Article Title Page

Classification of Scientific Publications According to Library Controlled Vocabularies: A New Concept Matching-based Approach

Author Details *(please list these in the order they should appear in the published article)*

Author 1 Name: **Arash Joorabchi***
Department: **Electronic and Computer Engineering**
University/Institution: **University of Limerick**
Town/City: **Limerick**
State (US only):
Country: **Ireland**

Author 2 Name: **Abdulhussain E. Mahdi***
Department: **Electronic and Computer Engineering**
University/Institution: **University of Limerick**
Town/City: **Limerick**
State (US only):
Country: **Ireland**

***Both authors contributed equally to this work**

NOTE: affiliations should appear as the following: Department (if applicable); Institution; City; State (US only); Country.
No further information or detail should be included

Corresponding author: Arash Joorabchi
Corresponding Author's Email: Arash.Joorabchi@ul.ie

□ *Please check this box if you do not wish your email address to be published*

Acknowledgments (if applicable):
This work is supported by the OCLC/ALISE Library & Information Science Research Grant Program (LISRGP) 2012 and the Irish Research Council New Foundations scheme 2012.

Biographical Details (if applicable):

Arash Joorabchi is currently a postdoctoral researcher in the Department of Electronic & Computer Engineering, University of Limerick, Ireland. He earned his 1st Class Honours BSc in Computer Science from Griffith College Dublin, Ireland in 2006 and received his PhD in 2010 from the Department of Electronic & Computer Engineering, University of Limerick, Ireland. The main focus of Arash's research is on developing and deploying text mining\analytics algorithms and techniques to automate the process of metadata generation in digital libraries and repositories. His major areas of interest include: Data Mining, Knowledge Organization, Text Analytics, Digital Libraries, and Linked Data. To find out more about Arash's work, visit http://www.csn.ul.ie/~arash/

Abdulhussain E. Mahdi is a senior lecturer at the Department of Electronic & Computer Engineering, University of Limerick – Ireland. He is a Chartered Engineer (CEng), Member of the Institution of Engineering and Technology - UK (IET), Member of the Engineering Council - UK, and Founder Member of the International Compumag Society (ICS). Dr. Mahdi is a graduate in Electrical Engineering from University of Basrah (BSc 1st Class Hon. 1978) and earned his PhD in Electronic Engineering at University of Wales – Bangor, UK in 1990. He is also a SEDA Accredited Teacher of Higher Education (University of Plymouth, UK 1998). His research interests include speech and natural language processing, data mining, machine learning & applications in text analytics, telecoms & biomed; real-time DSP tools for domain transformation, time-frequency analysis, and DSP-based controllers. He has authored and co-authored more than 110 peer-reviewed journal articles, book chapters and conference papers, and has edited one book. His published work has been cited in more than 86 journal articles.

Structured Abstract:

Purpose – this paper reports on the design and development of a new approach for automatic classification and subject indexing of research documents in scientific Digital Libraries and Repositories (DLR) according to library controlled vocabularies such as DDC and FAST.

Design/methodology/approach – the proposed Concept Matching-based Approach (CMA) detects key Wikipedia concepts occurring in a document and searches the OPACs of conventional libraries via querying WorldCat database to retrieve a set of MARC records which share one or more of the detected key concepts. Then the semantic similarity of each retrieved MARC record to the document is measured and, using an inference algorithm, the DDC classes and FAST subjects of those MARC records which have the highest similarity to the document are assigned to it.

Findings – the performance of the proposed method in terms of the accuracy of the DDC classes and FAST subjects automatically assigned to a set of research documents is evaluated using standard information retrieval measures of precision, recall, and F1. We have demonstrated the superiority of the proposed approach in terms of accuracy performance in comparison to a similar system currently deployed in a large scale scientific search engine.

Originality/value – the proposed approach enables the development of a new type of subject classification systems for DLR, and addresses some of the problems similar systems suffer from, such as the problem of imbalanced training data encountered by machine learning-based systems, and the problem of word-sense ambiguity encountered by string matching-based systems.

Keywords: Scientific digital libraries and repositories, Metadata generation, Subject metadata, Dewey Decimal Classification (DDC), FAST subject headings, Automatic classification, Subject indexing, Concept matching, Wikipedia, WorldCat

Article Classification: Research paper

For internal production use only

Running Heads:

Research paper

Classification of Scientific Publications According to Library Controlled Vocabularies: A New Concept Matching-based Approach

1. Introduction

The use of open access scientific Digital Libraries and Repositories (DLR) is fast-growing within research and academic communities. They provide open access platforms for efficient dissemination of research output by individuals or groups in research-oriented organizations such as universities, research and development companies, national research labs, centres, and institutes. The research output comprises scientific publications including journal articles, conference papers, technical reports, theses and dissertations, book chapters, and other materials about the theory, practice, and results of scientific inquiry. The size of DLR collections vary from a few thousands, e.g., small institutional repositories, to hundreds of thousands, e.g., arXiv[1], and even millions, e.g., PMC[2] (Adamick and Reznik-Zellen, 2010). Also, specialized search engines such as CiteSeerX[3] and BASE[4] harvest, aggregate and index up to tens of millions of academic open access materials archived in institutional repositories, authors' webpages, etc. As the practice of open-access archiving grows due to the policy and enforcement initiatives taken by many research funding agencies, and as DLR software systems mature, it is expected that the size of DLR collections will grow exponentially. However, as these collections grow in size, finding the most relevant and up-to-date archived materials becomes challenging for the patrons. This is due to the fact that a great majority of current DLR systems rely solely on traditional keyword-based search methods which are prone to yield a large volume of indiscriminate search results irrespective of their content. Therefore, in order to facilitate precision search and discovery of archived materials, which enables patrons to focus their exploration efforts on the most relevant items of interest and reduces the recall effort, i.e., the ratio of desired to examined, we need to go beyond the traditional keyword-based search methods currently deployed.

Classification and subject indexing of archived materials according to library controlled vocabularies can enhance the performance of DLR search and discovery services. They also facilitate browsing the collections by category, e.g., Dewey Decimal Classification (DDC) system or subject, e.g., Library of Congress Subject Headings (LCSH). For example, the study of users navigation behaviours in a large-scale European meta subject gateway, Renardus, via log analysis by Traugott et al. (2004) showed that the directory-style of browsing in the DDC-based browsing structure was clearly the dominant activity, constituting 60% of all activities. However, manual classification and subject indexing of archived materials in DLR collections is a resource-intensive task which requires expert cataloguers in each knowledge domain represented in the collection and, therefore, deemed impractical in many cases due to the sheer volume of new materials published on daily basis. For example, reportedly the number of new publications in the field of biomedical science alone exceeds 1,800 a day (Hunter and Cohen, 2006). Methods and approaches reported in the library and information science literature to address this problem by automating the classification and subject indexing process can be divided into two main categories:

1. String matching-based systems: these systems rely on a method which consists of string-to-string matching between words in a list of terms extracted from library thesauri and classification schemes, and words in the textual content of the document to be classified. In this approach, an unlabelled document can be thought of as a search query against the library classification schemes and thesauri, where the search results include the most probable classes and subjects for the document. One of the well-known examples of such systems is the Scorpion project by OCCL Research (Roger et al., 1997, Godby and Smith, 2000). Scorpion builds a set of reference clusters for DDC classes and deploys a term-frequency distance measure to find the most relevant cluster (and consequently DDC class) for the document to be classified. A similar experiment was conducted earlier by Larson (1992) who built normalised clusters for 8,435 classes in the Library of Congress Classification (LCC) scheme from manually classified MARC records[5] of 30,471 library holdings and experimented with a variety of term representation and matching methods. Golub (2006) and Yi (2007) provide reviews of similar string-matching based systems deployed in various web classification and subject

indexing projects such as Pharos (Dolin et al., 1999), WWlib (Jenkins et al., 1998), and GERHARD (Möller et al., 1999).

2. Machine learning-based systems: these systems utilize generic Machine Learning (ML) algorithms such as Naïve Bayes (NB), Support Vector Machines (SVM), and *k*-Nearest Neighbours (*k*-NN) to classify documents according to library thesauri and classification schemes. These systems aim to combine the power of ML-based text classification methods with the enormous intellectual effort that has been put into developing library controlled vocabularies over the last century. Chung and Noh (2003) built a specialised web directory for the field of economics by classifying web pages into 757 sub-categories of economics category in the DDC scheme using *k*-NN algorithm. Pong et al. (2008) developed a system for automatic classification of web pages and digital library holdings based on the LCC scheme. They experimented with both *k*-NN and NB algorithms and compared the results. Frank and Paynter (2004) used the linear SVM algorithm to classify over 20,000 scholarly Internet resources based on the LCC scheme. Wang (2009) experimented with both NB and SVM algorithms for automatic classification of a bibliographic dataset according to the DDC scheme and compared the results. Waltinger et al. (2011) used SVM algorithm to classify scientific documents archived in DLR collections according to the DDC by relying solely on the Open Access Initiative (OAI) metadata records of the documents as their representation. The developed system is deployed in the Bielefeld Academic Search Engine (BASE) (Lösch, 2011) to classify scientific documents within three top levels of the DDC hierarchy.

Golub et al. (2006) have conducted an objective performance comparison between the string matching-based approach and the ML-based approach. The results of this study show that the ML-based approach outperforms the string matching-based approach by a large margin. It also shows that combining the two approaches does not yield an improved accuracy performance. These findings indicate that the ML-based approach is superior to the string matching-based approach. However, as discussed in (Wang, 2009), the large-scale and complexities of library controlled vocabularies impose great obstacles on popular supervised ML-based classification algorithms, such as NB and SVM, and prevent them from reaching the high accuracy performances that these classifiers have reportedly achieved on standard benchmark text classification datasets. These obstacles include: (a) deep hierarchy, where the classification hierarchical tree can go as deep as twenty levels; (b) skewed data distribution, where the great majority of training instances belong to a small number of classes; and (c) data sparseness, where there is a substantial number of classes which only have a few training instances and, hence, not sufficient for creating an accurate classification model.

In this work, we propose a new approach to automatic classification and subject indexing of scientific documents according to library controlled vocabularies, called Concept Matching-based Approach (CMA). CMA provides an effective and efficient alternative to ML-based and string matching-based approaches for practitioners in DLR development.

The rest of the paper is organised as follows: Section 2 introduces the CMA and describes the implementation details of a prototype automatic subject metadata generation system for DLR, developed based on the CMA to evaluate its performance and demonstrate its viability. Section 3 describes the evaluation process and presents its results. This is followed by Section 4 which provides a conclusion along with a summary account of planned future work.

2. Concept Matching-based Approach (CMA)

As illustrated in Figure 1, the CMA is based on automating the following main processes:

1. Detecting Wikipedia concepts in the full text of the scientific document to be classified and indexed.
2. Ranking detected concepts in terms of their relevance to the document and its core subject and topics, and filtering those with the highest keyness probability scores.
3. Searching library catalogues for MARC records containing the document's key concepts and retrieving the most relevant records.
4. Inferring the most probable class(es) and subjects for the document based on the classification and subject metadata of the retrieved MARC records which share one or more key concepts with the document and, therefore, have a high probability to be semantically relevant to the document and share its core subject and topics.

Figure 1. Illustration of the main processes in the proposed Concept Matching-based Approach

2.1. Concept Detection

The CMA classification process starts by detecting all Wikipedia concepts occurring in the document. A Wikipedia concept is an entity for which there exists a representing article in Wikipedia. In our approach, Wikipedia as a crowd-sourced controlled vocabulary serves two main purposes: (a) enriching the subject metadata of the document directly by adding the descriptors of the indentified key Wikipedia concepts to the set of uncontrolled index terms (MARC field 653) of the document, (b) acting as an intermediary controlled vocabulary which facilitates automatic classification and subject indexing of the document according to library controlled vocabularies. Our reasons for adopting Wikipedia for above purposes are:

1. Comprehensive coverage: at the time of writing this paper, the English Wikipedia contains more than four million articles[6], which makes it the most comprehensive controlled vocabulary currently existing.
2. Up-to-dateness: the crowd-sourced nature of Wikipedia which allows anyone to create and edit its articles makes it exceptionally up-to-date. For example, a recent study on combining Twitter and Wikipedia for event detection shows that in case of major events Wikipedia lags Twitter only by about three hours (Osborne et al., 2012).
3. Richness of descriptive content: the majority of Wikipedia articles provide a rich descriptive content for their corresponding concepts. This is in contrast with traditional library controlled vocabularies (e.g., DDC, LCSH) which provide very little or no descriptive information about their classes and subject headings. Therefore, when documents are indexed with Wikipedia concepts, DLR patrons have the option to refer to the corresponding Wikipedia articles of the unfamiliar concepts they encounter and read their comprehensive description.
4. Semantic richness: each Wikipedia article has a descriptor which is the preferred and most commonly used term for the represented concept. Also each article is assigned a set of non-descriptors which are the less common synonyms and alternative lexical forms for the represented concept. This in effect turns Wikipedia to a thesaurus which is semantically enriched by the linkage among the articles (Related Terms) and the classification of articles according to the Wikipedia's community-built classification scheme (Broader Terms/Narrower Terms). Furthermore, Wikipedia addresses the problem of word-sense ambiguity (Beall, 2011) by allowing an ambiguous term to correspond to multiple articles each representing and describing a different sense of the term, e.g., Java (programming language), Java (town), Java (band), etc.

Following the work of Medelyan et al. (Medelyan et al., 2008, Medelyan, 2009), we utilize an open-source toolkit called Wikipedia-Miner (Milne, 2009) for detecting Wikipedia concepts occurring in a document. Wikipedia-Miner effectively unlocks Wikipedia as a general-purpose knowledge source for natural language processing (NLP) applications by providing rich semantic information on concepts and their lexical representations. We use the topic/concept detection functionality of the Wikipedia-Miner to identify all the Wikipedia concepts (i.e., Wikipedia articles) whose descriptor or non-descriptor lexical representations occur in the document.

2.2. Concept Ranking

In order to rank detected Wikipedia concepts in a scientific document according to their importance and relevance to the core subject(s) of the document, we have utilized a set of seventeen statistical, positional, and semantical features for concepts. These features aim to capture and reflect various properties of those concepts which have the highest keyness probability:

1. Concept Frequency (CF): the occurrence frequency of the concept in the document. This includes the descriptor of the concept and also its non-descriptors such as synonyms and alternative lexical forms/near-synonyms occurring in the document. The TF values are normalized by dividing them by the highest TF value of a concept in the document.
2. First Occurrence: the distance between the start of the document and the first occurrence of the concept, measured in terms of the number of characters and normalized by the length of the document.
3. Last Occurrence: the distance between the end of the document and the last occurrence of the concept, measured in terms of the number of characters and normalized by the length of the document.
4. Occurrence Spread: the distance between the first and last occurrences of the concept, measured in terms of the number of characters and normalized by the length of the document.
5. Length: the number of words in the descriptor of the concept.
6. Lexical Unity: a Wikipedia concept could appear in a document in various lexical forms, i.e., descriptor and non-descriptors, which is quantified by the lexical unity measurement.

7. Average Link Probability: the average value of the link probabilities of all the concept's lexical forms which occur in the document. The link probability of a lexical form is the ratio of the number of times it occurs in Wikipedia articles as a hyperlink (directing to its corresponding article) to the number of times it occurs as plain text.
8. Max Link Probability: the maximum value of all link probabilities of the lexical forms for a concept which appears in the document.
9. Average Disambiguation Confidence: in many cases a term from the document corresponds to multiple concepts in Wikipedia and hence needs to be disambiguated. We have set the Wikipedia-Miner's disambiguator to perform a strict disambiguation, i.e., each term in the document can only correspond to a single concept which has the highest probabilistic confidence. The value of the *average disambiguation confidence* feature for a concept is calculated by averaging the disambiguation confidence values of its descriptor and non-descriptor lexical forms that appear in the document.
10. Max Disambiguation Confidence: the maximum disambiguation confidence value among the lexical forms of a concept which appear in the document.
11. Link-Based Relatedness to Other Concepts: the Wikipedia-Miner measures the semantic relatedness between two concepts using a new approach called Wikipedia Link-based Measure (WLM). The *link-based relatedness to other concepts* feature value of a concept is calculated by measuring and averaging its relatedness to all the other concepts detected in the document.
12. Link-Based Relatedness to Context: the only difference between this feature and the *link-based relatedness to other concepts* is that the relatedness of the concept is only measured against those other concepts in the document which are unambiguous, i.e., their descriptor and non-descriptor lexical forms occurring in the document have only one valid sense.
13. Category-Based Relatedness to Other Concepts: we measure the category-based relatedness of two Wikipedia concepts as:

$$\text{Relatedness}(concept_a, concept_b) = 1 - \frac{\text{Distance}(concept_a, concept_b) - 1}{2D - 3} \qquad (\mathbf{1})$$

where D is the maximum depth of the taxonomy, i.e., 16 in case of the Wikipedia dump used in this work. The distance function returns the length of the shortest path between $concept_a$ and $concept_b$ in terms of the number of nodes along the path. The term $2D-3$ gives the longest possible path distance between two concepts in the taxonomy, which is used as the normalization factor.
14. Generality: the depth of the concept in the taxonomy measured as its distance from the root category in Wikipedia, normalized by dividing it by the maximum possible depth, and inversed by deducting the normalized value from 1.0.
15. Distinct Links Count: total number of distinct Wikipedia concepts which are linked in/out to/from the concept, normalized by dividing it by the maximum possible distinct links count value in Wikipedia.
16. Links Out Ratio: total number of distinct Wikipedia concepts which are linked out from the concept, divided by the *distinct links count* value of the concept.
17. Translations Count: number of languages that the concept is translated to in the Wikipedia, normalized by dividing it by the maximum possible translations count value in Wikipedia.

The number of Wikipedia concepts occurring in a document could range from tens to thousands depending on the length of the document. For example, the number of concepts per document in a collection of 20 research papers used in Section 3 as the evaluation dataset ranges from 131 for a 10-page paper to 708 for a 38-page paper with an average of 275 concepts per document. The function applied to rank the concepts and filter out those with highest keyness probabilities could be either supervised or unsupervised. Since all the concept features defined above are normalized to range from 0.0 to 1.0, a simple unsupervised ranking function could be defined as:

$$\text{Score}(concept_j) = \sum_{i=1}^{|F|} f_{ij} \qquad (\mathbf{2})$$

which computes the sum of all feature values of a given concept, $concept_j$, as its keyness score. After computing the score of all detected concepts in the document, top n key concepts with the highest keyness probabilities are filtered out. The main advantage of this unsupervised approach is that it does not involve a training process and, therefore, does not require any manually annotated documents for learning a rank and filtering function from. Hence, it may be readily applied to scientific document collections across all domains with minimum effort. We have already demonstrated the effectiveness of above features and ranking function for detecting key concepts in scientific documents and encourage readers to refer to (Joorabchi and Mahdi, 2013) for more details on these features and detailed evaluation results of the ranking function.

As Shown in Figure 1, the top key Wikipedia concepts resulted from applying the above process to a scientific document may be used directly to enrich its subject metadata as index terms, and also to assist in classification of the document according to conventional library controlled vocabularies as described in the rest of this section.

2.3. Querying WorldCat & Refining Key Concepts

In CMA we use the key Wikipedia concepts indentified in a scientific document as the starting point for inferring the most probable DDC class(es) and FAST subject heading(s) for the document. Dewey Decimal Classification (DDC) is the most widely used library classification scheme in libraries around the world. The Faceted Application of Subject Terminology (FAST) is a simplified version of the well-known Library of Congress Subject Headings schema (LCSH), designed to retain the rich vocabulary of LCSH while making it easier to understand and use[7] (Dean, 2004). The process starts by searching the WorldCat database[8] for MARC records which contain one or more of the top 30 key concepts appearing in the document. WorldCat is a union catalogue of more than 70,000 conventional libraries around the world. This is done by submitting the following SRU query[9] to the WorldCat Search API[10] per each key concept indentified in the document, *doc_key_concept*$_i$:

```
http://worldcat.org/webservices/catalog/search/sru
  ?query=srw.kw=[doc_key_concept_descriptor_i]
  AND srw.ln exact eng   AND srw.la all   eng
  AND srw.mt all   bks   AND srw.dt exact bks
  &servicelevel=full &maximumRecords=100 &sortKeys=relevance,,0 &wskey=[wskey]
```

where, the variable *doc_key_concept_descriptor*$_i$ contains the descriptor of the key concept being processed. The parameters *srw.ln* and *srw.la* restrict the search domain to English language; and parameters *srw.mt* and *srw.dt* restrict the type of materials included in the search results to books. The parameter *maximumRecords* sets the maximum number of returned results to one hundred and the parameter *sortKeys* specifies that the results should be sorted according to relevance in descending order. Each query returns 0 to 100 matching MARC records in MARCXML[11] format which at this stage are used to refine the set of key concepts indentified in the document. A key concept from the document, *doc_key_concept*$_i$, is added to the set of refined key concepts unless: (a) it does not match any MARC records, for example the Wikipedia concept "logical conjunction" does not occur in any MARC record in the WorldCat database; or (b) it matches too many MARC records (i.e., too generic), for example the concept "logic" occurs in 72,353 MARC records belonging to many different DDC classes, which indicates that it is too general and has little or no discriminative value; or (c) its keyness score is less than 80% of the score of the first top ranking key concept, and 10 key concepts have already passed conditions a and b and been added to the set of refined key concepts; or (d) 20 key concepts have already satisfied conditions (a), (b), (c) and been added to the refined set of key concepts. This process is described by the following pseudocode:

```
Input: set of key concepts identified in the document, Doc_Key_Concepts
Output: refined set of key concepts

Refined_Doc_Key_Concepts := {}
For each $doc\_key\_concept_i \in Doc\_Key\_Concepts$ Do :
        IF $total\_matches_i = 0$
        OR $\ln(total\_matches_i + 1) > \text{InDoc\_Score}(doc\_key\_concept_i)$
        OR ( $\text{InDoc\_Score}(doc\_key\_concept_i) < \text{InDoc\_Score}(doc\_key\_concept_1) \times 0.8$
                AND $|Refined\_Doc\_Key\_Concepts| >= 10$ )
        OR $|Refined\_Doc\_Key\_Concepts| >= 20$
        THEN   Discard $doc\_key\_concept_i$
        ELSE   $Refined\_Doc\_Key\_Concepts := Refined\_Doc\_Key\_Concepts \cup \{doc\_key\_concept_i\}$
        $Doc\_Key\_Concepts := Refined\_Doc\_Key\_Concepts$
```

The refining process described above reduces the number of key concepts from 30 to a minimum of 10 and a maximum of 20.

2.4. MARC Records Parsing, Concept Detection, and Classification

This step involves parsing the MARC records retrieved per document key concept, detecting Wikipedia concepts in metadata elements of each record, and finding the most common DDC classes and FAST subject headings assigned to the works represented by the retrieved MARC records.

Figure 2. Parsing, concept detection, and classification of MARC records

As illustrated in Figure 2, we first parse the textual content of the following metadata fields from each MARC record: control number, title statement, formatted contents note, summary, subject added entry-topical term, and index term-uncontrolled. We then use the Wikipedia-Miner toolkit to detect all Wikipedia concepts that occur in the above parsed metadata fields. The detected concepts in a MARC record are used to measure the semantic similarity between the MARC record and the document to be classified and indexed as described in 2.5. Finally we use the control number of the MARC record to query the OCLC Classify database[12] (Vizine-Goetz, 2010) to find the most frequently assigned DDC class and FAST subject headings to the work represented by the MARC record, based on its number of library holdings. This is achieved by sending a REST query to the Classify API[13] per each MARC record in the following format:

```
http://classify.oclc.org/classify2/Classify?stdnbr=[ControlNumber]&maxRecs=1&su
mmary=false
```

where, the variable *ControlNumber* contains the control number of the MARC record being processed. The returned result in XML format contains the most popular DDC class and FAST subjects for the work represented by the MARC record according to the OCLC FRBR Work-Set algorithm[14] (Hickey et al., 2002), which will be parsed and added to the MARC record.

2.5. Measuring Relatedness between MARC Records and the Document

This step involves measuring the semantic similarity between each MARC record and the document to be classified and indexed. This is achieved by examining the Wikipedia concepts shared between the MARC record and the document to compute a single relatedness value for each MARC record using the following formula:

$$\text{Relatedness}\left(Marc\_Concepts_{i,j}, Doc\_Key\_Concepts\right) = \frac{\begin{array}{l}\sum_{k=1}^{|Shared\_Concepts|} \log_2\left(\text{Normalized\_Freq}\left(shared\_concepts_k\right)+1\right) \\ \times \log_2\left(\text{Inverse\_Marc\_Freq}\left(shared\_concept_k\right)\right) \\ \times \text{InDoc\_Score}\left(shared\_concepts_k\right)^2\end{array}}{|Marc\_Concepts|} \qquad \textbf{(3)}$$

where,

$$Shared\_Concepts = \left\{x \in \left(Marc\_Concepts_{i,j} \cap Doc\_Key\_Concepts\right) : x \neq doc\_key\_concepts_i\right\} \;,$$

$$\text{Normalized\_Freq}\left(shared\_concepts_k\right) = \frac{\text{InMarc\_Freq}\left(shared\_concepts_k\right)}{|Marc\_Concepts|} \;,$$

$$\text{Inv\_Marc\_Freq}\left(shared\_concept_k\right) = \frac{|All\_Uniq\_Mark\_Recs|}{\left|\left\{marc\_recs_{i,j} \in All\_Uniq\_Marc\_Recs : shared\_concept_k \in Marc\_Concepts_{i,j}\right\}\right|}$$

$$All\_Uniq\_Marc\_Recs = \left(\bigcup_{i=1}^{|Doc\_Key\_Concepts|} Marc\_Recs_i\right) \;.$$

The relatedness function measures the semantic similarity between the set of concepts, *Marc_Concepts$_{i,j}$*, identified in a MARC record, *marc_recs$_{i,j}$*, and the set of key concepts indentified in the document, *Doc_Key_Concepts*, based on the return values of three subfunctions: Normalized_Freq, Inv_Marc_Freq, and InDoc_score. The Normalized_Freq function returns the normalized occurrence frequency of a concept shared between the MARC record and the document, *shared_concepts$_k$*, within the MARC record. The Inv_Marc_Freq is equivalent to the well-known Inverse Document Frequency (IDF) term weighting function (Jones, 2004) commonly used in information retrieval systems. The Inv_Marc_Freq returns an importance weight for a given shared concept, *shared_concepts$_k$*, depending on its rare or commonness among all the unique MARC records in the collection. The rarer the concept, the higher its Inv_Marc_Freq value, which indicates the concept's higher discriminatory potential. The third function, InDoc_Score, simply returns the keyness score of the shared concept within the document as computed by Equation 2. InDoc_Score function enables the keyness score of the shared concept to be taken into account in the relatedness measurement of the MARC record to the document. In the relatedness function given in Equation 3, the two less reliable parameters: Normalized_Freq and Inv_Marc_Freq are set to contribute to the relatedness value logarithmically, whereas the stronger parameter, InDoc_Score, is set to contribute exponentially. The relatedness values of MARC records is used as a strong factor for determining the most probable DDC classes and FAST subjects for the documents.

2.6. Weighting Candidate DDC Classes and FAST Subject Headings

At this stage of the process we have a pool of MARC records each corresponding to one of the key concepts indentified in the document. The semantic relatedness of each MARC record to the document is measured. Also the most popular DDC class and FAST subjects for the work represented by each MARC record is indentified. We use this gathered data to weight all the unique DDC classes and FAST subjects in the pool. The following formula is used to weight the DDC classes:

$$
\begin{aligned}
\text{Weight}(uniq\_ddcs_k) = {} & \log_2(\text{Freq}(uniq\_ddcs_k)) \times \log_2(\text{Normalized\_Freq}(uniq\_ddcs_k)) \\
& \times \log_2(\text{Inv\_Concept\_Freq}(uniq\_ddcs_k)) \\
& \times \log_2(Inv\_Avg\_Total\_Matches(uniq\_ddcs_k)+1) \\
& \times \text{Average\_Relatedness}(uniq\_ddcs_k)
\end{aligned}
$$

where,

$$
\begin{gathered}
Uniq\_DDCs = \{x \in DDC_{i,j} : \exists marc\_recs_{i,j} \in All\_Uniq\_Marc\_Recs(1 \le i \le |DKConcepts|, 1 \le j \le |Marc\_Recs_i|)\} \;, \\
All\_Uniq\_Marc\_Recs = \left(\bigcup_{i=1}^{|DKConcepts|} Marc\_Recs_i \right) \;; \\
\text{Freq}(uniq\_ddcs_k) = \sum_{i=1}^{|DKConcepts|} \sum_{j=1}^{|Marc\_\text{Re}cs_i|} [uniq\_ddcs_k \in DDC_{i,j}] \;, \\
\text{Normalized\_Freq}(uniq\_ddcs_k) = \sum_{i=1}^{|DKConcepts|} \frac{\sum_{j=1}^{|Marc\_\text{Re}cs_i|} [uniq\_ddcs_k \in DDC_{i,j}]}{\sum_{j=1}^{|Marc\_\text{Re}cs_i|} [\![|DDC_{i,j}| \ne 0]\!]} \times Max\_DDCs\_PerConcept \;, \\
Max\_DDCs\_PerConcept = \max\left\{ x \in \text{N} : \forall dkconcepts_i \in DKConcepts \left(x = \sum_{j=1}^{|Marc\_\text{Re}cs_i|} [\![|DDC_{i,j}| \ne 0]\!] \right) \right\} \;; \\
\text{Inv\_Concept\_Freq}(uniq\_ddcs_k) = \frac{|\, DKConcepts \,|}{|\{dkconcepts_i \in DKConcepts : \exists uniq\_ddcs_k \in DDC_{i,j}(1 \le j \le |Marc\_Recs_i|)\}|} \;, \\
\text{Inv\_Avg\_Total\_Matches}(uniq\_ddcs_k) = \frac{\text{Freq}(uniq\_ddcs_k)}{\sum_{i=1}^{|DKConcepts|} total\_matches_i [\exists uniq\_ddcs_k \in DDC_{i,j}(1 \le j \le |Marc\_Recs_i|)]} \;, \\
\text{Average\_Relatedness}(uniq\_ddcs_k) = \frac{\sum_{i=1}^{|DKConcepts|} \sum_{j=1}^{|Marc\_\text{Re}cs_i|} Relatedness_{i,j} [uniq\_ddcs_k \in DDC_{i,j}]}{\text{Freq}(uniq\_ddcs_k)} \;.
\end{gathered}
\tag{4}
$$

The weight of each unique DDC in the pool, *uniq_ddcs$_k$*, is computed as the product of the returned values for that DDC from five subfunctions: Freq, Normalized_Freq, Inv_Concept_Freq, Inv_Avg_Total_Matches, and Average_Relatedness. The Freq function simply counts the number of times that the given unique DDC is associated with a MARC record in the pool. The Normalized_Freq function is similar to Freq function with the difference that the unique DDC counts are normalized by dividing them by the total number of MARC records per concept which have valid DDC numbers assigned to them; this normalized count value is then scaled to a number in (1, 100] by multiplying it with the parameter *Max_DDCs_PerConcept*, which counts the maximum number of MARC records with valid DDCs, associated with a concept in the pool. Similar to Inv_Marc_Freq function defined in Equation 3, the Inv_Concept_Freq gives higher weights to those DDCs which are associated with less number of document key concepts. A key concept in the document could be found in many MARC records in WorldCat, however, as described in 2.3, we only retrieve the top 100 matching records; the Inv_Avg_Total_Matches function takes this fact into account by dividing the returned value of the Freq function by the total number of matching MARC records from which only 100 has been retrieved. Finally the Average_Relatedness function computes the average relatedness of the MARC records, which are assigned the given unique DDC class, to the document as measure by Equation 3.

After weighting all unique DDC classes in the collection, we use the same above method for weighting the unique FAST subjects in the pool:

$$\begin{aligned}\text{Weight}(uniq\_fasts_k) = {} & \log_2(\text{Freq}(uniq\_fasts_k)) \times \log_2(\text{Normalized\_Freq}(uniq\_fasts_k)) \\ & \times \log_2(\text{Inv\_Concept\_Freq}(uniq\_fasts_k)) \\ & \times \log_2(Inv\_Avg\_Total\_Matches(uniq\_fasts_k)+1) \\ & \times \text{Average\_Relatedness}(uniq\_fasts_k)\end{aligned}$$

where,

$$Uniq\_FASTs = \{x \in DDC_{i,j} : \exists marc\_recs_{i,j} \in All\_Uniq\_Marc\_Recs(1 \le i \le |DKConcepts|, 1 \le j \le |Marc\_Recs_i|)\} \quad ,$$

$$All\_Uniq\_Marc\_Recs = \left(\bigcup_{i=1}^{|DKConcepts|} Marc\_Recs_i \right) \quad ;$$

$$\text{Freq}(uniq\_fasts_k) = \sum_{i=1}^{|DKConcepts|} \sum_{j=1}^{|Marc\_\text{Re}cs_i|} [uniq\_fasts_k \in FAST_{i,j}] \quad ,$$

$$\text{Normalized\_Freq}(uniq\_fasts_k) = \sum_{i=1}^{|DKConcepts|} \frac{\sum_{j=1}^{|Marc\_\text{Re}cs_i|} [uniq\_fasts_k \in FAST_{i,j}]}{\sum_{j=1}^{|Marc\_\text{Re}cs_i|} [|FAST_{i,j}| \ne 0]} \times Max\_FASTs\_PerConcept \quad ,$$

$$Max\_FASTs\_PerConcept = \max\left\{ x \in \mathrm{N} : \forall dkconcepts_i \in DKConcepts \left(x = \sum_{j=1}^{|Marc\_\text{Re}cs_i|} [|FAST_{i,j}| \ne 0] \right) \right\} \quad ;$$

$$\text{Inv\_Concept\_Freq}(uniq\_fasts_k) = \frac{|\,DKConcepts\,|}{|\{dkconcepts_i \in DKConcepts : \exists uniq\_fasts_k \in FAST_{i,j}(1 \le j \le |Marc\_Recs_i|)\}|} \quad ,$$

$$\text{Inv\_Avg\_Total\_Matches}(uniq\_fasts_k) = \frac{\text{Freq}(uniq\_fasts_k)}{\sum_{i=1}^{|DKConcepts|} total\_matches_i [\exists uniq\_fasts_k \in FAST_{i,j}(1 \le j \le |Marc\_Recs_i|)]} \quad , \qquad \textbf{(5)}$$

$$\text{Average\_Relatedness}(unique\_fasts_k) = \frac{\sum_{i=1}^{|DKConcepts|} \sum_{j=1}^{|Marc\_\text{Re}cs_i|} Relatedness_{i,j} [uniq\_fasts_k \in FAST_{i,j}]}{\text{Freq}(uniq\_fasts_k)} \quad .$$

2.7. Weight Aggregation

After having all the candidate DDC classes and FAST subjects weighted, we iterate through all the candidates and aggregate the weight of those DDCs or FASTs which are related. The following pseudocode describes the weight aggregation process for the DDC classes:

```
Input: set of weighted unique DDC candidates, Uniq_DDCs
Output: set of unique DDC candidates with aggregated weights

Sort Uniq_DDCs set based on DDC candidates depth in descending order
For each $DDC_i \in Uniq\_DDCs$ Do :
    For each $DDC_j \in Uniq\_DDCs$ Do :
        IF subclass($DDC_i$ , $DDC_j$) THEN
            IF weight($DDC_i$) > highest_DDC_weight/10 THEN
                weight($DDC_i$) := weight($DDC_i$) + weight($DDC_j$)
                Discard $DDC_j$
            ELSE Discard $DDC_i$
```

For each candidate DDC class, DDC_i, we check if its parent class is among the candidates. If that is the case, and also the weight of the child class (DDC_i) is above a threshold (weight (DDC_i) > highest_DDC_weight/10), then the weight of the parent DDC class (DDC_j) will be added to the child and the parent will be discarded from the list of candidates. However, if the parent of a given DDC class is among the candidates but the weight of the child is below the threshold then the child will be discarded from the list of candidates.

The parent-child relationship among the DDC classes is encoded in the DDC class numbers, for example, the DDC class 006.31 (machine learning) is a subclass of class 006.3 (artificial intelligence). However, this is not the case for FAST subjects. Alternatively, we utilize two data elements in FAST subject records to deduce their parent-child relationship, namely: 'relatedness' and 'WorldCat subject usage'. The following pseudocode describes the weight aggregation process for FAST subjects:

```
Input: set of weighted unique FAST candidates, Uniq_FASTs
Output: set of unique FAST candidates with aggregated weights

$Uniq\_FASTs$ := {x ∈ $Uniq\_FASTs$ : weight(x) > highest_FAST_weight/10}
For each $FAST_i \in Uniq\_FASTs$ Do :
    For each $FAST_j \in Uniq\_FASTs$ Do :
        IF related($FAST_i$ , $FAST_j$) AND WC_SubjectUsage($FAST_i$) < WC_SubjectUsage($FAST_j$)
            THEN weight($FAST_i$) := weight($FAST_i$) + weight($FAST_j$)
```

The process starts by refining the set of candidate FAST subjects by removing those whose weight is less than 1/10 of the highest FAST weight in the set. We then iterate through the remaining candidates and add up the weight of those which appear to have a parent-child relationship. This is achieved by searching a locally stored copy of the FAST database[15] for each candidate FAST subject, $FAST_i$, and retrieving its record in MARCXML format. This record contains a list of other FAST subjects related to it (MARC field 550) and also shows how many times the subject has been used to index works catalogued in the WorldCat database (MARC field 688). If any of the related subjects is among the candidates and its WorldCat usage is greater than that of the candidate being processed, $FAST_i$, the weight of the related subject would be added to the $FAST_i$. For example, consider a case where the candidate subject being processed is "Expert systems (Computer science)" which according to its record has been used in WorldCat 14,685 times and is related to FAST subjects "Artificial intelligence", "Computer systems", and "Soft computing". In this case, if the subject "Artificial intelligence" is among the candidates, its weight would be added to the weight of the subject "Expert systems (Computer science)" as its WorldCat usage, which is 36,145 times, is greater than that of "Expert systems (Computer science)". Having a greater WorldCat usage between two related subjects does not guarantee their parent-child relationship, however based on our preliminary experiments it can be used as a strong indication of such relationship between two related subjects.

2.8. Outlier Detection

The final stage of the classification and subject indexing process involves choosing the most probable DDC classes and FAST subjects for the document according to the aggregated weights of the candidates. This is achieved by using boxplot outlier detection method to identify the extreme and/or mild upper outlier(s) in the candidate DDC and FAST sets. An outlier is a candidate DDC class or FAST subject whose weight value lies at an abnormal distance from the weight values of other candidates in the set such that:

$$\begin{aligned} \text{Weight}(uniq\_ddcs_k) > Q3 + 1.5(Q3 - Q1) \\ \text{Weight}(uniq\_fast_k) > Q3 + 1.5(Q3 - Q1) \end{aligned} \quad \textbf{(6)}$$

where, *Q1* and *Q3* represent lower and upper quartiles (defined as the 25th and 75th percentiles) in the respective data sets (i.e., Uniq_FASTs and Uniq_DDCs). Those DDC classes and FAST subjects whose weights pass the above outlier criterion plus the next highest weighting candidate in each set are chosen as the most probable DDC classes and FAST subjects for the document.

3. Experimental Results & Evaluation

For the purpose of evaluating the performance of the CMA in classification and subject indexing of documents using Wikipedia concepts and library controlled vocabularies, we have used a dataset called wiki-20[16] (Medelyan et al., 2008, Medelyan, 2009). The wiki-20 collection consists of 20 computer science (CS) related scientific documents, each manually annotated by fifteen different human teams independently. Each team consisted of two senior undergraduate and/or graduate CS students. The teams were instructed to assign about five key Wikipedia concepts to each document from a set of over two million concepts in English Wikipedia at the time the dataset was compiled. The detailed evaluation results of our key Wikipedia concept detection and ranking method (described in 2.1 and 2.2) on this dataset are reported in (Joorabchi and Mahdi, 2013). As shown in Table 1, performance of our concept detection and ranking method measured in terms of consistency with human annotators using Rolling's inter-indexer consistency formula (Rolling, 1981), is on a par with that achieved by humans and outperforms most of rival methods such as KEA++ (KEA-5.0) (Medelyan and Witten, 2008), (Grineva et al., 2009), Maui (Medelyan, 2009), and CKE (Mahdi and Joorabchi, 2010).

Table 1. Performance comparison with human annotators and rival machine annotators on the task of key concepts detection.

We have used standard measures of Precision (*Pr*), Recall (*Re*), and their harmonic mean, *F1*, to evaluate the performance of our system in automatic classification and subject indexing of the documents in the wiki-20 dataset according to library controlled vocabularies, i.e., DDC and FAST:

$$Pr = \frac{\text{Number of correctly assigned classes}}{\text{Total assigned}} = \frac{TP}{TP + FP} \quad \textbf{(7)}$$

$$Re = \frac{\text{Number of correctly assigned classes}}{\text{Total possible correct}} = \frac{TP}{TP + FN} \quad \textbf{(8)}$$

$$F1 = \frac{2Pr \times Re}{Pre + Re} \quad \textbf{(9)}$$

where, *Pr*, *Re*, and *F1* are computed in terms of the labels *TP* (True Positive), *FP* (False Positive), and *FN* (False Negative) to evaluate the validity of a given class label *i* assigned to a given document *j*, such that:

- *TP*: refers to the cases when both the classifier and human cataloguer agree on assigning class label *i* to document *j*;
- *FP*: refers to the cases when the classifier has mistakenly (as judged by a human cataloguer) has assigned class label *i* to document *j*;
- *FN:* refers to the cases when the classifier has failed (as judged by a human cataloguer) to assign a correct class label *i* to document *j*.

We have evaluated the performance of our method in classifying documents according to the DDC scheme in two different modes, namely: binary and hierarchical evaluation. In the binary mode of evaluation, a DDC class assigned to a document is strictly considered as either true or false. In the hierarchical evaluation mode however, we examine the truthness of the assigned DDC class in each level of the DDC hierarchy individually. Table 2 shows the performance results of our method in the binary evaluation mode.

Table 2. DDC evaluation results in binary mode

As shown in Table 2, we have compared the performance of our method on the wiki-20 dataset with that achieved by the Automatic Classification Toolbox for Digital Libraries (ACT-DL)[17]. The ACT-DL is maintained by Bielefeld University Library and deployed at Bielefeld Academic Search Engine (BASE) (Lösch, 2011) to classify catalogued documents according to the DDC scheme. The ACT-DL is an ML-based system and deploys the SVM algorithm to classify scientific documents up to the third level of the DDC hierarchy[18] (Waltinger et al., 2011). In comparison, our method does not limit the depth of classification and is based on the full DDC. As shown in Table 2, the *F1* performance of our CMA in the binary mode of evaluation is 0.6 and, hence, it outperforms the ACT-DL with a large margin. The poor performance of the ACT-DL may be contributed to the imbalance that exists in the dataset used to train the classification model used by the ACT-DL. Imbalanced training data is a well-known issue encountered by ML-based systems and can greatly reduce their accuracy performance. Most of the documents in the wiki-20 dataset belong to one of the following three main classes in the DDC: 004 (Computer science), 005 (Computer programming, programs, data), or 006 (Special computer methods). However, since the great majority of manually classified documents used to train the ACT-DL belonged to the DDC class 004, the learnt model has developed a strong bias towards this class and has wrongly classified 14 out of 20 documents in the wiki-20 collection under this class. Consequently, the current number of documents classified under the DCC class 004 in BASE is more than 78,000, whereas the number of documents classified under DDC classes 005 and 006 are 100 and 403 respectively, which clearly shows an unjustified bias towards the DDC class 004.

Table 3 shows the results of the CMA evaluation in hierarchical mode and compares it with that achieved by the ACT-DL on the wiki-20 dataset, and also the BASE dataset (Lösch et al., 2011) as reported in (Waltinger et al., 2011). The documents in the wiki-20 dataset belong to classes as deep as seventh level of the DDC hierarchy and a few of them are multi-faceted. However, due to the limitation of the ACT-DL, we can only compare the results of our CMA with that of the ACT-DL up to the third level of the DDC hierarchy. As highlighted in Table 2, the CMA outperforms the ACT-DL in all three levels, where direct comparison is possible.

Table 3. DDC evaluation results in hierarchical mode

The Fast subjects assigned to the documents in the wiki-20 collection may only be evaluated in the binary mode as there is no formal parent-child relationship among the FAST subjects. Table 4 presents the evaluation results of the CMA in assigning FAST subjects to the wiki-20 documents.

All the data gathered and created during the classification and subject indexing process of each document in the wiki-20 dataset including the log of key concept detection and ranking, querying WorldCat and Classify databases, and the weighting and inference processes are available for download[19].

Table 4. FAST evaluation results

4. Conclusion and Future Work

In this article, we introduced a new concept matching-based approach to automatic classification and subject indexing of scientific documents archived in digital libraries and repositories according to library controlled vocabularies, namely DDC and FAST. The evaluation results of the proposed CMA are promising and outperform those achieved by a similar automatic classification system, ACT-DL, currently deployed in one of the largest academic search engines, BASE. CMA may be implemented and deployed as a plug-in for current DLR software systems, such as Fedora, EPrints, and DSpace. This plug-in would extract the textual content of new materials as they are being deposited, and classify them according to the DDC and FAST. The classification results, i.e., the most probable DDC classes and FAST subject headings for the documents may then be either directly added to their metadata records, or presented to the depositors first for approval/amendment prior to the addition.

A limitation of this study is the small size of the test dataset used which consists of 20 documents mostly in the field of computer science. However, it should be noted that manual classification of research documents according to library controlled vocabularies is a tedious and time consuming task for which we could not find a large and accurate dataset. Therefore, evaluating the performance of the CMA on a larger set of scientific documents in various fields of science in future would give us a better understanding of its overall accuracy performance. Also as future work, we plan to improve the performance of the proposed approach and its developed prototype in terms of both computational and accuracy performance by developing and applying the following enhancements:

a. Eliminating the need for sending queries to the WorldCat database and repeating the process of concept detection on matching MARC records by performing a once-off concepts detection on a locally held FRBRized version of the WorldCat database.
b. Complementing concepts extracted from MARC records of works catalogued in the WorldCat database with common terms and phrases from the content of those works as extracted by Google Books project[20].

Finally, this experimental work paves the way for future work on probabilistic mapping of Wikipedia concepts/articles to their corresponding DDC classes and FAST subjects, which has been already initiated by the OCLC Research via developing VIAFbot[21] for mapping Wikipedia biography articles to the Virtual International Authority File (VIAF)[22].

Notes

1. http://arxiv.org
2. http://www.ncbi.nlm.nih.gov/pmc
3. http://citeseerx.ist.psu.edu
4. http://www.base-search.net
5. http://www.loc.gov/marc
6. http://en.wikipedia.org/wiki/Wikipedia:Size_in_volumes
7. http://www.oclc.org/research/activities/fast.html
8. http://www.oclc.org/worldcat
9. http://www.loc.gov/standards/sru
10. http://oclc.org/developer/services/worldcat-search-api
11. http://www.loc.gov/standards/marcxml
12. http://www.oclc.org/research/activities/classify.html
13. http://classify.oclc.org/classify2/api_docs
14. http://www.oclc.org/research/activities/frbralgorithm.html
15. http://www.oclc.org/research/activities/fast/download.html
16. http://maui-indexer.googlecode.com/files/wiki20.tar.gz
17. http://act-dl.base-search.net
18. http://www.ub.uni-bielefeld.de/wiki/OAIMEnglishSum
19. http://www.skynet.ie/~arash/zip/wiki20_DDC_FAST.zip
20. http://books.google.com
21. http://www.oclc.org/research/news/2012/12-07a.html
22. http://viaf.org

Acknowledgements

This work is supported by the OCLC/ALISE Library & Information Science Research Grant Program (LISRGP) 2012 and the Irish Research Council New Foundations scheme 2012.

References

ADAMICK, J. & REZNIK-ZELLEN, R. 2010. Trends in Large-Scale Subject Repositories. *D-Lib Magazine,* 16.

BEALL, J. 2011. Academic Library Databases and the Problem of Word-Sense Ambiguity. *The Journal of Academic Librarianship,* 37**,** 64-69.

CHUNG, Y.-M. & NOH, Y.-H. 2003. Developing a specialized directory system by automatically classifying Web documents. *Journal of Information Science,* 29**,** 117-126.

DEAN, R. J. 2004. FAST: Development of Simplified Headings for Metadata. *Cataloging & Classification Quarterly,* 39**,** 331-352.

DOLIN, R., AGRAWAL, D. & ABBADI, E. E. 1999. Scalable collection summarization and selection. *Proceedings of the fourth ACM conference on Digital libraries.* Berkeley, California, United States: ACM.

FRANK, E. & PAYNTER, G. W. 2004. Predicting Library of Congress classifications from Library of Congress subject headings. *Journal of the American Society for Information Science and Technology,* 55**,** 214-227.

GODBY, C. J. & SMITH, D. 2000-2002. *Scorpion* [Online]. OCLC Online Computer Library Center, Inc. Available: http://www.oclc.org/research/activities/scorpion.html [Accessed February 2013].

GOLUB, K. 2006. Automated subject classification of textual Web pages, based on a controlled vocabulary: Challenges and recommendations. *New Review of Hypermedia and Multimedia,* 12**,** 11-27.

GOLUB, K., ARDÖ, A., MLADENIĆ, D. & GROBELNIK, M. 2006. Comparing and Combining Two Approaches to Automated Subject Classification of Text. *Research and Advanced Technology for Digital Libraries.* Springer Berlin / Heidelberg.

GRINEVA, M., GRINEV, M. & LIZORKIN, D. 2009. Extracting key terms from noisy and multi-theme documents. *18th international conference on World wide web.* Madrid, Spain: ACM.

HICKEY, T. B., O'NEILL, E. T. & TOVES, J. 2002. Experiments with the IFLA functional requirements for bibliographic records (FRBR). *D-Lib Magazine,* 8**,** 1-13.

HUNTER, L. & COHEN, K. B. 2006. Biomedical Language Processing: What's Beyond PubMed? *Molecular Cell,* 21**,** 589-594.

JENKINS, C., JACKSON, M., BURDEN, P. & WALLIS, J. 1998. Automatic classification of Web resources using Java and Dewey Decimal Classification. *Computer Networks and ISDN Systems,* 30**,** 646-648.

JONES, K. S. 2004. IDF term weighting and IR research lessons. *Journal of Documentation,* 60**,** 521-523.

JOORABCHI, A. & MAHDI, A. E. 2013. Automatic keyphrase annotation of scientific documents using Wikipedia and genetic algorithms. *Journal of Information Science 0165551512472138, first published on February 8, 2013 doi:10.1177/0165551512472138.*

LARSON, R. R. 1992. Experiments in automatic Library of Congress Classification. *Journal of the American Society for Information Science,* 43**,** 130-148.

LÖSCH, M. 2011. A Multidisciplinary Search Engine for Scientific Open Access Documents. *In:* DEPPING, R. & CHRISTIANE, S. (eds.) *Elektronische Schriftenreihe der Universitaïs- und Stadtbibliothek Koln, 2. Cologne: EBSLG Annual General Conference.*

LÖSCH, M., WALTINGER, U., HORSTMANN, W. & MEHLER, A. 2011. Building a DDC-annotated Corpus from OAI Metadata. *Journal of Digital Information,* 12.

MAHDI, A. E. & JOORABCHI, A. 2010. A Citation-based approach to automatic topical indexing of scientific literature. *Journal of Information Science,* 36**,** 798-811.

MEDELYAN, O. 2009. *Human-competitive automatic topic indexing.* PhD thesis Ph.D, University of Waikato, New Zealand.

MEDELYAN, O. & WITTEN, I. H. 2008. Domain-independent automatic keyphrase indexing with small training sets. *Journal of the American Society for Information Science and Technology,* 59**,** 1026-1040.

MEDELYAN, O., WITTEN, I. H. & MILNE, D. 2008. Topic Indexing with Wikipedia. *first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08).* Chicago, US: AAAI Press.

MILNE, D. 2009. An open-source toolkit for mining Wikipedia. *New Zealand Computer Science Research Student Conference.*

MÖLLER, G., CARSTENSEN, K.-U., DIEKMANN, B. & WÄTJEN, H. 1999. Automatic Classification of the World-Wide Web using the Universal Decimal Classification. *In:* DECKER, R. & GAUL, W. (eds.) *Proceedings of the 23rd Annual Conference of the German Classification Society (GfKl).* Bielefeld: Springer-Verlag.

OSBORNE, M., PETROVIC, S., MCCREADIE, R., MACDONALD, C. & OUNIS, I. 2012. Bieber no more: First Story Detection using Twitter and Wikipedia. *SIGIR Workshop in Time-aware Information Access (TAIA'12).* Portland, Oregon, USA: ACM.

PONG, J. Y.-H., KWOK, R. C.-W., LAU, R. Y.-K., HAO, J.-X. & WONG, P. C.-C. 2008. A comparative study of two automatic document classification methods in a library setting. *Journal of Information Science,* 34**,** 213-230.

ROGER, T., KEITH, S. & DIANE, V.-G. 1997. Evaluating Dewey concepts as a knowledge base for automatic subject assignment. *Proceedings of the second ACM international conference on Digital libraries.* Philadelphia, Pennsylvania, United States: ACM.

ROLLING, L. 1981. Indexing consistency, quality and efficiency. *Information Processing & Management,* 17**,** 69-76.

TRAUGOTT, K., ANDERS, A. & KORALJKA, G. 2004. Browsing and searching behavior in the renardus web service a study based on log analysis. *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries.* Tuscon, AZ, USA: ACM.

VIZINE-GOETZ, D. 2010. Classify: a FRBR-based research prototype for applying classification numbers. *OCLC NextSpace***,** 14-15.

WALTINGER, U., MEHLER, A., LÖSCH, M. & HORSTMANN, W. 2011. Hierarchical Classification of OAI Metadata Using the DDC Taxonomy. *In:* BERNARDI, R., CHAMBERS, S., GOTTFRIED, B., SEGOND, F. & ZAIHRAYEU, I. (eds.) *Advanced Language Technologies for Digital Libraries.* Springer Berlin Heidelberg.

WANG, J. 2009. An extensive study on automated Dewey Decimal Classification. *Journal of the American Society for Information Science and Technology,* 60**,** 2269-2286.

YI, K. 2007. Automated Text Classification Using Library Classification Schemes: Trends, Issues, and Challenges. *International Cataloguing and Bibliographic Control (ICBC),* 36**,** 78-82.

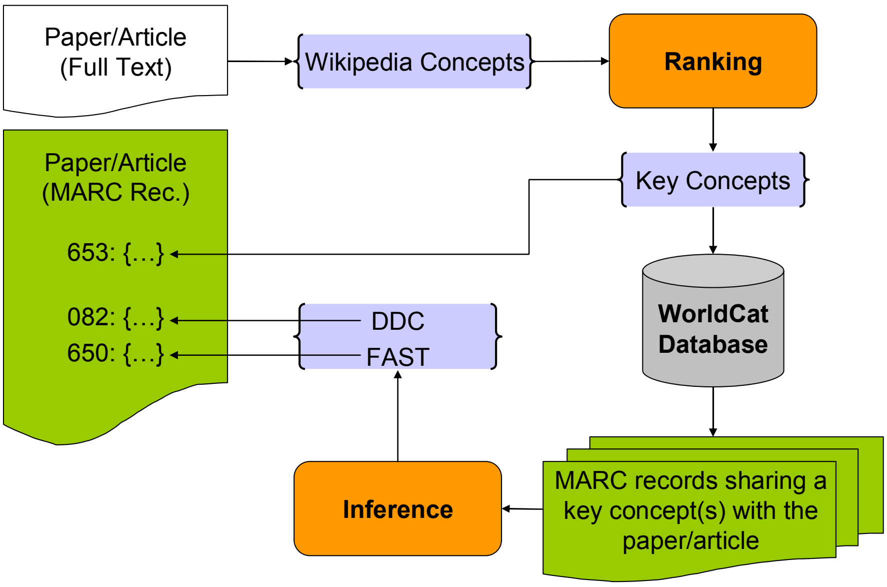
Paper/Article (Full Text)
Wikipedia Concepts
Ranking
Paper/Article (MARC Rec.)
653: {…}
082: {…}
650: {…}
Key Concepts
WorldCat Database
DDC
FAST
Inference
MARC records sharing a key concept(s) with the paper/article

Figure 1. Illustration of the main processes in the proposed Concept Matching-based Approach

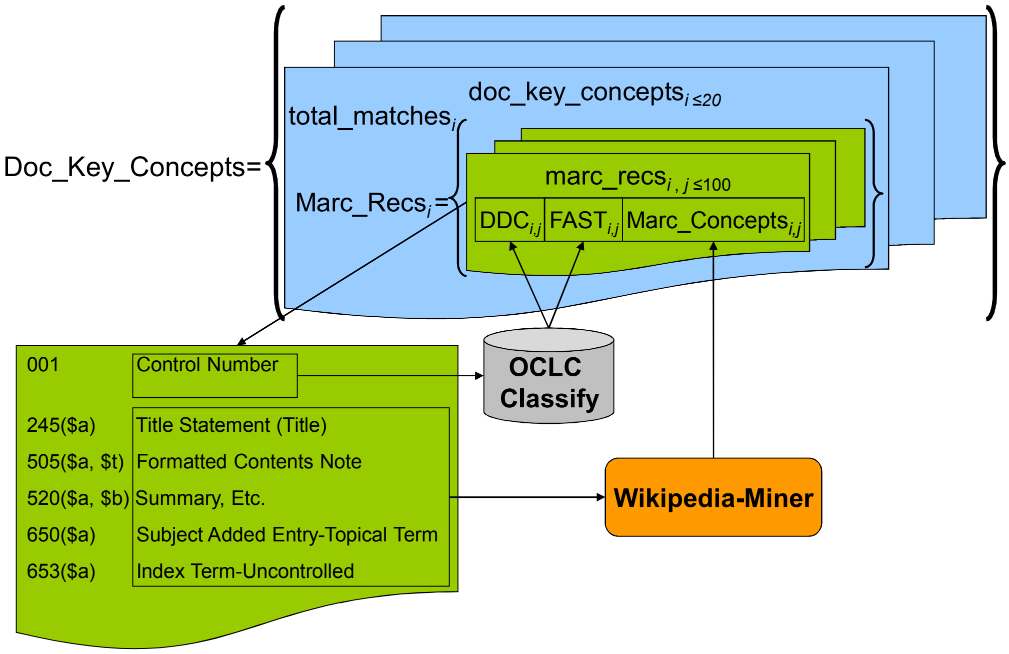

Figure 2. Parsing, concept detection, and classification of MARC records

Table 1. Performance comparison with human annotators and rival machine annotators on the task of key concepts detection.

Method	Learning Approach	Number of Key Concepts Assigned per document	Avg. inter consistency with human annotators (%)		
			Min.	**Avg.**	**Max.**
TFIDF (baseline)	n/a - unsupervised	5	5.7	8.3	14.7
KEA++ (KEA-5.0)	Naïve Bayes	5	15.5	22.6	27.3
Grineva et al.	n/a - unsupervised	5	18.2	27.3	33.0
Maui (Medelyan, 2009)	Naïve Bayes (all 14 features)	5	22.6	29.1	33.8
Maui	Bagging decision trees (all 14 features)	5	25.4	30.1	38.0
Human annotators (gold standard)	n/a - senior CS students	Varied, with an average of 5.7 per document	21.4	30.5	37.1
CKE	n/a - unsupervised	5	22.7	30.6	38.3
Current work	n/a - unsupervised	5	19.1	30.7	37.9

Table 2. DDC evaluation results in binary mode

Doc ID	**Predicted DDC (CMA)**	**True DDC**	**Predicted DDC (ACT-DL)**
287	519.542 Decision theory 006.35 Natural language processing	✓ ✓	00**4**
7183	006.333 Deduction, problem solving, reasoning	✓	00**4**
7502	00**5.131** Symbolic logic	00**6.333** Deduction, problem solving, reasoning	00**4**
9307	005.757**--0218** Object-oriented databases--Standards	005.757 Object-oriented databases	00**4**
10894	**621.3815--0287** Components and circuits--Testing and measurement	**005.14** Verification, testing, measurement, debugging	00**4**
12049	005.4**3** Systems programs	005.4**53** Compilers	00**4**
13259	001.**6443** *(invalid in DDC22 & DDC23)*	001.**4226** Presentation of statistical data	00**0**
16393	00**4.53** Internal storage (Main memory)	00**5.435** Memory management programs	00**4**
18209	005.115 Logic programming	✓	00**4**
19970	511.322 Set theory 005.275 Programming for multiprocessor computers	✓ ✓	00**4**
20287	004.35 Multiprocessing 004.33 Real-time processing	✓ ✓	004
23267	005.117 Object-oriented programming	✓	00**4**
23507	**495.6--5** Japanese--Grammar	**006.35** Natural language processing	**400**
23596	658.4036--028546 Group decision making--Computer communications	✓	**150**
25473	515.2433 Fourier and harmonic analysis *below threshold*	✓ **006.37** Computer vision	00**4**
37632	005.14 Verification, testing, measurement, debugging	✓	00**4**
39172	006.4--015116 Computer pattern recognition--Combinatorics	✓	**510**
39955	005.117 Object-oriented programming	✓	**150**
40879	00**4** Computer science	00**6.31** Machine learning	00**4**
43032	005.26**2** Programming in specific programming languages	005.26 Programming for personal computers	00**4**
Overall	**TP**= 14, **FP**=9, **FN**= 10, **Pr**= 0.61, **Re**= 0.58, **F1**= 0.60		**F1**= 0.05

Table 3. DDC evaluation results in hierarchical mode

	Level	L_1	L_2	L_3	L_4	L_5	L_6	L_7	Facet	Avg.
CMA (Wiki-20 dataset)	**TP**	21	21	18	17	15	10	2	2	
	FP	2	2	5	5	5	4	2	3	
	FN	3	3	6	7	8	4	1	0	
	Pr	**0.91**	**0.91**	**0.78**	**0.77**	**0.75**	**0.71**	**0.50**	**0.40**	**0.72**
	Re	**0.88**	**0.88**	**0.75**	**0.71**	**0.65**	**0.71**	**0.67**	**1.00**	**0.78**
	F1	**0.89**	**0.89**	**0.77**	**0.74**	**0.70**	**0.71**	**0.57**	**0.57**	**0.73**
ACT-DL (Wiki-20 dataset)	**Level**	**L_1**	**L_2**	**L_3**	**L_4**	**L_5**	**L_6**	**L_7**	**Facet**	**Avg.**
	TP	16	16	1	n/a					
	FP	4	4	19						
	FN	4	4	19						
	Pr	0.80	0.80	0.05						0.55
	Re	0.80	0.80	0.05						0.55
	F1	0.80	0.80	0.05						0.55
ACT-DL (BASE dataset)	**Level**	**L_1**	**L_2**	**L_3**	**L_4**	**L_5**	**L_6**	**L_7**	**Facet**	**Avg.**
	Pr	0.90	0.78	0.77	n/a					0.82
	Re	0.75	0.56	0.55						0.62
	F1	0.81	0.63	0.62						0.69

Table 4. FAST evaluation results

Doc ID	Predicted FAST	True FAST
287	Bayesian statistical decision theory	✓
	Bayesian statistical decision theory--Industrial applications	Natural language processing (Computer science)
	Maximum entropy method	Information retrieval
	Econometric models	Machine learning
7183	Model-based reasoning	✓
	Knowledge acquisition (Expert systems)	✓
	Expert systems (Computer science)	✓
7502	Semantics	Conceptual structures (Information theory)
	Case-based reasoning	✓
9307	Object-oriented databases	✓
	UML (Computer science)	Computer software—Development
	Booch method	Computer-aided software engineering
	Software patterns	✓
	Object-oriented methods (Computer science)	✓
	Object-oriented databases--Standards	Object-oriented programming (Computer science)
10894	Regression analysis	✓
	Struts framework	Computer software--Quality control
	Application software--Testing	✓
12049	Yacc (Computer file)	✓
	Assembling (Electronic computers)	Compiling (Electronic computers)
13259	Three-dimensional display systems	✓
	Interactive computer systems	✓
	Interactive multimedia	Information visualization
16393	Distributed shared memory	✓
	Intel i860 (Microprocessor)	Memory management (Computer science)
	Cache memory	✓
	Virtual storage (Computer science)	✓
18209	Predicate (Logic)	✓
	Modality (Logic)	✓
19970	Set theory	✓
	Sorting (Electronic computers)	✓
	Parallel algorithms	✓
20287	Data transmission systems	Real-time data processing
	Virtual computer systems	✓
	Parallel computers	✓
23267	Modula-3 (Computer program language)	Object-oriented methods (Computer science)
	ML (Computer program language)	Object-oriented programming (Computer science)
	Object-oriented databases	Computer software--Reusability
	Abstract data types (Computer science)	✓
23507	English language--Noun phrase	✓
	Grammar, Comparative and general--Noun phrase	✓
	Automatic speech recognition	Computational linguistics
23596	Teams in the workplace--Data processing	✓
25473	Data compression (Telecommunication)	✓
	Image compression	✓
	Signal processing--Mathematics	✓
	Wavelets (Mathematics)	✓
	Video compression	✓
	Digital video	✓
	Data compression (Computer science)	✓
37632	Software visualization	✓
	Debugging in computer science	✓
39172	Matching theory	✓
	Text processing (Computer science)	✓
	Graphical user interfaces (Computer systems)	Combinatorial analysis
39955	Smalltalk (Computer program language)	Object-oriented programming languages
	Objective-C (Computer program language)	Object-oriented programming (Computer science)
40879	Automatic speech recognition	Machine learning
	Speech processing systems	Classification
	Supervised learning (Machine learning)	✓
43032	HP-UX	Software localization
	Hewlett-Packard computers--Programming	User interfaces (Computer systems)
	HP 9000 (Computer)	Computer interfaces
	C (Computer program language)	✓
Overall	**TP**= 40, **FP**= 24, **FN**= 24, **Pre**= **Re**= **F1**= 0.625	