DE GRUYTER
OPEN

Research Paper

Automatic Classification of Swedish Metadata Using Dewey Decimal Classification: A Comparison of Approaches

Koraljka Golub[1†] Johan Hagelbäck[2], Anders Ardö[3] (emeritus)

[1]Department of Cultural Sciences, Faculty of Arts and Humanities, Linnaeus University, Växjö, Sweden
[2]Department of Computer Science and Media Technology, Faculty of Technology, Linnaeus University, Kalmar, Sweden
[3]Department of Electrical and Information Technology, Lund University, Lund, Sweden

Citation: Golub, Koraljka, Johan Hagelbäck, and Anders Ardö. "Automatic Classification of Swedish Metadata Using Dewey Decimal Classification: A Comparison of Approaches." *Journal of Data and Information Science*, vol.5, no.1, 2020, pp. 18–38.

DOI: 10.2478/jdis-2020-0003

Received: Feb, 4, 2020
Revised: Mar. 20, 2020
Accepted: Mar. 25, 2020

Abstract

Purpose: With more and more digital collections of various information resources becoming available, also increasing is the challenge of assigning subject index terms and classes from quality knowledge organization systems. While the ultimate purpose is to understand the value of automatically produced Dewey Decimal Classification (DDC) classes for Swedish digital collections, the paper aims to evaluate the performance of six machine learning algorithms as well as a string-matching algorithm based on characteristics of DDC.

Design/methodology/approach: State-of-the-art machine learning algorithms require at least 1,000 training examples per class. The complete data set at the time of research involved 143,838 records which had to be reduced to top three hierarchical levels of DDC in order to provide sufficient training data (totaling 802 classes in the training and testing sample, out of 14,413 classes at all levels).

Findings: Evaluation shows that Support Vector Machine with linear kernel outperforms other machine learning algorithms as well as the string-matching algorithm on average; the string-matching algorithm outperforms machine learning for specific classes when characteristics of DDC are most suitable for the task. Word embeddings combined with different types of neural networks (simple linear network, standard neural network, 1D convolutional neural network, and recurrent neural network) produced worse results than Support Vector Machine, but reach close results, with the benefit of a smaller representation size. Impact of features in machine learning shows that using keywords or combining titles and keywords gives better results than using only titles as input. Stemming only marginally improves the results. Removed stop-words reduced accuracy in most cases, while removing less frequent words increased it marginally. The greatest impact is produced by the number of training examples: 81.90% accuracy on the training set is achieved when at least 1,000 records per class are available in the training set, and 66.13% when too few records (often less than

† Corresponding author: Koraljka Golub (E-mail: koraljka.golub@lnu.se).

100 per class) on which to train are available—and these hold only for top 3 hierarchical levels (803 instead of 14,413 classes).

Research limitations: Having to reduce the number of hierarchical levels to top three levels of DDC because of the lack of training data for all classes, skews the results so that they work in experimental conditions but barely for end users in operational retrieval systems.

Practical implications: In conclusion, for operative information retrieval systems applying purely automatic DDC does not work, either using machine learning (because of the lack of training data for the large number of DDC classes) or using string-matching algorithm (because DDC characteristics perform well for automatic classification only in a small number of classes). Over time, more training examples may become available, and DDC may be enriched with synonyms in order to enhance accuracy of automatic classification which may also benefit information retrieval performance based on DDC. In order for quality information services to reach the objective of highest possible precision and recall, automatic classification should never be implemented on its own; instead, machine-aided indexing that combines the efficiency of automatic suggestions with quality of human decisions at the final stage should be the way for the future.

Originality/value: The study explored machine learning on a large classification system of over 14,000 classes which is used in operational information retrieval systems. Due to lack of sufficient training data across the entire set of classes, an approach complementing machine learning, that of string matching, was applied. This combination should be explored further since it provides the potential for real-life applications with large target classification systems.

Keywords LIBRIS; Dewey Decimal Classification; Automatic classification; Machine learning; Support Vector Machine; Multinomial Naïve Bayes; Simple linear network; Standard neural network; 1D convolutional neural network; Recurrent neural network; Word embeddings; String matching

1 Introduction

Subject searching (searching by topic or theme) is the most common and at the same time the most challenging type of searching in library catalogs and related quality information services, compared to, for example, a known-title or a known-author search. Subject index terms taken from standardized knowledge organization systems (KOS), like classification systems and subject headings systems, provide numerous benefits compared to free-text indexing of commercial search engines: consistency through uniformity in term format and the assignment of terms, provision of semantic relationships among terms, support of browsing by provision of consistent and clear hierarchies (for a detailed overview, see, for example, Lancaster, 2003). However, controlled subject index terms are expensive to produce manually and there is a huge challenge facing library catalogs and digital collections of various types: how to provide high quality subject metadata for increasing numbers

of digital information at reasonable costs. (Semi)-automatic subject classification and indexing represent some potential solutions to retain the established objectives of library information systems.

With the ultimate purpose of establishing the value of automatically produced classes for Swedish digital collections, the paper aims to develop and evaluate automatic subject classification for Swedish textual resources from the Swedish union catalogue (LIBRIS[①]). Based on a data set of 143,756 catalogue records, six machine learning algorithms and one string-matching algorithm were chosen and evaluated.

The paper is structured as follows: Section 2 sets out the rationale for the study and discusses challenges surrounding automatic subject indexing and classification when applied in quality information systems; in Section 3 the data collection, two algorithms and evaluation are described; Section 4 reports on major outcomes; in Section 5 a brief discussion of the impact of the results and implications for operational systems is given.

2 Background

Subject searching is a common type of searching in library catalogs (Hunter, 1991; Villén-Rueda et al., 2003) and discovery services (Meadow & Meadow, 2012). However, in comparison to known-item searching (finding an information object whose title, author etc. is known beforehand) searching by subject is much more challenging. This is due to difficulties such as ambiguities of the natural language and poor query formulation, which can be due to lack of knowledge of the subject matter at hand and of information searching. In order to alleviate these problems, library catalogues and related information retrieval systems (could) employ:

1. Hierarchical browsing of classification schemes and other controlled vocabularies with hierarchical structures, which help further the user's understanding of the information need and provide support to formulate the query more accurately;
2. Controlled subject terms from vocabularies such as subject headings systems, thesauri and classification systems, to help the user to, for example, choose a more specific concept to increase precision, a broader concept or related concepts to increase recall, to disambiguate homonyms, or to find which term is best used to name a concept.

[①] http://libris.kb.se

The Swedish National Library recently adopted the Dewey Decimal Classification (DDC) to be used as a new national classification system (Svanberg, 2013), replacing SAB (Klassifikationssystem för svenska bibliotek) used earlier since 1921. However, cataloguing with a major classification system, such as DDC, is resource intensive. While fully automatic solutions are not currently feasible, semi-automated solutions can offer considerable benefit, both in assisting the workflow of expert cataloguers and in encouraging wider use of controlled indexing by authors and other users. Although some software vendors and experimental researchers claim to entirely replace manual indexing in certain subject areas (Rotiblat et al., 2010), others recognize the need for both manual (human) and computer-assisted indexing, each with its (dis)advantages (Anderson & Perez-Carballo, 2001; Svarre & Lykke, 2014). Reported examples of operational information systems include BASE by Bielefeld University Library[2].

NASA's machine-aided indexing which was shown to increase production and improve indexing quality (Silvester, 1997); and the Medical Text Indexer at the US National Library of Medicine, which by 2008 was consulted by indexers in about 40% of indexing throughput (Ruiz, Aronson, & Hlava, 2008).

However, hard evidence on the success of automatic indexing tools in operating information environments, is scarce; research is usually conducted in laboratory conditions, excluding the complexities of real-life systems and situations. It is difficult to compare evaluation results on different systems with widely different datasets. The practical value of automatic indexing tools is largely unknown due to problematic evaluation approaches. Having reviewed a large number of automatic indexing studies, Lancaster concluded that the research comparing automatic versus manual indexing is "seriously flawed" (Lancaster, 2003). One common evaluation approach is testing the quality of retrieval based on the assigned index terms. But retrieval testing is fraught with problems; the results depend on many factors, so retrieval testing cannot isolate the quality of the index terms. Another approach is to measure indexing quality directly. One method of doing so is to compare automatically assigned metadata terms against existing human-assigned terms or classes of the document collection used (as a "gold standard"), but this method also has problems. When indexing, people make errors, such as related to exhaustivity (too many or too few subjects assigned) or specificity (usually because the assigned subject is not the most specific available); they may omit important subjects, or assign an obviously incorrect subject. In addition, it has been reported that different people, whether users or professional subject indexers, assign different subjects to

[2] https://www.base-search.net/about/en/

the same document. For a more detailed discussion on these challenges and proposed approach, see Golub et al. (2016).

Research related to automated subject indexing or classification can be divided between three major areas: document clustering, text categorization and document classification (Golub, 2006; Golub, 2017). In document clustering, both clusters (classes) into which documents are classified and, to a limited degree, relationships between them, are produced automatically. Labelling the clusters is a major research problem, with relationships between them, such as those of equivalence, related-term and hierarchical relationships, being even more difficult to automatically derive (Svenonius, 2000). In addition, "Automatically-derived structures often result in heterogeneous criteria for class membership and can be difficult to understand" (Chen & Dumais, 2000). Also, cluster labels, and the relationships between them, change as new documents are added to the collection; unstable class names and relationships are user-unfriendly in information retrieval systems, especially when used for subject browsing. Related to this is keyword indexing whereby topics of a document are identified and represented by words taken from the document itself (also referred to as derived indexing).

Text categorization (machine learning) is often employed for automatic classification of free text. Here characteristics of subject classes, into which documents are to be classified, are learnt from documents with manually assigned classes (a training set). However, the problem of inadequate training sets for the varied and non-uniform hierarchies of the DDC has been recognized. Wang (2009) argues that DDC's deep and detailed hierarchies can lead to data sparseness and thus skewed distribution in supervised machine learning approaches. Lösch et al. (2011) classified scientific documents to the first three levels of DDC from the Bielefeld Academic Search Engine. They found an "asymmetric distribution of documents across the hierarchical structure of the DDC taxonomy and issues of data sparseness", leading to a lack of interoperability that was problematic.

In the document classification approach, string matching is conducted between a controlled vocabulary and the text of documents to be classified (Golub, 2006; Golub, 2017). A major advantage of this approach is that it does not require training documents, while still maintaining a pre-defined structure of the controlled vocabulary at hand. If using a well-developed classification scheme, it will also be suitable for subject browsing in information retrieval systems. Apart from improved information retrieval, another motivation to apply controlled vocabularies in automated classification is to re-use the intellectual effort that has gone into creating such a controlled vocabulary. It can be employed with vocabularies containing uneven hierarchies or sparse distribution across a given collection. It lends itself

to a recommender system implementation since the structure of a prominent classification scheme, such as the DDC, will be familiar to trained human indexers.

Automatic document classification based on DDC remains challenging. In early work, OCLC reported on experiments in the Scorpion project to automatically classify DDC's own concept definitions with DDC (Thompson, Shafer, & Vizine-Goetz, 1997). The matching was based on captions. In more recent work, relative index terms from DDC were also incorporated (Khoo et al., 2015); the aim was to investigate automatic generation of DDC subject metadata from English language digital libraries in the UK and USA. The algorithm approximates the practice of a human cataloguer, first identifying candidate DDC hierarchies via the relative index table and then selecting the most appropriate hierarchical context for the main subject. Using a measure called mean reciprocal rank, calculated as 1 divided by the ranked position of the first relevant result, they achieved 0.7 mean reciprocal rank for top 2 levels of DDC and 0.5 for top 3 levels. They considered the results competitive and promising for a recommender system. Golub (2007) and Golub et al. (2007) use a different controlled vocabulary and also report on competitive results.

3 Methodology

3.1 Dewey Decimal Classification (DDC)

The DDC was named after its conceiver Melvil Dewey; its first edition was published in 1876. Today the DDC is the most widely used classification system in the world: it has been translated to over 30 languages and is used by libraries in more than 130 countries.

The DDC covers the entire world of knowledge. Basic classes are organized by disciplines or fields of study. At the top level there are 10 main classes each of which is further divided into 10 divisions; each division is further subdivided into 10 sections. As a result, the DDC is hierarchical, and well serves purposes of hierarchical browsing. Each class is represented using a unique combination of Arabic numerals which are the same in all languages, providing the potential for cross lingual integrated search services.

The first digit in the class number represents the main class, the second digit indicates the division, and the third digit the section. For example: 500 stands for sciences, 530 for physics, 532 for fluid mechanics. The third digit in a class number is followed by a decimal point used as a psychological pause since after that the division by 10 continues to a number of other more specific degrees of classification, as needed.

The DDC research permit, the Swedish language version, edition 23, was obtained by the research team from OCLC in 2017. The file received was in MARCXML format[3] comprising over 128 MB. MARCXML is an XML Schema based on MARC (MAchine Readable Cataloguing) format for bibliographic data, derived from the ISO 2709 standard titled "Information and documentation—Format for information exchange" used to exchange electronic records between libraries. For ease of application, relevant data were extracted and re-structured into a MySQL database. The data chosen were the following:

- Class number (field 153, subfield a);
- Heading (field 153, subfield j);
- Relative index term (persons 700, corporates 710, meetings 711, uniform title 730, chronological 748, topical 750, geographic 751; with subfields);
- Notes for disambiguation: class elsewhere and see references (253 with subfields);
- Scope notes on usage for further disambiguation (680 with subfields); and,
- Notes to classes that are not related but mistakenly considered to be so (353 with subfields).

The total of 14,413 unique classes was extracted, of which 819 three-digit classes were found in the LIBRIS data collection described below.

3.2 Data collection

The dataset of 143,838 catalogue records was derived from the Swedish National Union Catalogue, LIBRIS, which is the joint catalogue of the Swedish academic and research libraries. It was harvested using the Open Archives Initiative Protocol for Metadata Harvesting (OAIPMH)[4] in the period from 15 April to 21 April 2018. LIBRIS makes its data available in the MARCXML format.

In total 143,838 records with unique id numbers, containing a DDC class (i.e. with MARC field 082[5]), were harvested. The records were parsed and all fields and subfields considered relevant were saved in an SQL-database, one field/subfield per row. Relevant fields were the following ones:

- Control number (MARC field 001), unique record identification number;
- Dewey Decimal Classification number (MARC field 082, subfield a);

[3] https://www.loc.gov/standards/marcxml/
[4] https://www.openarchives.org/OAI/openarchivesprotocol.html
[5] For a list and description of MARC fields, see http://www.loc.gov/marc/bibliographic/.

- Title statement (MARC field 245, subfield a for main title and subfield b for subtitle); and,
- Keywords (a group of MARC fields starting with 6*), where available—85.8% of records had at least one keyword.

The records were formatted into an SQL table containing the total of 1,464,046 rows where each row contained 4 columns: ID, field, subfield, and value. The dataset had to be further pruned and cleaned before it could be used for classification experiments. All text features were stripped from special symbols, with the exception of the &-symbol which was replaced by the Swedish word for *and* (*och*), leaving only letters and numbers in the data. For each record, values for title, subtitle and keywords were concatenated into a list of words separated by whitespace, a process known as tokenization.

In the sample, only records containing DDC classes truncated to a three-digit code, ranging from 001 to 999, were used. Records in lower hierarchical levels were reduced to the third-level class to which they belong. Records with other codes as well as those missing both title and subtitle were excluded from the dataset. Duplicates (records with identical title + subtitle) were also removed. This cleaning phase resulted in a total of 143,838 records spread over 816 classes; or, 121,505 records spread over 802 classes when extracting only records which contained at least one keyword.

From the cleaned LIBRIS data a number of datasets were generated, which are presented in Table 1 below. One difficulty with the LIBRIS data is the extreme imbalance between DDC classes, the problem recognized also in previous research (see section 2). The most frequent class is 839 (other Germanic literatures) with 18,909 records, while 594 classes have less than 100 records (70 of those have only one single record). To see how this class imbalance affects classifiers, we have also generated a dataset containing only classes with at least 1,000 records, called major classes below. The latter resulted in 72,937 records spread over 29 classes, and 60,641 records spread over 29 classes when selecting records with keywords.

Table 1. The different datasets generated from the raw LIBRIS data.

Dataset	ID	records	classes
Titles	T	143,838	816
Titles and keywords	T_KW	121,505	802
Keywords only	KW	121,505	802
Titles, major classes	T_MC	72,937	29
Titles and keywords, major classes	T_KW_MC	60,641	29
Keywords only, major classes	KW_MC	60,641	29

3.3 Machine learning

Machine learning is the science of getting computers to learn, and improve their learning over time in autonomous fashion, by feeding them data. Instead of explicitly programming a computer what to do, the computer learns what to do by observing the data.

To automatically classify a resource, we need to build models that map input features, i.e. title, subtitle and, optionally, keywords, to a DDC class. These models learn from known, already classified, data (the LIBRIS database) and can later be used to automatically classify new resources. This is referred to as a supervised learning problem; both input features and correct classifications are known.

Machine learning algorithms cannot work with text data directly, so the list of words representing each record in the dataset needs to be encoded as a list of integer or floating point values (referred to as vectorization or feature extraction). The most intuitive way to do so is the "bag of words" representation. The "bag" contains all words that occur at least once in the dataset. A record in the dataset is represented as a vector with the number of occurrences for each word in the title, subtitle and, optionally, keywords. Since the number of distinct words is very high, the vector representing a record is typically very sparse (most values are 0). For the dataset with titles and subtitles, the bag contains a total of 130,666 unique words, and for the dataset with titles, subtitles and keywords, the bag comprises 134,790 unique words. Rare words are later removed, as described below.

When counting occurrences of each single word, all information about relationships between words in the data is lost. This is typically solved using n-grams. An n-gram is a sliding window of size n moving over a list of words, at a pace of one word forward in each step. If a 2-gram is applied, combinations of two words are used as input features instead of, or in combination with, single words (unigrams). For example the text "machine learning algorithm" contains unigrams "machine", "learning", "algorithm", and 2-grams "machine learning" and "learning algorithm". Using n-grams drastically increases the size of the bag, but can possibly give better classification performance of models. Using unigrams and 2-grams for the datasets with titles, subtitles and keywords as input increases the size of the bag from 134,790 to 828,122 words/word combinations. We have also evaluated higher n-grams (3-grams and 4-grams) but the results did not improve and the computation time of the algorithms increased dramatically.

However, only counting occurrences is problematic: records with longer inputs (title, subtitle and, optionally, keywords) will have higher average count values than records with shorter inputs, even if they belong to the same DDC class. To get around this problem, the number of occurrences for each word is divided by the

total number of words in the record, referred to as Term Frequency (TF). A further improvement is to downscale weights for words that occur in many records and are therefore less informative than words that occur in only a few records. This is referred to as Inverse Document Frequency (IDF). Typically both of these approaches are used, called TF-IDF conversion.

The preprocessing of the text inputs results in high-dimensional, sparse input vectors of either integer values (counting occurrences only) or floating point values (TF-IDF conversion). Many machine learning algorithms are not suited for this type of input data, leaving only a few options left for our task. Historically, good results for different text classification tasks have been achieved with the Multinomial Naïve Bayes (NB) and Support Vector Machine with linear kernel (SVM) algorithms (Aliwy & Ameer, 2017; Trivedi et al. 2015; Wang, 2009). SVM typically gives better results than NB, but is slower to train.

The problem with the bag-of-words approach is that it cannot model any relationships between words. In natural language, some words are related (mango, apple) while others have very different meaning (apple, car). In word embeddings, a numerical representation for words is learned. Each word is represented as a high-dimensional numerical vector, in our case 128 features. The idea with word embeddings is that words that have similar meaning, such as mango and apple, have vectors that are closer to each other (in n-dimensional space) than words with very different meaning. There are pre-learned standard word embeddings that can be used. In our case we used the built-in word embeddings in the Keras machine learning library. Transforming the dataset using the average of all word embeddings results in a set of dense real-valued vectors suitable for neural network algorithms. We have evaluated four different types of neural networks. A simple linear network (Linear), a standard neural network with one hidden layer (NN), a deep neural network using convolutional layers (CNN) (a simplified explanation is that it uses a sliding window over the inputs to reduce the size of the network) and a recurrent neural network (RNN) (can handle relationships between words by allowing recurrent loops in the network, often used for natural language processing tasks).

In addition, of the 143,838 records, 98.6% had one assigned DDC class and 1.4% had more than one assigned class. Because of this, the choice of machine learning algorithms was to apply those producing single output and the 1.4% of records with more than one assigned class were not included in the sample; this also aligned with classification policies of libraries—it is one class per information resources that is typically assigned.

3.4 String matching

The approach for string matching is from the Scorpion system by OCLC (Thompson et al. 2017), which implements a ranked retrieval database using terms

and headings from DDC. Introducing text from LIBRIS records as query for such a database produces ranked results that present a list of potential DDC classifications.

As the ranked retrieval database system we used Solr version 8.2.0[6] which is based on the Lucene full-text search engine library[7]. Each document in the database consists of two fields: one with just the DDC class and the second with terms representing that DDC class, extracted as explained in section 3.1 and thus including corresponding DDC heading, relative index term and, if available, any notes. The terms field is just a sequence of concatenated terms ranging in size from 1 word to 168 words depending on the DDC class. These records are indexed in the database using default Solr configuration with a Swedish stop word list consisting of 531 words. Stemming was not used. Some example documents are presented below, with DDC class followed by DDC terms:

- 005.435: Virtuellt minne program Minneshantering Program för minneshantering
- 565.38: Decapoda Tiofotade kräftdjur paleozoologi Teuthida Eucarida Lysräkor Lysräkor paleozoology
- 760.1: Filosofi och teori
- 781.7101: Allmänna principer för kristen religiös music
- 917.3: geografi Förenta staterna Geografi och resor gällande Förenta staterna

The database field with terms representing a DDC class is queried (relevance-ranked best match using BM25 ranking (Robertson & Zaragoza, 2009)) with queries constructed from title and keywords fields taken from LIBRIS DDC-records. As a result, we get a ranked list of DDC classes.

4 Results

4.1 Machine learning on Naïve Bayes and Support Vector Machines

Results on machine learning reported in the remainder of the document refer to the top three DDC levels only (due to lack of training examples, as discussed above). Tables 2 and 3 below show classification accuracy (amount of records classified into the correct DDC class divided by the total number of records) of Naïve Bayes (NB) and Support Vector Machines (SVM) algorithms. The columns labeled "Training set" show results when training and evaluating a classifier on all records in the dataset. This gives an indication of how effectively we can map inputs to classes, but does not show the generalization capabilities of the classifiers, i.e. how

[6] https://lucene.apache.org/solr
[7] https://lucene.apache.org/core/

good they are at classifying records they have not seen before. Therefore, we have also trained the classifiers on 95% randomly selected records from the dataset, and used the remaining 5% of the records for evaluation (shown in the columns labeled "Test set").

Table 2. Accuracy of the Multinomial Naïve Bayes classifier on the different datasets.

Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T	83.54%	34.89%	95.82%	34.15%
T_KW	90.01%	55.33%	98.14%	55.45%
KW	75.28%	59.15%	84.95%	58.11%
T_MC	90.83%	54.21%	98.63%	50.51%
T_KW_MC	95.42%	76.52%	99.66%	75.96%
KW_MC	86.94%	77.25%	94.24%	77.09%

Table 3. Accuracy of the Support Vector Machine classifier on the different datasets.

Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T	93.74%	40.91%	99.59%	40.45%
T_KW	97.50%	65.25%	99.90%	66.13%
KW	83.09%	64.02%	92.38%	64.09%
T_MC	93.95%	57.99%	99.62%	57.80%
T_KW_MC	97.89%	80.75%	99.93%	81.37%
KW_MC	90.58%	79.56%	96.30%	80.38%

The best results were achieved when combining titles and keywords as input. Using only titles as input results in considerably worse accuracy than when combining titles and keywords or using only keywords as input, a difference around 22–23 percentage units for the major classes datasets. The results show that keywords have much higher information value than titles.

As expected, SVM has higher accuracy scores than NB on all datasets. This is in line with previous research on bibliographic data (Trivedi et al., 2015). The best result for SVM when using all classes was 99.90% accuracy on the training set and 66.13% on the test set, when using both unigrams and 2-grams. When removing all classes with less than 1,000 records, the accuracy on the test set increased to 81.37%.

The overall lower accuracy scores on the test sets compared to the training set even when removing classes with few records may be affected by a phenomenon called *indexing consistency*. A number of studies have shown that humans assigning classes or keywords to bibliographic records often do this in an inconsistent manner, both compared to themselves (intra-indexing consistency) and compared to other humans (inter-indexer consistency) (Leininger, 2000). Since the classifiers learn

from LIBRIS data categorized by humans, this inconsistency may affect their generalization capabilities leading to difficulties when classifying records they have not seen before. The extreme class imbalance also affects the generalization capabilities negatively.

Combining both unigrams and 2-grams only marginally improved the results on the test sets. The highest accuracy was achieved when using SVM and both titles and keywords as input and only major classes. For this dataset the accuracy only increased with 0.62 percentage units when combining unigrams and 2-grams. For NB, the accuracy scores were for most datasets lower than when using unigrams only. We have done some initial testing with higher n-grams (3-grams and 4-grams) but with slightly worse results and significant increases in training time for the algorithms. This needs however to be explored in more detail.

To summarize, using keywords or combining titles and keywords gives much better results than using only titles as input. SVM outperforms NB on all datasets, and the class imbalance where many DDC classes only have few records greatly affects classification performance. Combining unigrams and 2-grams in the input data only marginally improved classification accuracy but leads to much longer training times.

4.2 Stop words, stemming and less frequent words

One approach to improve classification accuracy is to pre-process the input data before feeding it to a machine learning algorithm. We have used three different pre-processing techniques: removing stop words, removing less frequent words and stemming. We have also tested combinations of these techniques.

Stemming is the process of reducing words to their base or root form. There are several stemming algorithms that can be used, for example lemmatization or rule-based suffix-stripping algorithms. To investigate how stemming affects accuracy, we have generated two new datasets where the Snowball stemming algorithm for Swedish was used on titles and subtitles[8]. No stemming was used on keywords as they are typically already in base form. We confirmed that this was a good choice by running some tests which showed that accuracy decreased when using stemming on keywords.

We have used a pre-defined list of Swedish stop words from the ranks.nl website[9] to remove stop words from the titles and subtitles.

When converting text data to bag-of-words and TF-IDF conversion, frequency scores of how often each word appears in the whole dataset is calculated. By setting

[8] http://snowball.tartarus.org/algorithms/swedish/stemmer.html
[9] https://www.ranks.nl/stopwords/swedish

a minimum threshold value, less frequent words are removed from the bag-of-words. We have used a threshold value of 0.00001, effectively reducing the bag-of-words size to one third.

Tables 4 and 5 below show results for the six algorithms when removing stop words (_sw) and less frequent words (_rem) in combination with stemming (_stm). Removing stop words lead to a small decrease in accuracy for SVM (81.37% to 81.24%) and a small increase for NB (75.96% to 76.62%), using 2-grams. When using stemming, a small increase in accuracy was obtained for both NB (75.96% to 76.36%) and SVM (81.37% to 81.80%), using 2-grams. Removing less frequent words lead to an accuracy increase for both NB (75.96% to 78.21%) and SVM (81.37% to 81.83%). The best result for NB was obtained when combining all three approaches leading to an accuracy of 78.90%. The best results for SVM was when combining stemming and removal of less frequent words, leading to an accuracy of 82.20%. This is slightly better than using no pre-processing which resulted in an accuracy of 81.37%.

Table 4. Accuracy of the Naïve Bayes classifier using different pre-processing.

Naïve Bayes				
Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T_KW_MC	95.42%	76.52%	99.66%	75.96%
T_KW_MC_rem	90.17%	76.79%	93.25%	78.21%
T_KW_MC_stm	94.32%	76.36%	99.59%	76.36%
T_KW_MC_stm_rem	89.62%	76.26%	92.95%	78.27%
T_KW_MC_sw	95.50%	76.46%	99.64%	76.62%
T_KW_MC_sw_rem	90.28%	77.09%	92.33%	78.60%
T_KW_MC_sw_stm	94.49%	76.59%	99.53%	76.95%
T_KW_MC_sw_stm_rem	89.79%	76.36%	91.96%	78.90%

Table 5. Accuracy of the Supper Vector Machine classifier using different pre-processing.

Support Vector Machine				
Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T_KW_MC	97.89%	80.75%	99.93%	81.37%
T_KW_MC_rem	92.51%	80.94%	95.02%	81.83%
T_KW_MC_stm	97.21%	81.07%	99.91%	81.80%
T_KW_MC_stm_rem	92.18%	81.34%	94.89%	82.20%
T_KW_MC_sw	95.44%	80.98%	98.48%	81.24%
T_KW_MC_sw_rem	92.46%	81.04%	94.30%	82.13%
T_KW_MC_sw_stm	94.87%	81.40%	98.72%	81.24%
T_KW_MC_sw_stm_rem	92.17%	81.54%	94.16%	81.90%

4.3 Word embeddings

Tables 6 and 7 below show accuracy metrics for word embeddings combined with four different types of neural networks: Simple linear network (Linear), Standard neural network (NN), 1D convolutional neural network (CNN) and Recurrent neural network (RNN).

Table 6. Accuracy of NN and CNN classifiers using word embeddings.

Dataset	NN		CNN	
	Training set	Test set	Training set	Test set
T_KW_MC	96.19%	79.40%	95.33%	79.92%
KW_MC	90.54%	78.23%	90.39%	79.15%
T_KW_MC_stm	95.92%	79.57%	94.60%	80.38%

Table 7. Accuracy of Linear and RNN classifiers using word embeddings.

Dataset	Linear		RNN	
	Training set	Test set	Training set	Test set
T_KW_MC	97.17%	79.99%	92.76%	78.70%
KW_MC	91.30%	78.41%	88.03%	78.74%
T_KW_MC_stm	96.90%	80.81%	92.38%	79.16%

The results show that all the four algorithms perform worse than SVM, but very close—in best example, Simple linear network yields 80.8% compared to 82.2% of best SVM, for main classes and with stemming applied. Like in the case of SVM and NB, stemming slightly improves accuracy. An advantage of word embeddings is having a smaller representation size (then the stored data takes less space); and since differences are not large, these approaches may work sufficiently well when working with large data sets. We have not tried any of the pre-processing techniques (stop words removal, stemming or removal of less frequent words) in combination with word embeddings.

4.4 Machine learning and training examples

A problem when using machine learning algorithms on the dataset is the huge number of possible classes (802 when using three digits). Considering that DDC is hierarchical, one approach to increase the performance of the machine learning models could be a hierarchical set of classifiers. A hierarchical classifier first determines values of most specific classes (lowest in the hierarchy) and these outputs are then combined for classification results at a higher level; the process is iterated till top levels. Using two digits instead of three (99 classes instead of 802) increased the accuracy when using all examples from 58.10% to 73.30%, see Tables 8 and 9 below. This indicates that a hierarchical approach could work, but it

needs more investigation as models must be trained and evaluated for each of the ten subsets.

Table 8. Accuracy of the Naïve Bayes classifier when using two digits.

Naïve Bayes				
Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T_KW_stm_2d	87.40%	65.64%	93.18%	67.79%
T_KW_2d	88.26%	64.78%	93.55%	66.92%
KW_2d	78.36%	68.12%	82.53%	67.94%

Table 9. Accuracy of the Support Vector Machine classifier when using two digits.

Support Vector Machine				
Dataset	Accuracy, unigrams		Accuracy, unigrams + 2-grams	
	Training set	Test set	Training set	Test set
T_KW_stm_2d	90.60%	72.68%	96.23%	73.32%
T_KW_2d	91.21%	72.14%	95.48%	73.24%
KW_2d	81.75%	71.86%	86.18%	71.96%

4.5 String matching results

First we generated the query by using title and keyword fields; we also tried using a query generated from title, keyword and summary fields in LIBRIS records, but the results were almost identical since only 11,000 records had any summary.

For each query (LIBRIS record) against the database (DDC terms) we analyzed both the best hit and the top three ranked hits for each query. When looking at top-ranked 1000 DDC classes, the results were below 50% correct: in 11.8% of cases were top classes identified accurately and in 32.7% cases was the accurate class found among top 3 results. Reducing the number of classes to top-ranked 100, the best we could achieve was 43.7% correct among top 3 hits; top ranking accuracy increased to 15.4%.

4.6 Evaluation based on specific classes

Looking at accuracy of specific classes in all the classes (802), machine learning algorithms in general do worst on class 3xx (social sciences, sociology & anthropology). It often happens that other classes are misclassified as belonging to this class, and what should be 3xx documents, often get misclassified as other classes. Most misclassifications take place between 3xx and 6xx (technology). In the major classes dataset, most problems appear with fiction, which is in a way expected since topics there are mostly about genre, language and country rather than

actual themes. In particular, 823 (English fiction) is often misclassified as 839 (other Germanic literatures) and 813 (American fiction in English) get misclassified as 823 and 839. In addition, the other worst example is class 306 (culture and institutions) which is often misclassified as 305 (groups of people)—class most problematic when looking at all classes in the dataset.

In the lack of training documents, string matching may complement machine learning (Wartena & Franke-Maier, 2018), provided that the terms denoting the class at hand are appropriate for the task. Looking at a number of individual classes with high accuracy using the string-matching approach, this is best achieved when terms used to denote a class are unambiguous. Top five performing classes are listed below together with their terms and accuracy scores:

- 324.623: kvinnor kvinnlig rösträtt rösträtt kvinnlig rösträtt; 100% accuracy. A close examination shows that women's voting rights in all the records were mentioned in either title of keywords, which lead to total accuracy of this individual class.
- 597.3: Havsängelartade hajar Carcharhiniformes Notidanoidei zoologi Såghajartade hajar Hajfiskar Hajar Wobbegongartade hajar Gråhajartade hajar Squatiniformes Heterodontiformes Chondrichthyes Broskfiskar Pristiophoriformes Jättehajar Kamtandhajartade hajar Tjurhuvudhajar Selachii Orectolobiformes Hexanchiformes Elasmobranchii Lamniformes Håbrandsartade hajar Selachii hajfiskar Holocephali helhuvudfiskar Sarcopterygii lobfeniga fiskar; 100% accuracy. Although there are many highly specific terms, only two of them led to complete accuracy for this class: sharks (hajar) and fish (fiskar).
- 616.24: medicin KOL Lungor Pulmonell hypertension Pulmonella sjukdomar Kroniskt obstruktiv lungsjukdom Lungsjukdomar Obstruktiv lungsjukdom Lunghypertoni Lungsjukdomar; 97.7% accuracy. This class is denoted by highly unique terms referring to chronic obstructive pulmonary disease, leading to high accuracy.
- 745.61: Textning Skönskrift Konstnärlig textning Kalligrafi Alfabet konsthantverk Kalligrafi; 91.6% accuracy. This is another example with highly specific and unambiguous terms, leading to good automatic classification results.
- 947.0841: Lenin Vladimir Ryssland Ryska revolutionen 1917 rysk historia Kerenskij Aleksandr 1917–1924 Perioden under revolutionerna Aleksandr Kerenskij Vladimir Lenin 1917–1924; 90.3% accuracy. Words denoting Russian revolution and Lenin are another example of specific terms that rather uniquely represent the unambiguous concepts.

In contrast, examples of classes with 0 accuracy are:

- 005.369: Specific programs. This class will have works on programs like "Word for Windows" but will not be classified rightly since the term used to denote the class is generic and works on specific programs will use titles with words denoting the specific program.
- 510.71: Matematik utbildning. This an example of very short term list which is also rather general; titles are usually more specific, leading often to classification failures.
- 782.42164092: Västerländsk populärmusik Populärmusik biografier västerländska sånger Populärmusik sånger västerländska biografier. In this class misclassifications are due to usage of "1900" or "2000" in the record which then got misclassified as another class that used that year. Or a typical problem with metaphors used in arts and humanities—a work titled "En ung naken kvinna: mitt grekiska drama" (in English: "A young naked woman: My Greek drama") is misclassified as class 480 for classical Greek. Sometimes the problem with strings is impossible to address, such as in "Adjö det ljuva livet" (in English: "Goodbye the sweet life") which is misclassified as 236.2 for life after death since "livet" is listed here and it would not have make any sense to list it as a synonym for the class at hand.
- 839.736: Svenska romaner 1809–1909 Svenska romaner och noveller 1800-talet Svenska romaner och noveller 1809–1909. This class is a typical problem of describing fiction with terms denoting genre and periods which will not normally be present in the titles of the works described, leading to many works not having any class assigned or works being misclassified.

5 Concluding remarks

State-of-the-art machine learning algorithms require at least 1,000 training examples per class. The complete data set we were able to get access to at the time of research involved 143,838 records for 14,413 classes, meaning that DDC had to be reduced to top three hierarchical levels in order to provide sufficient training data, totaling only 802 classes. Achieving high accuracy of 81% reported when using SVM has proven to be dependent on the availability of a good amount of training data, i.e. at least 1,000 records per class. The lack of training data for a large number of classes is even more severe when looking at more specific classes beyond the top three levels; here out of 14,413 available DDC classes, only about 6% of classes had a sufficient number of training examples.

Previous research has demonstrated value in string matching when applying equivalence, hierarchical and related relationships between terms, built in knowledge

organization systems such as classification schemes. This may be a good complement for machine learning approaches, especially when lacking training data. Our results show that this may work for specific classes, while in general machine learning outperforms string matching. Fiction seems a hard problem to address in both approaches, not surprisingly so, due its language which is on purpose often vague and metaphorical.

In all, it seems that automatic approaches could be approved in two main ways: 1) increasing the number of training data for machine learning algorithms, 2) enriching DDC with synonyms to increase performance of string-matching algorithms (e.g., with Swedish thesaurus called Swesaurus®)—the latter would additionally serve another purpose, that of increasing the number of subject access points for end users, which would make classification systems like DDC more end-user friendly and help with term disambiguation and query re-formulation, often lacking in library catalogs. For higher levels, also terms belonging to the subclasses could be taken. Increasing the number of training data may be expected over time; however, having 1,000 records per each of over 14,000 classes may be hard to expect, due to the fact that distribution of materials over all classes is rather skewed (as also seen in previous research, see above). Automatic translation using other languages may also create additional noise and lead to smaller accuracy.

Purely automatic approaches for DDC creation cannot be applied in operative systems. On one hand, because performance is not good enough, and on the other, because evaluation is hard to estimate due to low indexing consistency when applying large classification systems like DDC (with many options to choose from): cannot be used as "the gold standard": the classes assigned by algorithms (but not human-assigned) might be wrong or might be correct but omitted during human indexing by mistake. A more comprehensive approach to 'gold standard' production is needed (Golub et al., 2016). As a result, machine-aided indexing would be the best approach to implement in operative systems, similar to the one used by the National Library of Medicine in the USA (see above).

In conclusion, for operative information retrieval systems applying purely automatic DDC does not work, either using machine learning (because of the lack of training data for the large number of DDC classes) or using string-matching algorithm (because DDC characteristics perform well for automatic classification only in a small number of classes). Over time, more training examples may become available, and DDC may be enriched with synonyms in order to enhance accuracy of automatic classification which may also benefit information retrieval performance based on DDC. In order for quality information services to reach the objective of highest possible precision and recall, automatic classification should never

® https://spraakbanken.gu.se/resource/swesaurus

be implemented on its own; instead, machine-aided indexing that combines the efficiency of automatic suggestions with quality of human decisions at the final stage should be the way for the future.

Acknowledgments

Thanks are due to OCLC which provided the project with electronic DDC, Swedish version. We are very grateful to Rebecca Green and Sandi Jones for all their advice on how to best process and use the electronic DDC files. Many thanks also to the National Library of Sweden who provided all the training and testing data, especially Harriet Aagaard. Special thanks to anonymous reviewers whose detailed comments significantly helped to improve the paper.

Author contributions

Koraljka Golub (koraljka.golub@lnu.se) conducted the major part of the literature review, coordinated machine learning and string matching analyses and processes, and wrote the majority of the paper. Johan Hagelbäck (johan.hagelback@lnu.se) implemented the machine learning approaches, contributed to the paper with parts related to machine learning and reviewed the paper. Anders Ardö (anders.ardo@gmail.com) prepared the database, implemented the string-matching algorithm, contributed to the paper with parts related to string matching and reviewed the paper.

References

Aliwy, A.H., & Ameer, E.H.A. (2017). Comparative study of five text classification algorithms with their improvements. International Journal of Applied Engineering Research, 12(14), 4309–4319.

Anderson, J., & Perez-Carballo, J. (2001). The nature of indexing: How humans and machines analyze messages and texts for retrieval. Part II: Machine indexing, and the allocation of human versus machine effort. Information Processing and Management 37(2), 255–277.

Chen, H., & Dumais, S. (2000). Bringing order to the web: Automatically categorizing search results. In Proceedings of the ACM International Conference on Human Factors in Computing Systems, Den Haag, 145–152.

Golub, K. (2006). Automated subject classification of textual web documents. Journal of Documentation, 62(3), 350–371.

Golub, K. (2007). Automated subject classification of textual documents in the context of web-based hierarchical browsing: PhD thesis. Lund: Department of Electrical and Information Technology, Lund University.

Golub, K. (2017). Automatic subject indexing of text. In ISKO Encyclopedia of Knowledge Organization, 2017. http://www.isko.org/cyclo/automatic.

Golub, K., Soergel D., Buchanan, G., Tudhope, D., Lykke, M., & Hiom, D. (2016). A framework for evaluating automatic indexing or classification in the context of retrieval. Journal of the Association for Information Science and Technology, 67(1), 3–16.

Golub, K., Hamon, T., & Ardö, A. (2007). Automated classification of textual documents based on a controlled vocabulary in engineering. Knowledge Organization, 34(4), 247–263.

Hunter, R.N. (1991). Successes and failures of patrons searching the online catalog at a large academic library: A transaction log analysis. RQ, 30(3), 395–402.

Khoo, M. et al. (2015). Augmenting Dublin Core digital library metadata with Dewey Decimal Classification. Journal of Documentation, 71(5), 976–998.

Lancaster, F.W. (2003). Indexing and Abstracting in Theory and Practice. Facet: London.

Leininger, K. (2000). Interindexer consistency in PsycINFO. Journal of Librarianship and Information Science, 32(1), 4–8.

Lösch, M., Waltinger, U., Hortsmann, W., & Mehler, A. (2011). Building a DDC-annotated corpus from OAI metadata. Journal of Digital Information, 12(2).

Meadow, K., & Meadow, J. (2012). Search query quality and web-scale discovery: A qualitative and quantitative analysis. College & Undergraduate Libraries 19(2–4), 163–175.

Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance model: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333–389.

Roitblat, H.L., Kershaw, A., & Oot, P. (2010). Document categorization in legal electronic discovery: Computer classification vs. manual review. Journal of the American Society for Information Science and Technology, 61(1), 70–80.

Ruiz, M.E., Aronson, A.R., & Hlava, M. (2008). Adoption and evaluation issues of automatic and computer aided indexing systems. In Proceedings of the American Society for Information Science and Technology, 45(1), 1–4.

Silvester, J.P. (1997). Computer supported indexing: A history and evaluation of NASA's MAI system. In Encyclopedia of Library and Information Services, 61(24), 76–90.

Svanberg, M. (2013). Slutrapport: Dewey-Projektet. http://www.kb.se/Dokument/Deweyprojektet%20slutrapport%2020130614.pdf

Svarre, T.J., & Lykke, M. (2014). Simulated work tasks: The case of professional users. In Proceedings of the 5th Information Interaction in Context Symposium, 215–218.

Svenonius, E. (2000). The Intellectual Foundation of Information Organization. Cambridge, MIT Press.

Thompson, R., Shafer, K., & Vizine-Goetz, D. (1997). Evaluating Dewey concepts as a knowledge base for automatic subject assignment. In Proceedings of the Second ACM Int. Conf. on Digital libraries (DL '97), 37–46.

Trivedi, M., Sharma, S., Soni, N., & Nair, S. (2015). Comparison of text classification algorithms. International Journal of Engineering Research & Technology, 4(2).

Villén-Rueda, L., Senso, J.A., & De Moya-Anegón, F. (2007). The use of OPAC in a large academic library: A transactional log analysis study of subject searching. The Journal of Academic Librarianship, 33(3), 327–337.

Wang, J. (2009). An extensive study on automated Dewey Decimal Classification. Journal of the American Society for Information Science and Technology, 60(11), 2269–2286.

Wartena, C., & Franke-Maier, M. (2018). A hybrid approach to assignment of Library of Congress Subject Headings. Archives of Data Science, 1(4). https://publikationen.bibliothek.kit.edu/1000105121