LOHAI: Providing a baseline for KOS based automatic indexing

Kai Eckert

University of Mannheim
University Library
Mannheim, Germany
eckert@bib.uni-mannheim.de

Abstract. Automatic KOS based indexing – i.e. indexing based on a restricted, controlled vocabulary, a thesaurus or a classification – can play an important role to close the gap between the intellectually, high quality indexed publications and the mass of unindexed publications. Especially for unknown, heterogeneous publications, like web publications, simple processes that do not rely on manually created training data are needed. With this contribution, we propose a straight-forward linguistic indexer, that can be used as a basis for own developments and for experiments and analyses to explore own documents and KOSs; it uses state-of-the-art information retrieval techniques and hence forms a suitable baseline for evaluations. Finally, it is free and open source.

1 Introduction

Intellectual indexing of publications based on a Knowledge Organization System (KOS) – like controlled vocabularies, thesauri or classifications – is still performed to ensure high accuracy in information retrieval. Even with the availability to search the electronic full text, the resolution of synonyms and homonyms by the introduction of a controlled vocabulary – functioning as a common language between creators and searchers of the indexed content – is very important. To close the gap between the subset of publications that are traditionally indexed intellectually – books in libraries, but also selected journal articles in mostly commercial databases – automatic indexing approaches are widely introduced.

The German National Library, for instance, decided, that web publications, while being collected, will not be indexed intellectually, but only by means of automatic processes and search engine technology [9]. A recent workshop focused on automatic indexing, called PETRUS[1] showed that there are mainly two types of approaches: linguistic and statistical approaches. While there are smooth transitions between both, linguistic approaches use techniques from natural language processing (NLP) to process the texts and extract meaningful concepts, while statistical approaches use machine learning techniques to assign concepts based on a manually created training set.

[1] `http://www.d-nb.de/wir/projekte/workshop_petrus.htm`

Based on the discussions and contributions of the workshop, there is currently a preference for statistical approaches, although the reported quality of the results varies. A mentioned problem was the bias that is introduced by the training set. For example, for recent news articles, the indexer learned that the occurrence of "Nuclear power plant" should lead to "Japan" as a concept to be assigned. More general is the observation that the indexing quality relies on the homogenity of the documents to be indexed. If they vary very much regarding content, style or even length, the quality of the indexing result is affected.

In the semantic web, heterogenous contents are the rule rather than the exception. A lot of different KOSs are widely used to describe all kind of resources[2], but for a real semantic interoperability, we have to be able to match the concepts between different KOSs or to quickly assign concepts of a KOS to a new resource. Albeit with inferior quality, such a bridging is needed to connect all kinds of resources, especially in the area of libraries, archives and museums.

With Maui [4], there exists a statistical indexer that incorporates a lot of NLP techniques, but there is to the best of our knowledge no free and open source implementation for a strictly lingustic indexer that can be used without any training data on arbitrary documents. Especially for the evaluation of "real" automatic indexers, such a simple implementation is useful. There are a lot of additional scenarios where this indexer can be used, be it for experimental services or whenever more sophisticated approaches are just not needed. And of course, as a reasonable baseline, more sophisticated approaches have to outperform it in the first place.

In this paper, we present LOHAI[3], a strictly linguistic indexer that incorporates mainly all these techniques that are state-of-the-art in information retrieval. The development of LOHAI is led by the following motivational thoughts:

Simplicity over quality: While every single step could be improved or replaced by a more sophisticated technique that is already developed and published somewhere, we tried to develop everything as simple as possible. Everything should be easy to use, easy to understand and easy to improve if needed.

Knowledge-poor and without any training: To be usable for arbitrary KOS and documents, the indexer can not rely on any *additional* knowledge sources, however, of course, the KOS itself can and will be used. The indexer must not employ a training step, as there are many settings where no preindexed documents are available and the creation of a training set would be too cumbersome for the user.

With these prerequisites in mind, we compose the indexer as a pipeline with several components, as illustrated in Figure 1 on page 3.

[2] E.g. by means of SKOS, `http://www.w3.org/2004/02/skos/`.

[3] LOHAI is pronounced like Low-High and means something like LOw HAnging Fruits Automatic Indexer, which gives a brief summary about the development process.

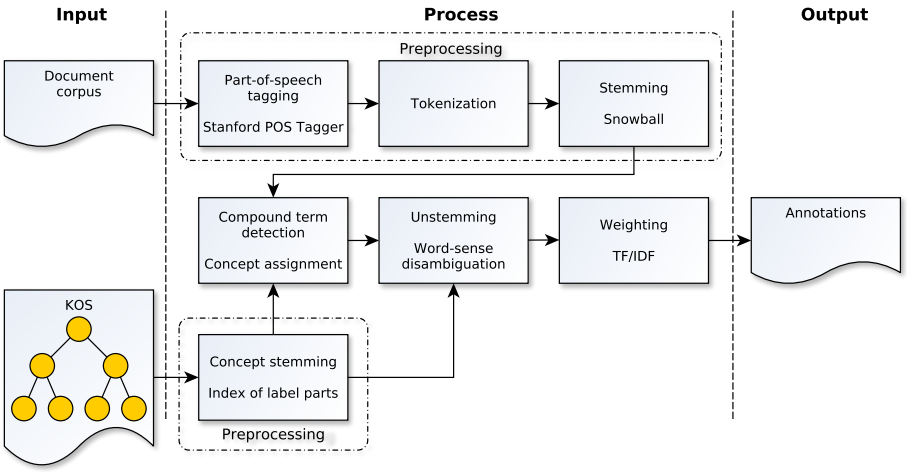

Fig. 1: The indexing pipeline

2 Preprocessing

The preprocessing consists of the several steps:

1. **Part-of-speech tagging:** We use the Stanford Log-linear Part-Of-Speech Tagger[4], as described by Toutanova et al. [10]. Part-of-speech (POS) tagging simply means the identification of nouns, verbs, adjectives and other word-types in a text. To avoid wrong concept assignments like the assignment of the concept "need" (as noun in the sense of requirements) whenever the verb "to need" is used, we only consider nouns (NN[5], NNP, NPS, NNS), adjectives (JJ, JJR, JJS), foreign words (FW) and unknown words (untagged).
2. **Tokenization:** The tokenization splits the text into single terms. The tokenization is performed together with the POS tagging. The result is a list of terms that are further investigated for proper concept assignments. In the tokenization step, there is also a cleaning of the terms included, where everything is truncated that is not a letter, a hyphen or a space. Note, that numbers are truncated, too, as they usually contain no meaning and are generally highly ambiguous. In some domains, this would not be desired, consider for example history or chemistry.
3. **Stemming:** Finally, the single terms are stemmed, i.e. they are reduced to their stem. That way, same terms can be matched, even, if they use different grammatical forms, like "banks" and "bank". We use the English (Porter2) stemming algorithm for the Snowball stemmer [6].
4. **KOS preparation:** This is only performed once per KOS. All concept labels are stemmed by means of the same stemmer that is employed on the

[4] `http://nlp.stanford.edu/software/tagger.shtml`
[5] Tag definitions according to the Penn Treebank tag set [8].

document texts. An index is created that maps the single stems to the corresponsing concepts. Additionally an index of stemmed label parts is created that is used for the identification of compound terms – for instance, "insurance market" would be stemmed to "insur market", mapping to the corresponding concept, additionally, both stem parts are indexed and mapped to the stemmed compound term.

The preprocessing uses only freely available standard approaches. The POS tagging and the stemming are language dependent; both algorithms employed are implemented for various languages, including English and German. We assume that both the KOS and the documents are in the same language and that only one language is used in the document, so that the appropriate implementations can be used. If the KOS is multilingual and the documents use different language, an additional language detection step has to be employed.

After the preprocessing steps, the actual concept assignment and weighting takes place, as described in the next section.

3 Concept assignment with compound term detection

The general assignment strategy is a pure string based matching: If a stem that belongs to a concept in the KOS appears in the stems extracted from the text, the concept is assigned. In this step, we consider every concept a matching concept that contains a label that has the same stem or contains the same stem in the case of an compound term. Under this assumption, we have to deal with three possibilities that consequently would lead to wrong assignments:

1. A stem could belong to several concepts, including compound term concepts, e.g. "insur" that belongs among others to "insurance" and "insurance market".
2. A stem could belong to several concepts that have different labels with the same stem (overstemming), like "nationalism", "nationality" and "nation".
3. A stem could belong to several concepts that have the same labels with the same stem (homonyms), like "bank" (the financial institution) and "bank" (a raised portion of seabed or sloping ground along the edge of a stream, river, or lake).

Approaches to handle the latter two variants are described in the next section, the first variant is dealt with directly in the assignment phase: The basic assumption is that we want to assign the most specific concept, i.e. in the above example, we would like to assign "insurance market", but not "market".

We implement this as follows: Whenever a stem is recognized as a potential part of a compound term, the stem is temporarily stored in a list. When a stem is found that can not be part of a compound, the list is analyzed for contained concepts. In this step, the algorithm simply checks every chain of stems for every starting stem if it corresponds to a compound concept. The algorithm starts with the longest possible chain and stops if a compound is found, thus

avoiding the assignment of additional concepts contained in the compound. With this approach, the algorithm has generally a linear runtime with respect to the words contained in the text. Only the parts that potentially contain compounds have to be further analysed with a runtime of $O(n^2)$ with n denoting the number of words within such a part.

4 Unstemming and word-sense disambiguation

Whenever one or more stems could be assigned to more than one concept, we would like to identify a single concept as the correct one in the given context. This task is generally denoted as word-sense disambiguation (WSD). We use two different approaches for WSD, the first being a specific check that tackles the problem of overstemming mentioned above. If this step is not able to disambiguate the potential concepts, the actual WSD is performed.

Unstemming. Overstemming – the reduction of two different terms to the same stem – leads to ambiguous stems that have to be disambiguated during indexing. Consequently, we first unstem the stem, i.e. we go back to the original, unstemmed form of the term, as found in the text. If the unstemmed term corresponds directly to an unstemmed label of a concept, we assign this concept. If there is only one such concept, we finish the WSD step. Otherwise, we continue with the actual WSD, as described in the following.

KOS based word-sense disambiguation. Word-sense disambiguation is a broad field in the area of natural language processing. Leaving the technical issues of overstemming aside, it generally consists of the task to determine the correct sense of a word that appears in a particular context. The variety of possible senses is often based on some background-knowledge, like a thesaurus or other types of KOS. As Manning and Schuetze [2, pp. 229 f.] pointed out, this can be unsatisfactory from a scientific or philosophical point of view, as the definitions in the background knowledge are often quite arbitrary and possibly not sufficient to describe the actual sense of a word in a given context. However, our goal is not the perfect assignment of a sense to a word, our goal is the assignment of the best fitting concept in the KOS.

WSD approaches can be divided in supervised and unsupervised approaches, additionally in knowledge-rich and knowledge-poor approaches [5]. In our setting, we clearly need an approach that is unsupervised – as it has to work without any previously tagged texts – and knowledge-rich – as we clearly have a KOS at hand and of course want to use it to improve the disambiguation quality.

A supervised, knowledge-rich approach would be the adaptive thesaurus-based disambiguation, as presented by Yarowsky [11], where a Bayes classifier is trained on a large document corpus and thus probabilities for the occurrence of specific words in the context of a specific sense are determined.

Yarowsky [13] also proposed an (almost) unsupervised approach that makes use of two assumptions:

One sense per collocation. We assume that words collocated with the word to be disambiguated are unique for the correct sense and would not be collocated with the word for other senses. This basically is the rationale to use the context of a word – usually a window of words before and after the word in question – for disambiguation.

One sense per discourse. We assume that only one sense for a given word is used throughout a whole document. With this assumption, we can make use of any occurrence of the word in the text and thus get a more stable disambiguation result.

Both assumptions have been examined and verified [1, 12]. However, as Yarowsky's approach is not completely unsupervised – a small set of pretagged senses is needed as seed – we only make use of the two assumptions, but use a much simpler approach: Word-sense disambiguation based on a Jaccard comparison (cf. Ramakrishnan et al. [7]).

For this comparison, we define two sets of words: W as the context of an occurrence of the ambiguous word w, and C as the context of a candidate concept c, respectively. We then compute the Jaccard measure as follows:

$$\text{Jaccard}(W, C) = \frac{|W \cap C|}{|W \cup C|} \tag{1}$$

Based on the assumption "One sense per discourse", we assign each occurrence of w the concept c that was mostly assigned in the document, i.e. got in most cases the highest Jaccard value. If only abstracts are available for indexing, this procedure can be further simplyfied by just assuming the whole abstract as the context for each occurrence of w, which leads to the direct assignment of the concept c with the highest Jaccard value.

As context of an ambiguous word w, we either define a window of 100 words before and after the word or just use the whole document in case of short texts, like abstracts. The context of a concept c is defined as the union of all labels of the concept, its direct child concepts, its parent concepts and the direct children of the parent concepts, i.e. its siblings. Other definitions are of course possible, for example the weighting of words and labels depending on the distance to the word or concept, but for our purpose as part of a simple baseline indexer, our approach is sufficient.

5 Weighting

The last step in the indexing pipeline is the weighting of the assigned concepts. As the baseline indexer so far assigns every concept that can be identified by an occurring word, the weighting of these concepts is vitally important to determine which concepts are important and descriptive for the given text and which concepts are only marginally touched. It is also desirable to give concepts a higher weight when they are not used in the majority of documents, because these concepts usually only denote common terms and are not important for the indexing result.

The common approach for this kind of weighting is *tf-idf*, which is based on the term frequency $tf_{c,d}$ of a term (in our case concept c) in a given document d and on the document frequency df_c of a concept c, i.e. the number of documents, where the concept appears:

$$w(c,d) = tf_{c,d} \cdot \log \frac{D}{df_c} \tag{2}$$

D denotes the total number of documents in the indexed corpus. The last term is called inverse document frequency (idf), as the overall weight becomes smaller the higher df_c is.

6 Results

To show the weaknesses and strengths of LOHAI, we investigate some of the indexing results. For our experiments, we used the German STW Thesaurus for Economics[6]. A concept in the STW consists of preferred and alternative labels, both in English and in German. For example, there is the concept "Migration theory" with alternative labels "Economics of migration" and "Theory of migration".

Figure 2 shows an example abstract that we indexed. LOHAI produces the output as shown in Figure 3. Additionally, we listed the intellectually assigned concepts by a librarian. It can easily be seen that the characteristics of the results are quite different. But if one takes the weighting into account, it can be seen that there are no wrong assignments with a weight above 0.3. Below that threshold, there are especially common terms that form a concept in the thesaurus and that are either not helpful or wrongly assigned, as "Exchange". "Government", for example, seems to be correct, but is rather a coincidence, as it is assigned due to the verb "govern" in the text – an indication for a mistake during the POS tagging. On the other hand, the very abstract concepts that are assigned by the librarian (besides "Theory") are not found by LOHAI, as the terms do not directly appear in the text in some form.

All in all, the results are very promising, even with a relatively simple approach like ours. Most assignments are correct, even if a human indexer would not assign all of them. The indexing quality correlates with the employed weighting, especially assignments with lower rank often contain more common concepts that sometimes are just wrong. A lot of these mistakes could be avoided if the thesaurus would be more precise about homonyms and would provide additional information to disambiguate them, when necessary. The indexer could be further improved, e.g. common concepts should not be assigned, if more specific concepts down the tree are found in the text (Like "Law" and "Contract Law" above). On the other hand, we wanted to keep it simple. Such adaptions and

[6] Standard Thesaurus Wirtschaft, `http://zbw.eu/stw/versions/latest/about.en.html`. The thesaurus is published and maintained by the German National Library of Economics (Deutsche Zentralbibliothek für Wirtschaftswissenschaften, ZBW)

Title	Contractarianism: Wistful Thinking
Authors	Hardin, Russell
Abstract	The contract metaphor in political and moral theory is misguided. It is a poor metaphor both descriptively and normatively, but here I address its normative problems. Normatively, contractarianism is supposed to give justifications for political institutions and for moral rules, just as contracting in the law is supposed to give justification for claims of obligation based on consent or agreement. This metaphorical association fails for several reasons. First, actual contracts generally govern prisoner's dilemma, or exchange, relations; the so-called social contract governs these and more diverse interactions as well. Second, agreement, which is the moral basis of contractarianism, is not right-making per se. Third, a contract in law gives information on what are the interests of the parties; a hypothetical social contract requires such knowledge, it does not reveal it. Hence, much of contemporary contractarian theory is perversely rationalist at its base because it requires prior, rational derivation of interests or other values. Finally, contractarian moral theory has the further disadvantage that, unlike contract in the law, its agreements cannot be connected to relevant motivations to abide by them.
Journal	Constitutional Political Economy, 1 (2) 1990: 35-52

Fig. 2: Example of a document abstract used for annotation

Constitutional economics
Influence of government
Ethics
Theory

(a) Intellectual indexing

Contract Law (1.21)
Contract (0.76)
Social contract (0.64)
Law (0.51)
Politics (0.37)
Prisoner's dilemma (0.34)
Theory (0.32)
Rationalism (0.24)
Association (0.23)
Exchange (0.20)
Knowledge (0.19)
Government (0.16)
Information (0.12)

(b) LOHAI

Fig. 3: Intellectual indexing vs. LOHAI

improvements are easy to implement, if they are needed. A quantitative evaluation of the results by comparing them to a manually created gold standard is still missing, but former experiments [3] with a comparable indexer showed that such results are nevertheless not very meaningful due to the very different characteristics of such an automatic approach and a trained librarian.

7 Conclusion

To the best of our knowledge, there is no free indexer available that does not require any data preparation step or the creation of some training data. With LOHAI, we developed such an indexer by just using the standard approaches in natural language processing and information retrieval for the single steps in the indexing pipeline. Each and every step could be improved by employing new and more sophisticated approaches, but we intentionally restricted ourselves to the well-understood approaches that are state of the art in information retrieval and natural language processing. All in all, the indexer consists of about 500 lines of code in Java, without the POS tagger and the Snowball stemmer. We showed that the indexer performs quite well and – maybe most important – does not behave like a black box, every assignment is easily understandable. We expect that the indexer would even be usable in serious indexing projects. LOHAI is already successfully employed in SEMTINEL[7], a thesaurus evaluation framework, where it is used to quickly process large document sets for a given thesaurus to determine its concept coverage. LOHAI is not a stupid indexer, it is a baseline indexer. It is free, open source and available at `https://github.com/kaiec/LOHAI`.

References

1. Gale, W.A., Church, K.W., Yarowsky, D.: One sense per discourse. In: Proceedings of the workshop on Speech and Natural Language. pp. 233–237. HLT '91, Association for Computational Linguistics, Stroudsburg, PA, USA (1992), http://dx.doi.org/10.3115/1075527.1075579
2. Manning, C.D., Schuetze, H.: Foundations of Statistical Natural Language Processing. MIT Press (1999)
3. Maynard, D., Dasiopoulou, S., Costache, S., Eckert, K., Stuckenschmidt, H., Dzbor, M., Handschuh, S.: D1.2.2.1.3 Benchmarking of annotation tools. Tech. rep., Knowledge Web Project (2007)
4. Medelyan, O.: Human-competitive automatic topic indexing. Ph.D. thesis, University of Waikato (2009)
5. Navigli, R.: Word sense disambiguation: A survey. ACM Comput. Surv. 41(2), 1–69 (2009)
6. Porter, M.: Snowball: A language for stemming algorithms. Published online. (2001), http://www.snowball.tartarus.org/texts/introduction.html

[7] http://www.semtinel.org

7. Ramakrishnan, G., Prithviraj, B., Bhattacharyya, P.: A gloss-centered algorithm for disambiguation. In: SENSEVAL-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text, Barcelona, Spain, July 2004. ACM, New York, NY, USA (2004)
8. Santorini, B.: Part-of-speech tagging guidelines for the Penn Treebank Project (3rd revision). Tech. rep., University of Pennsylvania (1990), http://repository.upenn.edu/cis_reports/570/
9. Schwens, U., Wiechmann, B.: Netzpublikationen in der Deutschen Nationalbibliothek. Dialog mit Bibliotheken 1(1), 10–13 (2009)
10. Toutanova, K., Klein, D., Manning, C., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of HLT-NAACL 2003. pp. pp. 252–259 (2003)
11. Yarowsky, D.: Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In: Proceedings of the 14th conference on Computational linguistics - Volume 2. pp. 454–460. COLING '92, Association for Computational Linguistics, Stroudsburg, PA, USA (1992), http://dx.doi.org/10.3115/992133.992140
12. Yarowsky, D.: One sense per collocation. In: Proceedings of the workshop on Human Language Technology. pp. 266–271. HLT '93, Association for Computational Linguistics, Stroudsburg, PA, USA (1993), http://dx.doi.org/10.3115/1075671.1075731
13. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: Proceedings of the 33rd annual meeting on Association for Computational Linguistics. pp. 189–196. ACL '95, Association for Computational Linguistics, Stroudsburg, PA, USA (1995), http://dx.doi.org/10.3115/981658.981684