - is calculated as below, using the outcome from classifying the examples from repository 2 with the classifier from repository 1:

| | Class 1 | Class 2 | Class 3 | Class 4 |
|---|---|---|---|---|
| Class a | 1/2 | 0 | 1/4 | 1/4 |
| Class b | 1/4 | 0 | 1/2 | 1/4 |
| Class c | 0 | 3/4 | 0 | 1/4 |

The association score – the C matrix in the Xclassify algorithm - is calculated as below:

| | Class a | Class b | Class c |
|---|---|---|---|
| Class 1 | 1/3 | 1/12 | 0 |
| Class 2 | 0 | 0 | ¾ |
| Class 3 | 1/12 | 1/3 | 0 |
| Class 4 | 1/4 | 0 | 0 |

Based on this table, we can then conclude the following association rules:

- Class {2} from repository 1 corresponds exactly to class {c} from repository 2 (one-to-one);
- Classes {1,3} from repository 1 correspond to classes {a,b} from repository 2 (many-to-many), and no further reduction or decomposition of the association is possible based on the existing information;
- Class {4} from repository 1 maps to class {a} from repository 2, but class {a} also corresponds to class {1,3} from repository 1.

Note that the numerical value of the correspondence represents the strength of the correspondence between two classes (or two groups of classes). A threshold can be established at the table, column or row levels to prune those cells that are close to zero.

## 5. DISCUSSION

The classifier proposed in Section 3 has two responsibilities: (1) extract the features from the examples (2) produce label(s) from the features extracted from the examples according to a predefined vocabulary or taxonomy. As a result, interoperability among repositories is achieved through standardizing the format of the examples. For image repositories, the minimum requirement for interoperable repositories is that every repository understands a common set of image formats (tiff, jpeg, gif, etc.). Substantial computation may be required in order to extract features from all of the examples of a repository. Alternatively, standardized feature descriptors such as those currently being defined within the MPEG-7[9,10,11,12,13] effort can be used to describe examples.

These descriptors can be used directly as inputs to the classifier, and thus reducing the computational requirements on the classifier. MPEG-7 Descriptors for images include (1) low-level features, such as color histogram, texture, shape, etc (2) high-level semantics, including those syntactic and semantic descriptions of the objects within an image (3) metadata, including authors, location, and time of the image.

Note that the classifier from each repository needs to be trained on a common set of descriptors. As a result, each repository will be required to establish:

- Pre-trained classifier expressed in terms of portable format which can be executed in a heterogeneous environment
- Schema for describing the vocabulary/taxonomy, including the descriptors of the examples
- Descriptors for the examples

This information will be exchanged when two repositories begin to establish the mapping rules between two vocabularies.

## 6. SUMMARY

This paper proposed an algorithm for evaluating semantic similarity between entities stored in two information repositories, where entities are defined in terms of sets of class labels produced by supervised classifiers.

We assume that a query will contain a set of definitions of classes to be searched. Each such definition will consist of a set of training data (examples), and a trained classifier. Similarly, we assume that the repository contains a trained classifier, with the training data and associated labels available.

The proposed algorithm consists of determining similarity between each class defined in each repository. The question to be answered is "How well does class A from repository 1 correspond to class X in repository 2?" This is accomplished by providing the training set from the repository 1 as input to the trained classifier from repository 2, and symmetrically, providing the training set from the repository as input to the trained. The two sets of output labels, with associated scores, are input to a inference engine which reconciles the two sets, and outputs a ranking score for each query class - repository class combination. From these ranking scores, we can then derive one-to-one, one-to-many, many-to-one, and many-to-many correspondence between two vocabularies.

## 7. ACKNOLWEDGEMENT

This paper was supported in part by contract NASA/IBM Cooperative Agreement Notice 5-305.

## 8. REFERENCES

1. G. Wiederhold, "Mediators in the architecture of future information systems," IEEE Computer, vol. 25, no. 3, pp. 51-60, March 1992.
2. J. D. Ullman, "Information Integration Using Logical Views," Proc. ICDT, Jan. 1997.
3. M. Beigi, A. B. Benitez, and S.-F. Chang, "MetaSeek: A Content Based Meta Search Engine for Images," SPIE Storage and Retrieval for Image and Video Databases VI, vol. 3312, 1997 pp.118-128.
4. K. C. C. Chang, H. Garcia-Molina, "Approximate Query Translation Across Heterogeneous Information Sources", Proc. VLDB, Sept. 2000.
5. K. C. C. Chang, H. Garcia-Molina, "Mind Your Vocabulary: Query Mapping Across Heterogeneous Information Sources," Proc. ACM SIGMOD, June, 1999.
6. R. Hull, "Managing semantic heterogeneity in databases: a theoretical perspective," Proc. ACM PODS, pp. 51-61, 1997.
7. O. Bukhres, Z. B. Miled, E. Lynch, L. Olsen, Z. Tari, "Effective Standards for Metadata in the GCMD data Access System," Proc. DOA'00, Belgium 21-23, Sept. 2000.
8. G. R. Major, T. Byrd, and R. Vogel, "Citing Earth Science data: the Global Change Master Directory as a resource for the citation of earth science data sets," 2nd International joint AESE/CBE/EASE joint meeting, Sept. 1998, pp. 25-28.
9. MPEG Multimedia Description Scheme Group, "MPEG-7 Multimedia Description Schemes XM (v4.0)", ISO/IEC JTC1/SC29/WG11 MPEG00/N3465, Beijing, CN, July 1999.
10. MPEG Multimedia Description Scheme Group, "MPEG-7 Multimedia Description Schemes WD (v4.0)", ISO/IEC JTC1/SC29/WG11 MPEG00/N3465, Beijing, CN, July 1999.
11. MPEG Requirements Group, "MPEG-7: Context, Objectives and Technical Roadmap, V.12", ISO/IEC JTC1/SC29/WG11 MPEG99/N2861, Vancouver, July 1999.
12. J. R. Smith, "Quantitative Assessment of Image Retrieval Effectiveness", To appear in Journal of Information Access.
13. J. R. Smith and A. B. Benitez, "Conceptual Modeling of Audio-Visual Content", Proc. Intl. Conf. On Multimedia and Expo (ICME-2000), July 2000.