Wikipedia and Artificial Intelligence: An Evolving Synergy

Papers from the 2008 AAAI Workshop

Technical Report WS-08-15

AAAI Press

Association for the Advancement of Artificial Intelligence

AAAI Press

445 Burgess Drive
Menlo Park, California 94025

WS-08-15

978157735-3836

Wikipedia and Artificial Intelligence: An Evolving Synergy

Papers from the 2008 AAAI Workshop

Razvan Bunescu, Evgeniy Gabrilovich,
and Rada Mihalcea, Cochairs

Technical Report WS-08-15

AAAI Press

Menlo Park, California

The Association for the Advancement of Artificial Intelligence
445 Burgess Drive
Menlo Park, California 94025 USA

ISBN 978-1-57735-383-6 WS-08-15

Manufactured in the United States of America

Organizing Committee

Program Committee

This AAAI–08 Workshop was held Sunday, July 13, 2008,
in Chicago, Illinois, USA

Preface

Since its inception less than seven years ago, Wikipedia has become one of the largest and fastest growing online sources of encyclopedic knowledge. One of the reasons why Wikipedia is appealing to contributors and users alike is the richness of its embedded structural information: articles are hyperlinked to each other and connected to categories from an ever expanding taxonomy; pervasive language phenomena such as synonymy and polysemy are addressed through redirection and disambiguation pages; entities of the same type are described in a consistent format using infoboxes; related articles are grouped together in series templates.

As a large-scale repository of structured knowledge, Wikipedia has become a valuable resource for a diverse set of artificial intelligence (AI) applications. Major conferences in natural language processing and machine learning have recently witnessed a significant number of approaches that use Wikipedia for tasks ranging from text categorization and clustering to word sense disambiguation, information retrieval, information extraction and question answering. On the other hand, Wikipedia can greatly benefit from numerous algorithms and representation models developed during decades of AI research, as illustrated recently in tasks such as estimating the reliability of authors' contributions, automatic linking of articles, or intelligent matching of Wikipedia tasks with potential contributors.

The goal of this workshop was to foster the research and dissemination of ideas on the mutually beneficial interaction between Wikipedia and AI. The workshop took place on July 13, 2008, in Chicago, immediately preceding the Annual Meeting of the Association for the Advancement of Artificial Intelligence.

This report contains papers accepted for presentation at the workshop. We issued calls for regular papers, short late–breaking papers, and demos. We received an impressive number of submissions, demonstrating the community interest and the timeliness of the "Wikipedia and AI" research theme: 22 regular papers, 10 short papers and 3 demos were submitted to the workshop. After careful review by our program committee, 10 regular papers, 2 short papers and 1 demo were accepted for presentation. Consistent with the original aim of the workshop, the accepted papers address a highly diverse set of problems, in which Wikipedia is seen either as a useful resource, or as a target for algorithms seeking to make it even better. As a rich knowledge source, Wikipedia is shown to benefit applications in information extraction, machine translation, summarization, ontology mining and mapping, and information retrieval. We also learned of interesting applications, most of them using machine learning, that could further enhance the breadth and the quality of Wikipedia, such as predicting the quality of edits, vandalism detection, infobox extraction, crosslingual links creation, and semantic annotation.

We were truly impressed by the high quality of the reviews provided by all the members of the program committee, particularly since deadlines were very tight. All of the committee members provided timely and thoughtful reviews, and the papers that appear have certainly benefited from that expert feedback.

Finally, when we first started planning this workshop, we agreed that having a high quality invited speaker was crucial. We thank Michael Witbrock not only for his talk, but also for the boost of confidence provided by his quick and enthusiastic acceptance.

– Razvan Bunescu, Evgeniy Gabrilovich, and Rada Mihalcea
July 2008

Contents

Invited Speaker

Michael Witbrock (Vice President of Research at Cycorp, USA)

The Fast and the Numerous – Combining Machine and Community Intelligence for Semantic Annotation

Sebastian Blohm, Markus Krötzsch and **Philipp Cimiano**
Institute AIFB, Knowledge Management Research Group
University of Karlsruhe
D-76128 Karlsruhe, Germany
{blohm, kroetzsch, cimiano}@aifb.uni-karlsruhe.de

Abstract

Starting from the observation that certain communities have incentive mechanisms in place to create large amounts of unstructured content, we propose in this paper an original model which we expect to lead to the large number of annotations required to semantically enrich Web content at a large scale. The novelty of our model lies in the combination of two key ingredients: the effort that online communities are making to create content and the capability of machines to detect regular patterns in user annotation to suggest new annotations. Provided that the creation of semantic content is made easy enough and incentives are in place, we can assume that these communities will be willing to provide annotations. However, as human resources are clearly limited, we aim at integrating algorithmic support into our model to bootstrap on existing annotations and learn patterns to be used for suggesting new annotations. As the automatically extracted information needs to be validated, our model presents the extracted knowledge to the user in the form of questions, thus allowing for the validation of the information. In this paper, we describe the requirements on our model, its concrete implementation based on Semantic MediaWiki and an information extraction system and discuss lessons learned from practical experience with real users. These experiences allow us to conclude that our model is a promising approach towards leveraging semantic annotation.

Introduction

With the advent of the so called Web 2.0, a large number of communities with a strong will to provide content have emerged. Essentially, these are the communities behind social tagging and content creation software such as del.icio.us, Flickr, and Wikipedia. Thus, it seems that one way of reaching massive amount of annotated web content is to involve these communities in the endeavour and thus profit from their enthusiasm and effort. This requires in essence two things: semantic annotation functionality seamlessly integrated into the standard software used by the community in order to leverage its usage and, second, an incentive mechanism such that people can immediately profit from the annotations created. This is for example the key idea behind projects such as Semantic MediaWiki (Krötzsch *et al.* 2007) and Bibsonomy (Hotho *et al.* 2006). Direct incentives for creating semantic annotations in a Semantic MediaWiki are for example semantic browsing and querying functionality, but most importantly the fact that queries over structured knowledge can be used to automatically create views on data, e.g. in the form of tables.

However, creating incentives and making annotation easy and intuitive will clearly not be enough to really leverage semantic annotation at a large scale. On the one hand, human resources are limited. In particular, it is well known from Wikipedia and from tagging systems that the number of contributors is relatively small compared to the number of information consumers. On the other hand, we need to use human resources economically and wisely, avoiding that people get bored by annotating the obvious or the same things again and again. This is where standard machine learning techniques which detect regularities in data can help. However, any sort of learning algorithm will produce errors, either because they overgenerate or they overfit the training data. Thus, human verification is still needed. We argue that this verification can be provided by the community behind a certain project if the feedback is properly integrated into the tools they use anyway. This opens the possibility to turn information consumers into "passive annotators" which, in spite of not actively contributing content and annotations, can at least verify existing annotations if it is easy enough.

The idea of semi-automatically supporting the annotation process is certainly not new and has been suggested before. However, we think that it is only the unique combination of large community efforts, learning algorithms and a seamless integration between both that will ultimately lead to the kind of environments needed to make large scale semantic annotation feasible.

In this paper we thus describe a novel paradigm for semantic annotation which combines the effort of communities such as Wikipedia (the *community intelligence* or *"the numerous"* dimension in our model) which contribute to the massive creation of content with the benefits of a machine learning approach. The learned model captures people's annotation behaviour and is thus able to quickly extract new information and suggest corresponding annotations to be verified by the user community (this the *machine intelligence* or *"the fast"* dimension in our model).

The remainder of this paper is organised as follows. In the next section we describe our approach to combining ma-

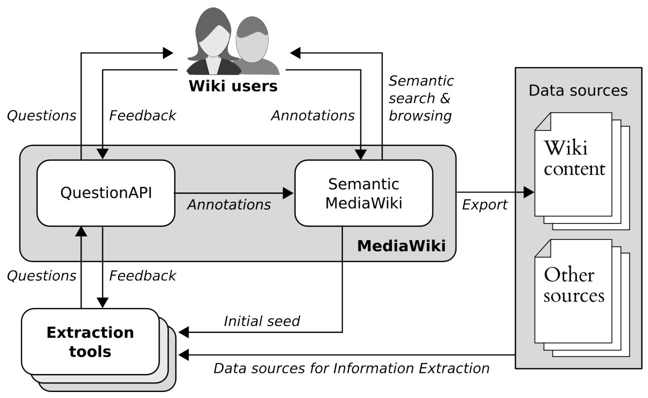


Figure 1: Integrating (semantic) wikis with Information Extraction tools – basic architecture.

chine and human intelligence for semantic annotation in a wiki setting and describe how Semantic MediaWiki can be used for this purpose. Then, we derive requirements for such an integration and describe its corresponding architecture subsequently. We present an implementation based on the English Wikipedia and discuss practical experiences before reviewing related work and concluding.

Combining Human and Machine Intelligence

The crucial aspect of our model is that community members and information extraction algorithms interact in such a way that they can benefit from each other. Humans benefit from the fact that information extraction systems can support them in the tedious work of manual annotation, and algorithms exploit human annotations to bootstrap and learn patterns to suggest new annotations. The workflow in our model is thus as follows:

1. Extraction tools use existing high-quality and community-validated human annotations to learn patterns in data, leading to the extraction of new annotations.
2. Users are requested to verify extracted data so as to confirm or reject it. This is done by presenting questions to users.
3. Confirmed extraction results are immediately incorporated into the wiki, if possible.
4. User replies are evaluated by extraction tools to improve future results (learning), and to gather feedback on extraction quality (evaluation), returning to (1) in a bootstrapping fashion.

The model thus is cyclic, but also asynchronous in nature, since learning, annotation, verification, and incorporation into the wiki interact with each other asynchronously and not in a serialised manner. This mode of operation is reflected in the requirements we present below.

Assuming the model above, we present a concrete architecture and implementation that realises the above model in which extraction tools and wiki users interact in a rather asynchronous mode, benefiting from each other. Figure 1 shows the relevant components – *(Semantic) MediaWiki*, the *extraction tools*, a novel *QuestionAPI* as well as their basic interactions. We have selected the wiki-engine MediaWiki as a basis for our work, since this system is widely used on publicly accessible sites (including Wikipedia), such that large amounts of data are available for annotation. Moreover, the free add-on *Semantic MediaWiki* (SMW) extends MediaWiki with means for creating and storing semantic annotations that are then exploited to provide additional functionality to wiki-users (Krötzsch *et al.* 2007). This infrastructure is useful in two ways: first, it allows wiki-users to make direct use of the freshly acquired annotations, and, second, it can support extraction tools by providing initial (user-generated) example annotations as seeds for learning algorithms.

As shown in Figure 1, our general architecture makes little assumptions about the type and number of the employed extraction tools, so that a wide range of existing tools should be useable with the system (see the Related Work section for an overview). As a concrete example for demonstrating and testing our approach, we have selected the *Pronto* information extraction system (Blohm & Cimiano 2007).

Requirements on User Interaction

Successful wiki projects live from vivid user communities that contribute and maintain content, and therefore social processes and established interaction paradigms are often more important than specific technical features. Likewise, any extended functionality that is to be integrated into existing wikis must also take this into account. This has led us to various requirements.

(U1) Simplicity Participating in the annotation process should be extremely simple for typical wiki users, and should ideally not require any prior instruction. The extension must match the given layout, language, and interface design.

(U2) Unobtrusiveness and opt-out In order to seriously support real-world sites an extension must not obscure the actual main functions of the wiki. Especially, it must be acknowledged that many users of a wiki are passive readers who do not wish to contribute to the collaborative annotation process. Registered users should be able to configure the behaviour of the extension where possible.

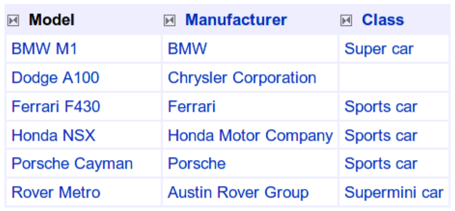
(U3) User gratification Wiki contributors typically are volunteers, such that it is only their personal motivation which determines the amount of time they are willing to spend for providing feedback. Users should thus be rewarded for contributions (e.g. by giving credit to active contributors), and they should understand how their contribution affects and improves the wiki.

(U4) Entertainment Even if users understand the relevance of contributing feedback, measures must be taken to ensure that this task does not appear monotone or even stupid to them. Problems can arise if the majority of changes proposed by extraction tools are incorrect (and maybe even unintelligible to humans), or if only very narrow topic areas are subject to extraction.

(U5) "Social" control over extraction algorithms Wiki users and contributors take responsibility for the quality of the wiki as a whole. Changes to wiki content are

```
The '''Peugeot 204''' is a
[[class::compact car]] produced
by the [[French]] manufacturer
[[manufacturer::Peugeot]] between [[1965]]
and [[1976]].
```

Figure 2: Annotated wiki source text.



Model	Manufacturer	Class
BMW M1	BMW	Super car
Dodge A100	Chrysler Corporation	
Ferrari F430	Ferrari	Sports car
Honda NSX	Honda Motor Company	Sports car
Porsche Cayman	Porsche	Sports car
Rover Metro	Austin Rover Group	Supermini car

Figure 3: Query result in Semantic MediaWiki: automobiles with mid-engine/rear-wheel drive, their manufacturers, and classes where specified.

frequently discussed and reverted if deemed inappropriate. Credibility and authority play a crucial role here. Frequent inappropriate feedback requests and content modifications by information extraction systems may lead to frustration within the community. Therefore we propose to make the extraction tools identifiable by giving their name, methodology and author so that users can identify the origin of an annotation and contact responsible persons.

Semantic MediaWiki

Semantic MediaWiki (SMW) is an open source semantically enhanced wiki engine that enables users to annotate the wiki's contents with explicit, machine-readable information. This information is then used to offer semantic search and browsing facilities within the wiki, as well as to export semantic data in the standardised OWL/RDF format, thus supporting data reuse in other applications. A brief overview of both aspects is provided here – for further details and related work see (Krötzsch *et al.* 2007).

SMW's main annotation mechanism is the assignment of property-value-pairs to pages. Property values might be other pages (e.g. to express relations like "father of"), or data values of a variety of specialised datatypes (e.g. for describing properties like "birthdate" or "population").

Formally, these annotations are interpreted in the Web Ontology Language OWL DL[1], using the *Semantic Wiki Vocabulary and Terminology SWIVT*[2]. Categories map to OWL classes, and categorised pages map to elements of such a class. Properties are directly interpreted as *object* or *datatype properties* in OWL DL, depending on their datatype as declared in the wiki.

Semantic search and browsing features in SMW are included into the wiki interface. One major feature of this kind are semantic queries formulated in a wiki-like query syntax.

[1]`http://www.w3.org/2004/OWL/`
[2]`http://semantic-mediawiki.org/swivt/`

Figure 2 provides a simple example of annotated wiki text, which is the basis for the HTML output of a wiki-page. Square brackets is the standard syntactic notation for hyperlinks, and in SMW these links can be annotated with properties separated by `::` from the link-target. Based on such annotations, SMW can dynamically generate lists of query results, as e.g. the one shown in Figure 3.

Pronto

Pronto is an information extraction system able to extract relations from large collections of text such as the Web on the basis of minimal supervision. The minimal supervision consists of between 5-30 seed examples for the relation in question. Pronto works in a bootstrapping-like fashion by starting from the examples provided and learns new patterns to extract the relation in question by looking at the occurrences of the seed examples in the text collection, generalising these to yield general patterns. These patterns are then used to extract new examples and iterate the process. A pattern extracting the relation *productOf* between products and their manufacturing companies could for example look as follows:

"ARG_1 | is made by ARG_2 and runs ANY at"

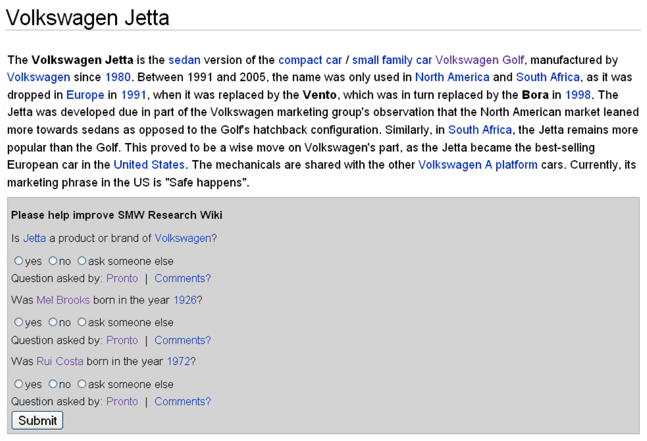
where ARG_1 and ARG_2 represent the argument slots, "|" marks the separation between title and link context (in the case of applying Pronto to a wiki), and ANY is a wildcard that may represent any token. A more detailed description of the Pronto system can be found in (Blohm & Cimiano 2007).

System Design

In this section we discuss the concrete design and implementation of our approach, which realises the basic interactions shown in Figure 1. In order to enable easy integration of many extraction tools in asynchronous operation, all information exchange between wiki and extractors is realised via simple web interfaces, and this web API forms one major part of our *QuestionAPI* extension of MediaWiki developed in the context of the work described here. The other two main components of this module are its internal management of questions and answers, and its user interface extensions in the wiki. All three components will be described in detail below, and it will be explained how the requirements identified are addressed by our particular design. Finally, we explain how contextual information is used to control information requests based on user preferences and content.

User Interface

The main visible component of the QuestionAPI is its extension of the wiki user interface. Requests for feedback on extraction results are presented to the user as multiple-choice questions in a simple web-form, as shown at the bottom of Figure 4. Although we consider further answer formats, the current implementation supports only the answers "yes" and "no", as well as a third option to defer a question. This last option allows users to skip questions without answering them, so that they can continue with other questions instead of accumulating questions that they are unable


Volkswagen Jetta

The **Volkswagen Jetta** is the sedan version of the compact car / small family car Volkswagen Golf, manufactured by Volkswagen since 1980. Between 1991 and 2005, the name was only used in North America and South Africa, as it was dropped in Europe in 1991, when it was replaced by the **Vento**, which was in turn replaced by the **Bora** in 1998. The Jetta was developed due in part of the Volkswagen marketing group's observation that the North American market leaned more towards sedans as opposed to the Golf's hatchback configuration. Similarly, in South Africa, the Jetta remains more popular than the Golf. This proved to be a wise move on Volkswagen's part, as the Jetta became the best-selling European car in the United States. The mechanicals are shared with the other Volkswagen A platform cars. Currently, its marketing phrase in the US is "Safe happens".

Please help improve SMW Research Wiki

Is Jetta a product or brand of Volkswagen?

○ yes ○ no ○ ask someone else

Question asked by: Pronto | Comments?

Was Mel Brooks born in the year 1926?

○ yes ○ no ○ ask someone else

Question asked by: Pronto | Comments?

Was Rui Costa born in the year 1972?

○ yes ○ no ○ ask someone else

Question asked by: Pronto | Comments?

Submit

Figure 4: Questions to users might be displayed at the bottom of wiki pages.

or unwilling to answer. Details on question scheduling are discussed in the following section.

Providing feedback is thus extremely simple, even for users who are not normally editing wiki-text (U1). Simplicity is also achieved by suitable question construction:

- Questions should be specific and informative, and they should use natural formulations instead of technical terms.
- Questions can contain wiki mark-up, and especially they can contain hyperlinks to relevant wiki-pages. This makes it easier for users to look up information.

The architecture assumes that the information extractors implementing the question API will provide their questions in natural language. Note that the right formulation of a question can not be meaningfully automated by using generic templates. Thus, we assume that every information extraction system is responsible to deliver an appropriate formulation of their questions in natural language.

All questions are associated with the extraction tool that requested the feedback, and this information is displayed with each question. A wiki page is maintained for each extraction tool, so that users can find additional information or provide comments (U5).

Besides the general form of the request interface, an important question is *where* to display questions in the wiki. Following our requirement for unobtrusiveness and opt-out (U2), the QuestionAPI can be configured to display a variable number of questions either at the bottom of all wiki pages, or only via a specific web interface ("special page") of the wiki.

After answering one or more questions, users are shown a summary of the submitted answers, as well as the option to answer further questions. The QuestionAPI supports *direct changes* based on answers to questions such that if a user has confirmed a certain semantic information, the QuestionAPI directly adds this fact as an annotation to the wiki. If this is enabled, changes will be done immediately when submitting an answer, and the answering user will get credit for the change just as if she would have edited the wiki manually. While this helps to increase user motivation (U3), it may also seem somewhat risky. But direct changes only *simplify* the editing process – the question whether or not a single user may modify a page still depends on the wiki's settings.

The Web API

The *QuestionAPI* extends MediaWiki with a simple web-based API that extraction tools can use to exchange information with the wiki. The API is protected by a permission control system based on MediaWiki's user permission management. Authentication works by associating to every extraction tool a wiki user-account that is then granted permission to access the question API. Other than being an indispensable feature for preventing abuse of the QuestionAPI, this mechanism also facilitates the management of requests and answers by extraction tools, such that extractors can access only data related to their own requests. Besides the simple use of session cookies for this purpose, all communication is completely stateless.

The QuestionAPI enables extraction systems to pose questions, to request gathered user feedback, and to remove questions from the system. Questions are added by supplying the question text as a parameter (possibly with wiki markup), after which a numerical question ID is returned by the wiki (or 0 if the question was denied). Lists of answers are provided in a simple XML format, and extraction tools may request either all available answers (to their questions), or specify a single question directly. A question is deleted from the system by supplying its ID, and this will also cause all answers to that question to be dropped from the system (though it is possible to have both archived by QuestionAPI as well, e.g. for later statistical evaluation).

The specification of direct changes currently works by specifying a string replacement *and* the page context of that replacement. The latter ensures that replacements happen only if the page still (locally) corresponds to the version inspected by the extraction tool. If other changes occurred, modifications need to be done manually by users.

Practical Experiences

We now present experiences gathered with the implementation of our collaborative semantic annotation framework. We have set up an integrated system based on Wikipedia data[3] which we presented to community members in order to collect feedback and usage data.

The observations discussed here are not meant to be a formal evaluation – information extraction with Pronto on SMW-like annotations on Wikipedia has been formally evaluated in (Blohm & Cimiano 2007), and performance and usage statistics for SMW have been published in (Krötzsch *et al.* 2007). What remains to be investigated is community uptake of the feedback extension as such, and the utility of the derived information. While extensive studies of these aspects must be left to future research, our initial tests have provided us with important insights for improving the current design.

[3]`http://testserver.semantic-mediawiki.org`

We created a mirror of the English Wikipedia based on a Wikipedia database dump from December 17th 2006. The server runs current versions of MediaWiki (1.12alpha) and SMW (1.0RC1), as well as our new extension QuestionAPI. For maintenance and performance reasons, software components were distributed over three server-sized computers: one running the PHP server for MediaWiki and its extension, one providing the database, and one running the Pronto extraction system. The systems were able to serve pages at below 1 second response time, and to run Pronto at its regular extraction rate of 0.3 facts per second.

Experienced wiki users and developers were asked to test the system via wiki-related mailing lists, and during a time of 5 days, 40 users (estimated from the number of distinct IPs) provided a total of 511 answers to the QuestionAPI.

Of the 511 questions answered, 51% were answered with "no", 34% were deferred, and the remaining 15% were answered with "yes" which in our setup led to automatic knowledge insertion. All users reacted positively to the interaction paradigm. The general purpose of the questions was quickly understood and appreciated, and no concerns were expressed with respect to obstructiveness or lack of simplicity. Several users mentioned that the questions reminded them of a quiz game, and suggested further uses of this extension beyond information extraction. We interpret this as positive effect with respect to the entertainment requirement (U4).

During the experiment, the option for deferring a question had been labelled "don't know" which was changed to "ask someone else" only later. This labelling is assumed to be responsible for the large portion of "don't know" answers: users who considered the questions as a kind of quiz mentioned that they perceived it as "cheating" to look up an answer that they were not sure about, such that "don't know" was considered more appropriate. This indicates that some introduction and/or clearer labelling is still needed to better convey the purpose of the questions. One consequence of this insight was the relabelling of "don't know" to "ask someone else" so as to communicate that personal knowledge is not to be tested, while still encouraging an answer by reminding the user that the task will otherwise be left to other users.

Besides some bug reports about character encoding, the only actual complains from users were related to the content of some types of questions, especially in cases where systematic errors occurred. This also produced some suggestions for filtering Wikipedia-specific extraction errors, e.g. caused by special kinds of frequent summary articles ("List of . . . ") that can normally not be involved in any relation.

In order to account for these observations, we formulate an extension of the *entertainment* requirement (U4): It is important to ensure that systematic errors in suggested relations are minimised beforehand, and excluded from verification through collaborative annotation. One interesting approach to do this automatically could be the use of unsupervised clustering methods that detect regularities, and to exclude questions belonging to large clusters for which only "no" answers have been provided so far. For this purpose, an additional answer option can be introduced to allow the users to mark individual relation instances as "unreasonable" suggestions.

Related Work

Annotation of web content has become very popular in particular as *tagging* of various kinds of media resources. Cameron Marlow et al. (Marlow *et al.* 2006) give an overview of tagging systems, and discuss dimensions in which they can differ. While not a tagging system in the stricter sense, the setup presented here would thereby be classified as a *free-for-all set model* system with high *resource connectivity* and a special form of *tag support*. The paper discusses various forms of incentives ranging from future retrieval to opinion expression. As Wikipedia already has a vivid community, we did not consider incentives for this study, and assume that our architecture helps to involve a larger user community by providing a low-entry threshold for contribution. An innovative approach with respect to incentives and human-machine collaboration in tagging is the ESP game (von Ahn & Dabbish 2004) which asks pairs of users to come up with common tags for images by guessing what the other user might tag. Further related work is done in the field of assisted semantic annotation of websites (e.g. (Dzbor, Domingue, & Motta 2003)). While our approach is largely tailored to semantifying sources like Wikipedia, other projects have studied the interaction between human input of facts and data mining technology. The Open Mind initiative studies the interaction of Internet users and knowledge bases. Their Common Sense (Pentney *et al.* 2007) system prompts users for natural language statements on a given entity. In a similar way, the Knowledge Base of the True Knowledge^{TM} question answering system can be extended by users.

Unlike in classical tagging, annotations in Semantic MediaWiki are structured statements that establish relationships between entities, or describe properties of these. This is possible because each page is assumed to describe an ontological element, and links are assumed to express relations between them. As described above, annotations in SMW have a formal semantics suitable for exchanging them via the Web. Some tagging systems are also working towards a more formal interpretability of tags. Flickr (`http://www.flickr.com`) introduced "machine tags" which allow unambiguous expression of facts about the annotated media. Bibsonomy (Hotho *et al.* 2006) provides the possibility to organise tags by asserting relations among them. The Spock person search engine (`http://www.spock.com`) provides the possibility to mark existing tags as correct and incorrect, which is not completely unlike the question based interaction in our setting.

While in our implementation we use information extraction from text to automatically derive suggested annotations of Wikipedia hyperlinks, our architecture is not limited to that setting. As reviewed and discussed in (Hendler & Golbeck 2008), much potential lies in the links and network structure as well as in social connections between users. The authors argue that the social interactions enabled by annotation constitute an important incentive for producing them.

Wikipedia is currently widely used for information extraction from text. Suchanek et al. (Suchanek, Kasneci, & Weikum 2007) have focussed on high-precision ontology learning and population with methods specifically tailored to Wikipedia. Wikipedia's category system is exploited assuming typical namings and composition of categories that allow to deduce semantic relations from category membership. In (Ruiz-Casado, Alfonseca, & Castells 2005) information extraction from Wikipedia text is done using hyperlinks as indicators for relations just like in the present study. As opposed to the work presented here it relies on WordNet as a hand-crafted formal taxonomy and is thus limited to relations for which such sources exist. Strube and Ponzetto use the taxonomy of the Wikipedia categories to define a measure for the semantic relatedness between words (Strube & Ponzetto 2006).

Conclusion and Next Steps

We have presented a new approach for facilitating semantic annotation of wikis by means of community-supervised information extraction, and we have presented a concrete practical realisation of this idea based on Semantic MediaWiki and an extraction system. Our robust and flexible design enables the loose, web-based integration of a wide range of extraction tools into existing community portals – thus tapping a large application field for information extraction on the one hand, and new content-creation solutions for community platforms on the other.

Our contribution removes the major barrier between two vibrant fields of research and application, and thus opens up a multitude of new opportunities for both areas. The first step certainly is to apply and evaluate information extraction tools on real-world community platforms. Our approach has been designed to be completely open, such that existing extraction tools can use our system with very little effort. We will open our Wikipedia-mirror to help developers of extraction tools to conduct tests in large scale real-world contexts, and to solicit user-feedback. We also consider a similar setup for conducting a "Wikipedia extraction challenge" where various types of extraction tools can demonstrate their utility in a kind of annotation contest. Further future work includes putting questions in contexts where visitors can be assumed to have the knowledge to answer them, integrating more question types. Additionally aggregating multiple user answers (e.g. by majority vote) could increase annotation quality.

On the other hand, there is a very real need for high quality and high coverage annotations in modern community sites. Many users of our Semantic MediaWiki system have made this request, both in community portals and in intranet applications.

Thus, when practical experiments have shown the maturity of extraction tools, there is also a clear path towards wide adoption and exploitation (economic or otherwise, e.g. in semantifying Wikipedia). In this way, information extraction – currently still mostly a mere consumer of Web-content – can take its proper place as a key technology for modern community platforms, and a major enabler of the Semantic Web.

Acknowledgements

This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) program under EC grant number IST-FP6-026978.

References

Blohm, S., and Cimiano, P. 2007. Using the web to reduce data sparseness in pattern-based information extraction. In *Proceedings of the ECML PKDD*. Springer.

Dzbor, M.; Domingue, J.; and Motta, E. 2003. Magpie – Towards a Semantic Web browser. In *Proc. 2nd International Semantic Web Conference (ISWC-03)*, volume 2870 of *Lecture Notes in Computer Science*, 690–705.

Hendler, J., and Golbeck, J. 2008. Metcalfe's law, Web 2.0, and the Semantic Web. *Journal of Web Semantics* 6(1):14–20.

Hotho, A.; Jäschke, R.; Schmitz, C.; and Stumme, G. 2006. BibSonomy: A social bookmark and publication sharing system. In *Proc. 2006 Conceptual Structures Tool Interoperability Workshop*, 87–102.

Krötzsch, M.; Vrandečić, D.; Völkel, M.; Haller, H.; and Studer, R. 2007. Semantic Wikipedia. *Journal of Web Semantics* 5(4):251–261.

Marlow, C.; Naaman, M.; Boyd, D.; and Davis, M. 2006. HT06, tagging paper, taxonomy, Flickr, academic article, to read. In *Proc. 17th Conf. on Hypertext and Hypermedia (HYPERTEXT-06)*, 31–40. New York, NY, USA: ACM.

Pentney, W.; Philipose, M.; Bilmes, J. A.; and Kautz, H. A. 2007. Learning large scale common sense models of everyday life. In *Proc. 22nd Nat. Conf. on Artificial Intelligence (AAAI-07)*, 465–470.

Ruiz-Casado, M.; Alfonseca, E.; and Castells, P. 2005. Automatic extraction of semantic relationships for WordNet by means of pattern learning from Wikipedia. In *Natural Language Processing and Information Systems*. Berlin/Heidelberg: Springer.

Strube, M., and Ponzetto, S. P. 2006. WikiRelate! Computing semantic relatedness using Wikipedia. In *Proc. 21st Nat. Conf. on Artificial Intelligence (AAAI-06)*, 1419–1424.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: A core of semantic knowledge. In *Proc. 16th Int. Conf. on World Wide Web (WWW-07)*, 697–706. ACM Press.

von Ahn, L., and Dabbish, L. 2004. Labeling images with a computer game. In *Proc. SIGCHI Conf. on Human Factors in Computing Systems (CHI-04)*, 319–326. New York, NY, USA: ACM Press.

Learning to Predict the Quality of Contributions to Wikipedia

Gregory Druck and **Gerome Miklau** and **Andrew McCallum**
{gdruck,miklau,mccallum}@cs.umass.edu
Department of Computer Science
University of Massachusetts
Amherst, MA 01003

Abstract

Although some have argued that Wikipedia's open edit policy is one of the primary reasons for its success, it also raises concerns about quality — vandalism, bias, and errors can be problems. Despite these challenges, Wikipedia articles are often (perhaps surprisingly) of high quality, which many attribute to both the dedicated Wikipedia community and "good Samaritan" users. As Wikipedia continues to grow, however, it becomes more difficult for these users to keep up with the increasing number of articles and edits. This motivates the development of tools to assist users in creating and maintaining quality. In this paper, we propose metrics that quantify the quality of contributions to Wikipedia through implicit feedback from the community. We then learn discriminative probabilistic models that predict the quality of a new edit using features of the changes made, the author of the edit, and the article being edited. Through estimating parameters for these models, we also gain an understanding of factors that influence quality. We advocate using edit quality predictions and information gleaned from model analysis not to place restrictions on editing, but to instead alert users to potential quality problems, and to facilitate the development of additional incentives for contributors. We evaluate the edit quality prediction models on the Spanish Wikipedia. Experiments demonstrate that the models perform better when given access to content-based features of the edit, rather than only features of contributing user. This suggests that a user-based solution to the Wikipedia quality problem may not be sufficient.

Introduction and Motivation

Collaborative content generation systems such as Wikipedia are promising because they facilitate the integration of information from many disparate sources. Wikipedia is remarkable because anyone can edit an article. Some argue that this open edit policy is one of the key reasons for its success (Roth 2007; Riehle 2006). However, this openness does raise concerns about quality — vandalism, bias, and errors can be problems (Denning et al. 2005; Riehle 2006; Kittur et al. 2007).

Despite the challenges associated with an open edit policy, Wikipedia articles are often of high quality (Giles 2005). Many suggest that this is a result of dedicated users that make many edits, monitor articles for changes, and engage in debates on article discussion pages. These users are sometimes referred to as "zealots" (Anthony, Smith, and Williamson 2007), and studies claim that they are motivated by a system of peer recognition that bears resemblance to the academic community (Forte and Bruckman 2005). However, the contributions of "good Samaritan" users, who edit articles but have no desire to participate in the community, cannot not be underestimated (Anthony, Smith, and Williamson 2007).

As Wikipedia continues to grow, however, it becomes more difficult for these users to keep up with the increasing number of articles and edits. Zealots comprise a relatively small portion of all Wikipedia users. Good Samaritan users are not likely to seek out errors, but instead rely on stumbling upon them. It is interesting to consider whether aiding users in detecting and focusing effort on quality problems could improve Wikipedia.

In this paper, we examine the problem of estimating the quality of a new edit. Immediately, we face the problem of defining edit quality. It has been argued that there is no general definition of information quality, and hence quality must be defined using empirical observations of community interactions (Stvilia et al. 2008). Therefore, we define quality using implicit feedback from the Wikipedia community itself. That is, by observing the community's response to a particular edit, we can estimate the edit's quality. The quality metrics we propose are based on the assumption that edits to an article that are retained in subsequent versions of the article are of high quality, whereas edits that are quickly removed are of low quality.

We use these community-defined measures of edit quality to learn statistical models that can predict the quality of a new edit. Quality is predicted using features of the edit itself, the author of the edit, and the article being edited. Through learning to predict quality, we also learn about factors that influence quality. Specifically, we provide analysis of model parameters to determine which features are the most useful for predicting quality.

We advocate using edit quality predictions and information gleaned from model analysis not to place restrictions on editing, but to assist users in improving quality. That is, we aim to maintain a low barrier to participation, as those users not interested in the Wikipedia community can still be valuable contributors (Anthony, Smith, and Williamson

2007). Restrictions might also discourage new users, and drive away users who were drawn to the idea of a openly editable encyclopedia. Consequently, we suggest that the quality models be used to help users focus on predicted quality problems or to encourage participation.

We evaluate the edit quality prediction models and provide analysis using the Spanish Wikipedia. Experiments demonstrate that the models attain better results when given access to content-based features, in addition to features of the contributing user. This suggests that a user-based solution to the Wikipedia quality problem may not be sufficient.

Although we focus on Wikipedia in this paper, we think of this as an instance of a new problem: automatically predicting the quality of contributions in a collaborative environment.

Related Work

Many researchers are skeptical of Wikipedia, as there are reasons to expect it to produce poor quality articles (Denning et al. 2005). Surprisingly, however, a study found that Wikipedia articles are only of slightly lower quality than their counterparts in Britannica, a professionally written encyclopedia (Giles 2005). As a result, Wikipedia has attracted much interest from the research community.

Stvilia et al. (2008) present an overview of the mechanisms used by the Wikipedia community to create and maintain information quality, and describe various categories of quality problems that occur. Roth (2007) analyzes factors that allow Wikipedia to remain viable while other wikis fail. The primary requirements for a viable wiki are quality content and a sufficient mass of users that can maintain it. It is suggested that an overworked user base may result in the abandonment of a wiki. This motivates our work, which aims to provide tools to help users maintain quality.

A controversial aspect of Wikipedia is that any user is allowed to edit any article. However, there is evidence that this openness is beneficial. Riehle (2006) interviews high profile Wikipedia contributors who support the open edit policy, but want more explicit incentive systems and better quality control (again motivating the work in this paper). Anthony et al. (2007) find that both unregistered users with few edits, or “good Samaritans”, and registered users with many edits, or “zealots” contribute high-quality edits. Interestingly, as the number of edits contributed increases, quality decreases for unregistered users, whereas it increases for registered users. Although the “good Samaritan” users seemingly make edits without the need for recognition, some registered users are clearly interested in being recognized. Forte and Bruckman (2005) examine the motivation of registered users in Wikipedia and compare their incentives to those in the scientific community. Similarly to researchers, Forte and Bruckman argue that some Wikipedia users want to be recognized by their peers, and gain credibility that will help them to effect change. This suggests that developing better methods for attributing credit to users would benefit Wikipedia.

A reputation system is one way to attribute credit to users. Adler and Alfaro (2007) propose an automatic user reputation system for Wikipedia. In addition to encouraging high-quality contributions, this system can be used to identify potential quality problems by flagging edits made by low reputation users. Similarly to the work in this paper, Adler and Alfaro quantify the quality of a user's edits by observing the community reaction to them in the edit history. When they use their author reputation scores to classify low-quality edits, the resulting precision is fairly low. This is to be expected because users with good intentions but few edits have low reputation. Additionally, a large portion of edits come from unregistered users who have low reputation by default. In the previous paragraph, it was suggested that these users can be beneficial. This motivates a quality prediction model that considers information about the edit itself in addition to information about the user. Although we do not compare directly with the method of Adler and Alfaro, we compare with a quality prediction model that only has access to user features, and find that the addition of edit content features consistently improves performance.

There has also been work that aims to detect quality at the granularity of articles, rather than edits. Wilkinson and Huberman (2007) find that articles with more editors and more edits are of higher quality. Dondio et al. (2006) propose a heuristic method for computing the trustworthiness of Wikipedia articles based on article stability and the collaborative environment that produced the article. Kittur (2007) shows that the number of edits to meta (non-article) pages is increasing, illustrating that more effort is being expended on coordination as Wikipedia grows. Kittur also uses features that quantify conflict to predict whether articles will be tagged *controversial*.

Defining Edit Quality

It has been argued that there is no general definition of information quality, and hence quality must be defined in relation to the community of interest (Stvilia et al. 2008). That is, quality is a social construct. If we are able to observe the operation of the community of interest, however, we can use the actions of the community to quantify quality.

In this paper, we quantify the quality of an edit to an article using implicit feedback from the Wikipedia community. This feedback can be obtained by observing the article edit history, which is openly available. We choose to use implicit feedback, rather than soliciting quality assessments directly, because it allows us to automatically estimate the quality of any contribution. We propose two measures of quality. Both are based on the assumption that edits to an article that are retained in subsequent versions of the article are of high quality, whereas edits that are quickly removed are of low quality. Although this assumption may be locally violated, for example by edits to a current events page, in the aggregate this assumption seems reasonable.

In Wikipedia terminology, a *revert* is an edit that returns the article to a previous version. That is, we say that the jth edit to an article was reverted if the ith version of the article is identical to the kth version of the article, where $i < j < k$. These events often occur when an article is vandalized, or when an edit does not follow the conventions of the article. The first quality measure we propose is simply whether or not an edit was reverted.

A problem with the revert quality judgment is that it only indicates contributions of the lowest quality — those which provide no value (as judged by the community) and are completely erased. We would also like to know about other ranges of the quality spectrum.

Therefore, we define a quality metric which we call *expected longevity*. To do so, we introduce some notation. We denote the ith version of article a as a_i. Each a_i is represented as a sequence of tokens, so that the kth token of the ith version of the article is a_{ik}. Each $a_{ik} = \langle s, i \rangle$, where s is the token text, and i indicates the edit that introduced the token[1]. Let $D(a_i, a_j)$ be the set of tokens that appear in a_j, but not in a_i. Let t_i be the time of the ith edit to article a. We define the expected longevity of edit i, $l(a, i)$, as:

$$l(a,i) = \sum_{j=i+1}^{k} \left(1 - \frac{|D(a_{i-1}, a_i) - D(a_j, a_{j-1})|}{|D(a_{i-1}, a_i)|}\right)\left(t_j - t_i\right),$$

where k is the first edit in which all of the tokens changed or added in edit i have been subsequently changed or deleted. The first parenthesized value in the above equation computes the proportion of the tokens added or changed in edit i that were removed or changed by edit j. Therefore, the expected longevity is the average amount of time before a token added or changed by edit i is subsequently changed or deleted. In some cases, the tokens added or changed by a particular edit are never subsequently changed or deleted. In this case, the unedited tokens are treated as though they were edited by the last edit to the article.

However, there is a potential problem with the above definition. Suppose that the tokens added or changed by edit i are entirely changed or deleted by edit $i + 1$, but edit $i + 1$ is reverted by edit $i + 2$. In this case, we expect $l(a, i)$ to be unfairly small. We handle this problem by ignoring edits that were reverted in the computation of expected longevity.

We note that the expected longevity quality metric (unlike the revert quality metric) is undefined for edits that only delete content. Additionally, we note that including time in expected longevity could be problematic for articles in which edits are very infrequent. We plan to address these issues in future work.

Predicting Edit Quality

We next develop models to predict the quality of new edits. We choose to use machine learning, rather than some handcrafted policy, because it gives statistical guarantees of generalization to unseen data, allows easy updating as new training data becomes available, and may provide more security since the model family, features, and parameters would (ideally) be unknown to attackers.

We choose to learn a single model that can predict the quality of an edit to any Wikipedia article, rather than separate models for each article. This allows the model to learn broad patterns across articles, rather than learn some very specific aspects of a particular article. However, we can still give the model access to article-specific information with this setup.

Predicting whether an edit will be reverted is a binary classification problem. We can model expected longevity directly using regression, but we may not necessarily require precise estimates. In this paper we instead use discrete categories for expected longevity intervals.

Importantly, the models need to be scalable to make learning on Wikipedia-scale data tractable. Therefore, we choose a simple, discriminatively-trained probabilistic log-linear model. Discriminative training aims to maximize the likelihood of the output variables conditioned on the input variables. An advantage of discriminative training is that it allows the inclusion of arbitrary, overlapping features on the input variables without needing to model dependencies between them. We leverage this capability by combining many different types of features of the user, the content of the edit, and the article being edited.

Importantly, we also use aggregate features, which can consider both the edit in question and all other edits in the training data. For example, in addition to a feature for the user of a particular edit, we can include a feature that counts the number of other edits the user contributed in the training data. We provide more discussion of features in the next section.

The probability of a quality label y given a particular edit $\mathbf{x}$ and the set of all edits in the training data $\mathbf{X}$ is

$$p_\lambda(y|\mathbf{x};\mathbf{X}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_i \lambda_i f_i(\mathbf{x}, \mathbf{X}, y)\right),$$

where f_i are feature functions. Although we allow arbitrary features over a single quality label y and all of the input variables in the training data, in this paper we do not model dependencies between the quality labels of different edits. We have some preliminary evidence that accounting for sequential dependencies between the quality predictions of edits to the same article may be beneficial, and plan to pursue this in future work.

We choose parameters $\hat{\lambda}$ for this model that maximize the conditional log likelihood of the training data D, with an additional Gaussian prior on parameters that helps to prevent overfitting. The optimization problem is then

$$\hat{\lambda} = \arg\max_{\lambda} \sum_{(\mathbf{x},y)\in D} \log p_\lambda(y|\mathbf{x};\mathbf{X}) - \sum_i \frac{\lambda_i^2}{2\sigma}.$$

We choose parameters $\hat{\lambda}$ using numerical optimization.

Features

In this section we describe the feature functions f_i that we use in the quality prediction models. We define content features as the set of all features below except those under the **User** heading.

Change Type features quantify the types of changes the edit makes. We include features for the log of the number of *additions*, *deletions*, and *changes* that the edit makes, as well as the proportions of each of these change types. Additionally, we use features that specify which of change types

[1]We determine correspondences between article versions using a differencing algorithm.

are observed, for example *delete only*, and differences between change type counts, for example the log of *additions - deletions*.

Structural features quantify changes to the structure of the article. Specifically, there are features for *links*, *variables*, *headings*, *images*, and other forms of wiki markup, all concatenated with the change type. For example, one possible feature is *add_link*.

Word features look at the specific words that are added, deleted, or changed by an edit. That is, for a specific word w we have features for w concatenated with a change type, for example *delete_w*. Before adding word features, we strip punctuation and lowercase the text. We also aggregate over the complete history to obtain *low-quality* and *high-quality lexicons*, which are used as features. Finally, we use regular expression features for capitalization patterns, numbers, dates, times, punctuation, and long, repetitive strings.

Article features provide a way to incorporate information about the specific article being edited into the global model. Specifically, we include a feature for each edit that indicates the article to which the edit was made, as well as a feature for the popularity (measured in terms of the log of the number of edits) of the article.

User features describe the author of the edit. We use the *username* of the author (or prefixes of the IP address if the user is unregistered) as a feature, as well as whether or not they are a *registered* user. We identify each registered user as a *bot* or a *human* with a binary feature. We additionally include aggregate features for the log of the number of edits the user has made to any article, the specific article being edited, any meta page, and the discussion page for the specific article being edited. There are also binary features that specify that a user has never edited one of the above types of pages. Finally, there are features that indicate the number of high and low quality edits the user contributed in the training data.

We also include a feature for the log of the epoch time at which the edit was contributed. This feature is helpful because we observe that reverts are becoming more common with time.

An additional set of features we are working to include are based on probabilities of changes under generative models of the articles. Features that quantify the formality or informality of language and features that identify subjectivity and objectivity would also be useful.

Data Processing

We perform experiments using a dump of the Spanish Wikipedia dated 12/02/07. We choose to use the Spanish Wikipedia because the English Wikipedia complete history dump failed for several consecutive months[2], and the authors have some familiarity with Spanish, making it easier to perform error analysis. The Spanish Wikipedia contains over 325,000 articles, and is one of the top 10 largest Wikipedias.

[2]A complete history dump of the English Wikipedia has recently completed successfully. We plan to use it for future experiments.

For the results that follow, we randomly selected a subset of 50,000 articles, each with at least 10 edits. It is common for a single user to make a sequence of edits to an article in a short amount of time. We collapse these sequences (within 1 hour) into a single edit. After collapsing, the total number of edits in this set is 1.6 million. We then tokenize the input so that words and wiki markup fragments are tokens.

To find reverted edits, we look for cases in which article version a_{i-c} is the same as article version a_i, for $2 \leq c \leq C$. This signifies that edits $i-c+1$ through $i-1$ were reverted. This requires $O(Cn)$ comparisons for each article, where n is the number of edits to the article, and C is a constant that specifies the maximum size of the window. In this paper, we use $C = 5$.

A naive implementation of the computation of expected longevity would require $O(n^2)$ runs of a differencing algorithm per article. However, if we maintain a data structure that represents the subsections of the article that were added or changed by each edit, we can do this in linear time. For each article version, we compute its difference from the previous article version using an implementation of longest common subsequence dynamic programming algorithm. We use this information to update the data structure, and the expected longevity of previous edits whose additions and changes were changed or deleted by the most recent edit. We ignore edits that were reverted, so that the expected longevity is a more accurate indicator of how long an edit remains in the article. This requires $O(n)$ runs of the differencing algorithm per article.

The processed data contains millions of features, making learning slow, and causing the model to overfit. As a result, we remove features that occur less than 5 times in the training data.

For these experiments, we use the edits from December of 2006 through May of 2007 for training, and the edits from June 2007 for evaluation. Importantly, the evaluation data is held-out during training, so that we observe the ability of the model to predict the quality of unseen edits, rather than describe the available edits.

Quality Prediction Experiments

In this section, we compare the quality predictions obtained with a model that uses all features, and a baseline model that uses only user features. We compare these models using precision-recall curves for predicting whether an edit is low quality (either reverted or in the lowest longevity category). We present summary statistics of these curves using the maximum F_α measure. The F_α measure is defined as $\alpha pr/(\alpha p + r)$, where p is precision and r is recall. The F_1 measure is then the harmonic mean of precision and recall. The F_2 measure weights recall twice as much as precision, and the $F_{.5}$ measure weights precision twice as much as recall.

We provide results with all three of these statistics because there are arguments for preferring both high recall and high precision systems. A high recall system does not miss many low-quality edits, but may be imprecise in its predictions. A high precision system only flags edits as low-quality

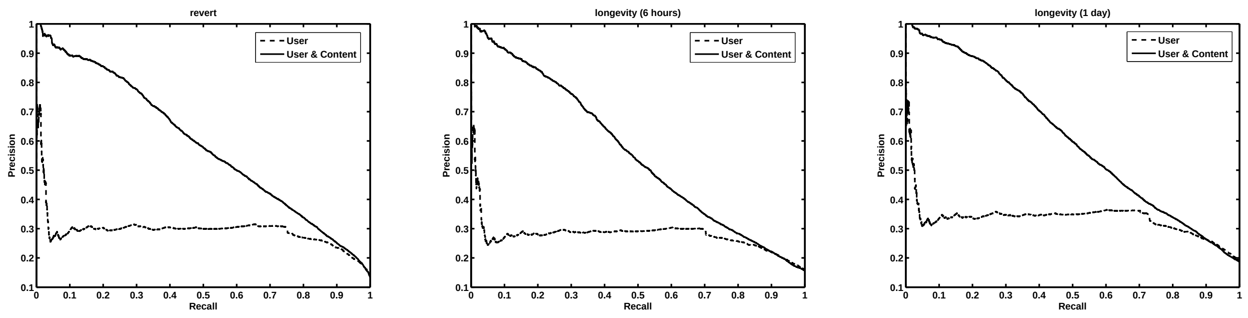


Figure 1: Precision vs. recall curves for models that use only user features and models that additionally use content features to predict the revert, expected longevity (6 hours), and expected longevity (1 day) quality metrics. The model with content features outperforms the user model except in some places at the high recall end of the curve.

features	max F_1	max $F_{.5}$	max F_2
user	0.435	0.382	0.505
+ content	**0.547**	**0.551**	**0.573**

Table 1: Results for predicting reverted edits on the test data.

features	max F_1	max $F_{.5}$	max F_2
user	0.419	0.370	0.483
+ content	**0.518**	**0.538**	**0.535**

Table 2: Results for predicting expected longevity (6 hours).

features	max F_1	max $F_{.5}$	max F_2
user	0.477	0.431	0.535
+ content	**0.550**	**0.567**	**0.569**

Table 3: Results for predicting expected longevity (1 day).

if the model is very confident, but this means that some edits that are actually low quality may be missed.

We first evaluate the quality prediction models on the task of predicting that a new edit will be subsequently reverted. The F_α results are presented in Table 1, while the precision recall curves are presented in Figure 1. We note that the model with access to content features, in addition to user features, performs better in all three F_α comparisons.

We next evaluate quality models on the task of predicting the expected longevity of a new edit. For these experiments we use expected longevity cutoffs such that edits with expected longevity less than 6 hours or 1 day are considered low quality. The results are presented in Tables 2 and 3, and Figure 1. Again, the model with content features performs better under all three F_α measures.

We next aim to understand the poor performance of the user features model. We observe in the user features model precision-recall curves in Figure 1 that there are a very small number of edits for which the precision is high. These edits are contributed by users who often submit low-quality edits. Near recall of 1, the precision dips because edits performed by good users are starting to be classified as low quality. In between, the precision-recall curves are essentially flat, because the users are either unseen during training (33% of the users in June 2007 test data are unobserved in training data), or there is not enough information about them to make a clear decision.

We consider these results promising. The learning task is extremely difficult because the quality metrics are noisy and correctly classifying some edits would require a deep semantic understanding of the article. We also note that there are many other features which would likely improve the results. However, we argue that a model that can, for example, predict whether an edit will be reverted with 80% precision and 30% recall could be useful to Wikipedia users.

Analysis

We now analyze the parameters of the model in order to increase our understanding of the factors that influence edit quality. Some of the most important features for predicting reverts are presented in Table 4 (the important features for predicting longevity are similar). Below, we discuss these and other important features in detail.

- Although unregistered users do contribute 75% of the low-quality edits, they also contribute 20% of all high-quality edits. Therefore, bias against unregistered users results in a system with high recall but low precision.
- Users who previously contributed high or low quality edits tend to continue to submit high and low quality edits, respectively.
- Unregistered users with one or no previously contributed edits often contribute high-quality edits.
- As the number of edits a registered user contributes increases, the quality of their edits increases.
- The percentage of low-quality edits is increasing over time. For example, in October 2005, 8.8% of all edits were reverted, whereas 11.1% of edits in October 2006 were reverted, and 17.8% of all edits in October 2007 were reverted.

↓ NUM USER REVERTED CONTRIBUTIONS
↑ NUM USER HIGH-QUALITY CONTRIBUTIONS
↓ EDIT EPOCH TIME
↓ ADD LOW-QUALITY WORD
↑ REGISTERED USER EDIT COUNT
↓ USER PAGE EDIT COUNT
↑ ADD LINK
↓ ADD ONLY
↓ CHANGE POSITIVE SIZE
↓ CHANGE NEGATIVE SIZE
↑ ADD PUNCTUATION
↓ DELETE LINK
↑ UNREGISTERED USER EDIT COUNT ZERO OR ONE
↑ ADD HIGH-QUALITY WORD
↓ ARTICLE EDIT COUNT

Table 4: The most important features for predicting quality. An ↑ indicates the feature is associated with high quality, whereas a ↓ indicates low quality.

- Articles that are more popular, where popularity is measured in terms of the number of edits, receive a higher percentage of low-quality edits.
- The adding of a link, heading, or other structural element tends to indicate high quality, whereas changing or deleting a structural element indicates low quality.
- Adding punctuation indicates high quality.
- There exist lists of words that tend to indicate high and low-quality edits.
- Large changes in the size of the article, whether a result of additions or deletions, indicate low quality. This suggests that the Wikipedia users who maintain articles are reluctant to allow major changes.
- Surprisingly, edits contributed by users who have edited the article in question many times are often low-quality. This is likely a result of edit wars.

Example Application: Improved Watch List

Wikipedia users can be notified of changes to articles by joining the article's *Watch List*. We suggest an improved Watch List that prioritizes edits according to the confidence of quality predictions. In addition to notifying users on the list, we can also seek out other qualified users to address quality problems. To ensure that the user is knowledgeable about subject of the article, we can leverage work on the reviewer assignment problem. We can determine reputable authors by using a simple reputation system based on the quality metrics. An advantage that the quality prediction models afford is that we can avoid the so-called "ramp-up" problem with author reputation. Typically, a reputation system cannot assign a meaningful reputation score to a new or unregistered user, or incorporate recent contributions, because time is needed for the community to assess them. With the aid of a quality prediction model, we can use estimated quality values for new edits, allowing us to have a meaningful reputation estimate immediately.

Conclusion

We have used the implicit judgments of the Wikipedia community to quantify the quality of contributions. Using relatively simple features, we learned probabilistic models to predict quality. Interestingly, a model that has access to features of the edit itself consistently outperforms a model that only considers features of the contributing user. Through analysis of the parameters of these models, we gained insight into the factors that influence quality. Although we have focused on Wikipedia, we think of this as an instance of a new problem: automatically predicting the quality of contributions in a collaborative environment

Acknowledgements

This work was supported in part by the Center for Intelligent Information Retrieval, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, and in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under contract number NBCHD030010, and AFRL #FA8750-07-D-0185. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

Adler, B. T., and de Alfaro, L. 2007. A content-driven reputation system for the wikipedia. In *WWW*, 261–270.

Anthony, D.; Smith, S. W.; and Williamson, T. 2007. The quality of open source production: Zealots and good samaritans in the case of wikipedia. Technical Report TR2007-606, Dartmouth College, Computer Science.

Denning, P.; Horning, J.; Parnas, D.; and Weinstein, L. 2005. Wikipedia risks. *Commun. ACM* 48(12):152–152.

Dondio, P.; Barrett, S.; Weber, S.; and Seigneur, J. 2006. Extracting trust from domain analysis: A case study on the wikipedia project. *Autonomic and Trusted Computing* 362–373.

Forte, A., and Bruckman, A. 2005. Why do people write for wikipedia? incentives to contribute to open-content publishing. In *GROUP 05 Workshop on Sustaining Community: The Role and Design of Incentive Mechanisms in Online Systems*.

Giles, J. 2005. Internet encyclopaedias go head to head. *Nature* 438:900–901.

Kittur, A.; Suh, B.; Pendleton, B. A.; and Chi, E. H. 2007. He says, she says: conflict and coordination in wikipedia. In *CHI*, 453–462.

Riehle, D. 2006. How and why wikipedia works: an interview with angela beesley, elisabeth bauer, and kizu naoko. In *Proceedings of the International Symposium on Wikis*, 3–8.

Roth, C. 2007. Viable wikis: struggle for life in the wikisphere. In *Proceedings of the International Symposium on Wikis*, 119–124. ACM.

Stvilia, B.; Twidale, M. B.; Smith, L. C.; and Gasser, L. 2008. Information quality work organization in wikipedia. *JASIST* 59(6):983–1001.

Wilkinson, D. M., and Huberman, B. A. 2007. Cooperation and quality in wikipedia. In *Proceedings of the International Symposium on Wikis*, 157–164.

Integrating Cyc and Wikipedia: Folksonomy Meets Rigorously Defined Common-Sense

Olena Medelyan
Department of Computer Science
University of Waikato, New Zealand
olena@cs.waikato.ac.nz

Catherine Legg
Department of Philosophy and Religious Studies
University of Waikato, New Zealand
clegg@waikato.ac.nz

Abstract

Integration of ontologies begins with establishing mappings between their concept entries. We map categories from the largest manually-built ontology, Cyc, onto Wikipedia articles describing corresponding concepts. Our method draws both on Wikipedia's rich but chaotic hyperlink structure and Cyc's carefully defined taxonomic and common-sense knowledge. On 9,333 manual alignments by one person, we achieve an F-measure of 90%; on 100 alignments by six human subjects the average agreement of the method with the subject is close to their agreement with each other. We cover 62.8% of Cyc categories relating to common-sense knowledge and discuss what further information might be added to Cyc given this substantial new alignment.

1. Introduction

As information sharing became ever more sophisticated and globalized from the 1980s onwards, a new research frontier formed around the ambitious goal of developing a machine-understandable conceptual scheme (a.k.a. 'formal ontology') which would mediate information transfer in any conceivable format (Gruber, 1995). Yet current ontology research is still far from delivering such a useful product. The enormous number of concepts in human language must somehow be represented in an ontology. However, it is not enough just to index the names of concepts in some canonical list – a useful ontology needs also to capture defining facts about them, and reason about these facts. For instance, given the term 'tree', an ontology should know at least that some trees are biological organisms, and some are mathematical objects, and they are not the same.

There is an obvious trade-off between the number of concepts covered ('breadth') and the amount of information represented about each concept ('depth'), and almost all current ontology projects emphasize one without the other. For instance WordNet defines 207,000 categories but solely organizes them into a few simple relations. After initial enthusiasm for using WordNet as an ontology due to its simplicity (Mann, 2002; Niles et al., 2003), today it is still mainly appreciated as an exceptionally comprehensive linguistic resource.

Relatively sophisticated definitions of concepts in narrow domains may be found in a number of ontologies pertaining to specific subject-areas which attract research funding, e.g. the Foundation Model of Anatomy (Rosse and Mejino, 2003), and various geospatial ontologies. However, ontology integration is still an enormous challenge. Thus a cursory search for our example 'tree' on Swoogle,[1] which queries 10,000 ontologies, returns merely scattered unitary assertions (e.g. 'A Tree is a kind of LandscapeProduct', 'A TreeRing is a kind of Vegetation'), confusingly mixed with assertions concerning 'trees' as mathematical structures.

Arguably the most heroic attempt to provide breadth and depth simultaneously is the famous Cyc project (Lenat, 1995), the 'Rolls Royce of formal ontologies'. As a 20 year project, US government-funded for over 700 person-years of work, it has been able to amass 200,000 categories and provides a custom-built inference engine. Thus Cyc knows not only that `#$Tree-ThePlant` is different from `#$Tree-PathSystem`, but also assertions on the former such as, "A tree is largely made of wood", and, "If a tree is cut down, then it will be destroyed". Since 2004 sections of Cyc have been released to the public, such as OpenCyc, covering the top 40% of Cyc, and later ResearchCyc, covering over 80% (and available to research institutions). So far, however, their utilization and integration with other resources has been limited, as the combination of coding skills and philosophical nous required to understand and work with Cyc is still possessed by few.

Meanwhile vast excitement has gathered around the possibilities of leveraging websites with user-supplied content. The most general and fastest growing of these, Wikipedia, surprised many with its growth and accuracy. From its launch in early 2001 to the present it has swiftly acquired 2M concepts (indexed via 5M synonyms) and researchers soon began to mine its structure for ontology (e.g. Hepp et al., 2006; Herbelot et al., 2006). This provides a potential vast increase in concept-coverage. However if all that is

[1] http://swoogle.umbc.edu

taken from Wikipedia are names for concepts, arranged in a subsumption hierarchy via Wikipedia 'category' links, it risks becoming another WordNet – merely a 10 times bigger bag of words with no real understanding of their meaning. What is needed is some way of adding definitional information. One natural candidate for this is Cyc.

Given that Wikipedia has more concepts than Cyc, and Cyc has a richer explicitly represented knowledge framework than Wikipedia, it makes sense to integrate Wikipedia concepts into Cyc, rather than vice versa. The starting point for such integration is to establish mappings between existing Cyc terms and corresponding Wikipedia articles. To overcome terminology differences, we use rich synonymy relations in both resources. To deal with sense ambiguity, we analyze semantic similarity of possible mappings to context categories in the neighboring Cyc ontology. To bypass inconsistency in both resources, we develop a step-by-step mapping heuristic. With this strategy we can map 52,690 Cyc categories to Wikipedia articles, with a precision of 93% tested on 9,333 human alignments. Further disambiguation based on Cyc's common-sense knowledge improves the precision of 42,279 mappings to over 95%.[2]

At each mapped node in Cyc's tree, it may now be determined what new information Wikipedia can teach Cyc. So far, we have managed to identify over 145,000 possible new synonymy assertions, over 1,400 URLs, over 500,000 translations into other languages. We discuss the addition of further facts, which would produce an enlarged ontology, which may then be used for further iterative ontology alignment, including learning more facts from Wikipedia, which continues to grow and improve.

2. Related work

On the Cyc side, from its inception Cycorp sought to map existing ontologies and knowledge bases into Cyc, for instance WordNet (Reed et al. 2002). However having been automatically entered the new information required cleaning and integrating by hand, which due to limited resources was never done fully. Matuszek et al. (2005) extend Cyc by querying a search engine, parsing the results, and checking for consistency with the Cyc knowledge base. However, they required a human check before each new concept or fact was entered – thus only added 2,000 new assertions. The Cyc Foundation[3] is currently developing a user-friendly interface to integrate Cyc and Wikipedia. So far however (there are no published results yet), this appears to be merely a browser via which a human can look at Cyc and Wikipedia categories side by side, rather than any deeper integration.

On the Wikipedia side, mining for semantic relations is the prevalent research topic. Gregorowicz et al. (2006) treat Wikipedia as a semantic network, extracting hyperlinks between categories and articles, which are treated as 'semantic' but not further differentiated. Ponzetto and Strube (2006) categorize relations between Wikipedia categories by analyzing the names of concept pairs, their position in the network, as well as their occurrences in corpora, to accurately label the relation between each pair as *isa* and *not-isa*. However, there is yet no separation into further types of relations, such as *is-an-instance-of* and *has-part*. Also the approach is restricted to the 126,700 Wikipedia 'categories' (as opposed to the 2M Wikipedia articles). Thus these approaches are still far from producing full-blooded ontologies.

Several authors explicitly look at mining Wikipedia as an ontology. Hepp et al. (2006) use URIs of Wikipedia entries as identifiers for ontological concepts. This provides WordNet-style breadth without depth. Herbelot et al. (2006) extract an animal ontology from Wikipedia by parsing whole pages. Although they achieve an impressively comprehensive coverage of their subject-matter, computational demands restricted the task to only 12,200 Wikipedia pages, a tiny fraction of the total.

Suchanek et al. (2007) create a new ontology, YAGO, which unifies WordNet and Wikipedia providing 1M categories and 5M facts. Its categories are all WordNet synsets and all Wikipedia articles whose titles are not listed as common names in WordNet. It therefore misses many proper names with homonyms in WordNet—e.g. the programming language Python and the film "The Birds". Our approach differs from Yago in that we identify links between synonymous concepts in Cyc and Wikipedia using explicit semantic disambiguation, whereas Yago merely adds Wikipedia to WordNet avoiding the ambiguous items.

The DBpedia project attempts to make all structured information in Wikipedia freely available in database form (Auer et al., 2007). RDF triplets are extracted by mining formatting patterns in the text of Wikipedia articles, e.g. infoboxes, as well as categorization and other links. These authors harvest 103M facts and enable querying of their dataset via SPARQL and Linked Data. They also connect with other open datasets on the Web. But this enormous increase in data comes at a huge cost in quality. Many of the triplets' relations are not ontological but rather trivial, e.g. the most common relation in infobox triplets (over 10%) is `wikiPageUsesTemplate`. Also, amongst the relations that are ontological there are obvious redundancies not identified as such, e.g. `placeOfBirth` and `birthPlace`, `dateOfBirth` and `birthDate.`

3. Mapping of Cyc concepts to Wikipedia

The number of categories in our distribution of ResearchCyc (12/2007) is 163,317, however a significant portion of these do not represent common-sense knowledge. We therefore filtered out:

- categories describing Cyc's internal workings
- knowledge required for natural language parsing
- project-specific concepts
- microtheories
- predicates and all other instances of `#$Relation`

This leaves 83,897 categories.

[2] Available here: http://www.cs.waikato.ac.nz/~olena/cyc.html
[3] http://www.cycfoundation.org

We begin the integration of Cyc and Wikipedia by mapping Cyc concepts to Wikipedia *articles*. We do not allow mappings to Wikipedia's categories or disambiguation pages, because the former do not specifically describe concepts and the latter are inconsistent, however we do use disambiguation pages for identifying ambiguous terms (cf. Section 3.2). To Cyc terms we apply a simple cleaning algorithm to align them with Wikipedia article titles. This includes splitting the name into component words while considering the acronyms, e.g. `#$BirdOfPrey` → 'Bird of Prey'. Expressions after the dash sign in Cyc we write in brackets as this is the convention in Wikipedia, e.g. `#$Virgo-Constellation` → 'Virgo (constellation)'. We also map all but the first capitalized words to lower case, since Cyc does not distinguish between these. For example, in Wikipedia 'Optic nerve' (the nerve) and 'Optic Nerve' (the comic book) are distinct concepts; in Cyc the former is encoded as `#$OpticNerve` and the latter is missing.

Next, we differentiate between two cases: first, where a string comparison produces only one candidate Wikipedia article per Cyc term (exact mapping), and second, where it produces more than one (ambiguous mapping). For the former we propose two steps that augment each other, whereas for the latter we use two alternative approaches, which we evaluate individually.

3.1 Exact mappings

Mapping 1: We identify Cyc terms which exactly match Wikipedia article titles—or redirects, in which case the target article is retrieved. If the match is to a disambiguation page, the term is treated as ambiguous and not considered. The result is a set of possible mappings for each Cyc term. At this stage we only allow a mapping if this set contains exactly one member.

Mapping 2: If for a Cyc term Mapping 1 gives no results, we check whether its synonyms exactly match a title of a Wikipedia article, or its redirect. Again, only unitary result sets are allowed.

With this exact mapping we linked 33,481 of the chosen 83,897 Cyc terms to Wikipedia articles (40%).

3.2 Ambiguous mappings

While the above mappings ensure high accuracy (cf. Section 4.1), their coverage can be improved because many Cyc terms map to more than one Wikipedia article. Also, where no mappings were found, a less strict string comparison can improve the coverage. Therefore, before proceeding with disambiguation, we use the following conflation strategy. To each Cyc term and Wikipedia title, we apply case folding and remove brackets that specify the term's meaning (a feature used inconsistently in both resources). We do not use stemming, because most common syntactic variations are covered in either resource. We now begin to make use of links to articles on disambiguation pages as well. The set of candidate Wikipedia articles for each Cyc term now consists of:

- articles with matching titles
- articles linked from matching redirects,
- articles linked first in each disambiguation on matching disambiguation pages.

We additionally utilize anchor names (i.e. hyperlinked text) in Wikipedia as a source for synonyms (Mihalcea and Csomai, 2007). Given a search term *a*, the likelihood it will link to an article *T* is defined as

$$Commonness_{a,T} = P(T \mid a),$$

which is the number of Wikipedia articles where *a* links to *T* over the total number of articles linked from *a*. For example, the word *Jaguar* appears as a link anchor in Wikipedia 927 times. In 466 cases it links to the article *Jaguar cars*, thus the commonness of this mapping is 0.5. In 203 cases it links to the description of *Jaguar* as an animal, a commonness of 0.22.

Thus, given a Cyc term, we add to its candidate set the 5 most common Wikipedia articles, and record the most common link for each synonym of this term.

Disambiguation I: A simple disambiguation is to weight each candidate article by the number of times it has been chosen via the title of the Cyc term, or any of its synonyms. The highest weight indicates the ideal match.

Disambiguation II: Instead of relying on synonyms encoded in Cyc, this alternative disambiguation method is based on the semantic similarity of each candidate article to the *context* of the given Cyc concept. We define this context using the Cyc ontology, retrieving the categories immediately surrounding our candidate term with the following queries from Cyc's inference engine:

MIN-GENLS – direct hypernyms (collection→collection)
MAX-SPEC – direct hyponyms (collection→collection)
GENL-SIBLINGS – sister collections of a given collection.
MIN-ISA – direct hypernyms (instance→collection)
MAX-INSTANCES –direct hyponyms (collection→instance)
ISA-SIBLINGS – sister instances for a given instance.

We retrieve additional context terms via assertions on selected Cyc predicates, for instance `#$conceptuallyRelated`, and the geographic `#$countryOfCity`.

In Cyc, specifications of a term's meaning are often provided after a dash – e.g. `#$Tool-MusicGroup`, and `#$PCS-Corporation`. If such a specification is parsed and mapped to a Wikipedia article, it serves as a context term as well. For example, 'Music group' helps mapping `#$Tool-MusicGroup` to 'Tool (band)' in Wikipedia.

Next, for each context term obtained from Cyc, we identify a corresponding Wikipedia article with Mapping I and II (Section 3.2) or ignore it if it is ambiguous.[4] Given a set of candidate Wikipedia articles and a set of related context articles, we determine the candidate that is most semantically related to a given context (Milne and Witten, 2008). For each pair, candidate article *x* and context article *y*, we retrieve the sets of hyperlinks *X* and *Y* to these articles, and compute their overlap $X \cap Y$. Given the total number *N* of articles in Wikipedia, the similarity of *x* and *y* is:

[4] Although some important information is discarded by doing so, we find that usually sufficient non-ambiguous terms are provided.

$$SIM_{x,y} = 1 - \frac{\max(\log|X|, \log|Y|) - \log|X \cap Y|}{N - \min(\log|X|, \log|Y|)}.$$

For each article in the set of possible mappings, we compute its average similarity to the context articles. If for all candidates, no similarity to the given context is observed, we return the candidate with the highest commonness weight. Otherwise, we multiply the article *T*'s average similarity to the context articles by its commonness given the n-gram *a*:

$$Score(a,T) = \frac{\sum_{c \in C} SIM_{T,c}}{|C|} \times Commonness_{a,T},$$

where $c \in C$ are context articles for *T*. The article with the highest score is the best candidate.

With this method we cover an additional 19,209 Cyc terms (23%). This gives us the maximum coverage for the proposed mapping strategy, a total of 52,690 mappings, i.e. 62.8% of Cyc's common-sense knowledge. However, inaccuracies are inevitable. The following section describes how we address them.

3.3 Common-Sense Disambiguation

After mapping all Cyc terms to Wikipedia articles, we find cases where several Cyc terms map to the same article. (This is the reverse of the problem addressed by **Disambiguation I** and **II** above, where several Wikipedia articles map to the same Cyc term.) Analysis has shown that in some cases, the differentiation in Cyc is too specific, and both mappings are correct, e.g.`#$ThoracicVertebra` and `#$ThoracicVertebrae`. In other cases, one or more of the mappings are incorrect, e.g. `#$AlJazeerah-TheNewspaper`→'Al Jazeera' and `#$AlJazeera-MediaOrganizaton`→ 'Al Jazeera' – since Wikipedia describes the Al Jazeera TV network. Thus we perform two consecutive tests to further correct such mappings.

1. Similarity test.

First, we examine the semantic similarity score of each mapping. The best scoring mapping determines the minimum score for other mappings to be considered. A candidate is not considered if its score is over 30% lower than the maximum score. This helps to eliminate many unlikely mappings that were only 'found' because the Cyc concept has no equivalent Wikipedia article, or it was not located. For example, we eliminate `#$PCS-Corporation` →'Personal Computer' with a score 0.13, because it is lower than 1.58, the score of the best mapping: `#$PersonalComputer`→'Personal Computer'.

2. Disjointness test.

If the above test still leaves more than one possible mapping, we leverage Cyc's common sense knowledge about 'different kinds of things', represented in its extensive knowledge about *disjointness* of collections. We ask Cyc, whether two candidate Cyc terms (or in the case of individuals, their direct hypernyms) are disjoint. Any mapping which is disjoint with our highest scoring candidate is eliminated. All mappings for which disjointness can not be proven are retained. The following example lists Cyc terms mapped to article 'Casino' and their scores:

`#$Casino-Object`	1.1
`#$Casino-TheMovie`	1.0
`#$Casino-TheGame`	0.4
`#$Casino-Organization`	0.1

The similarity test leaves us with `#$Casino-Object` and `#$Casino-TheMovie`, where the former is more likely. But Cyc knows that a casino is a `#$SpatialThing` and a movie is an `#$AspatialThing` and the two are disjoint. Thus we only accept `#$Casino-Object`, which is the correct mapping. The philosophical purity of Cyc's ontology can produce some remarkable discriminations. For instance, Cyc distinguishes between `#$ValentinesCard` and `#$ValentinesDay` given that the former generalizes to `#$SpatialThing-NonSituational` and the latter to `#$Situation`.

Alternatively, the test allows both of these mappings:
`#$BlackPeppercorn`→'Black pepper'
`#$Pepper-TheSpice`→'Black pepper'

This is correct as the Wikipedia article, despite its title, is more general than both Cyc terms, explaining how the spice (both black and white) is produced from the peppercorns. The strategy does make some mistakes. For instance having decided that `#$Countess`→'Count' has greater semantic similarity than `#$Count-Nobleman` →'Count', the method then proceeds to reject `#$Count-Nobleman` (which would in fact be a better match) because Cyc's collections of females and males are disjoint.

With this strategy we eliminate approximately 10K mappings, which gives us a total of 42,279 – 50% of the original 83,897. Next we evaluate, whether the precision of these mappings is improved.

4. Evaluation

We evaluate the proposed methods using two data sets. The first (Testset1), kindly offered to us by the Cyc Foundation, contains 9,436 synonymous mappings between Cyc categories and Wikipedia articles – created semi-automatically by one person. Evaluation is made more difficult by the fact that at times more than one answer can be correct (e.g. `#$BabyCarrier` can be mapped to either 'Baby sling' or 'Child carrier'). Therefore we also investigate human inter-agreement on the mapping task by giving a new set (Testset2) with 100 random Cyc terms to 6 human subjects. The goal of the algorithm is to achieve as high agreement with the subjects as they with each other.

4.1 Results for Testset1

Out of 9,436 examples in the first data set, we exclude

	Found	Correct	P	R	F
Mapping I	4655	4477	96.2	48.0	64.0
Mapping I & II	6354	5969	93.9	64.0	76.1

Table 1. Results for non-ambiguous mappings in Testset1: precision (%), recall (%), F-Measure (%).

	Before common-sense disambiguation					After common-sense disambiguation				
	Found	Correct	P	R	F	Found	Correct	P	R	F
Synonym-based	8884	7958	89.6	85.3	87.4	7715	7022	91.0	72.5	82.4
Context-based	8657	8054	93.0	**86.3**	**89.5**	7763	7386	**95.1**	79.1	86.4

Table 2. Results for disambiguated mappings in Testset1: precision (%), recall (%), F-Measure (%).

those that link to Wikipedia categories or particular parts of Wikipedia articles. Tables 1 and 2 investigate mapping of the remaining 9,333. Our Mapping I alone covers 4,655 examples, out of which 4,477 are correct. Moreover, manual examination of the 'incorrect' mappings reveals that their vast majority is actually correct—our method often identified more precise mappings than the given ones, e.g.:

- `#$Plumage` → 'Plumage' instead of 'Feather'
- `#$TransportAircraft` → 'Transport aircraft' instead of 'Cargo aircraft'

By including synonyms listed in Cyc in Mapping II, we found an additional 1,699 mappings with 1,492 correct according to the test set (precision 87.8%). Here often 'incorrect' mappings occur because the meaning is too close. For example, the Cyc term `#$SacAndFoxLanguage` was mapped to 'Fox (tribe)', via Cyc's synonym *sac and fox*. which in Wikipedia means the tribe. However, in the majority of cases this strategy worked well, e.g. `#$AeolicGreekDialect` → *aeolic greek* → 'Aeolic Greek'.

The last row of Table 1 summarizes the results of Mappings I and II combined. We covered 68% of the test set with precision of almost 94%. The remaining 32% of the test set, 2,979 terms, are either difficult to find or ambiguous. With the stronger conflation strategies (cf. Section 3.2), we identify an additional bulk of terms with at least one mapping and disambiguate them to Wikipedia articles with our two methods: synonym-matching vs. context-based similarity. We additionally evaluate, whether common-sense disambiguation (Section 3.3) improved the accuracy as expected. Table 2 compares the performance of the algorithm under each setting, giving the overall results, when disambiguation is combined with Mapping I and II.

Context-based disambiguation clearly outperforms the synonym-based approach and achieves maximum precision of 95.1%, when the disjointness test is used. The best recall, 86.3%, is achieved without the common-sense disambiguation, however the precision is more than 2 points lower. There is an obvious trade-off between precision and recall, and for some applications one could be more important than the other.

Manual analysis of errors shows different reasons for incorrect mappings, e.g. inaccuracies in Wikipedia, errors in the test set, insufficient context, very close meanings, or inconsistencies in Cyc. For instance, insufficient context led to erroneous mapping `#$AnticommunistIdeology`→'Communism', because it is more common than 'Anti-Communism'. Sometimes, very similar meanings could not be differentiated, e.g. `#$CityOfKyotoJapan` and `#$Kyoto-PrefectureJapan` are both mapped to 'Kyoto'. Both pages have high similarity with their context, whereas 'Kyoto Prefecture' is less a common page. Treating specification after the dash sign as context and not as a part of the title, results in `#$Tea-Iced`→'Tea' instead of 'Iced tea.' This is an example of inconsistency in Cyc.

	Agreement with other subjects	Agreement with algorithm before final disambiguation	after final disambiguation
Subject 1	37.6	34.0	28.0
Subject 2	40.4	41.0	31.0
Subject 3	40.8	40.0	29.0
Subject 4	40.8	41.0	30.0
Subject 5	42.4	44.0	32.0
Subject 6	37.0	35.0	28.0
Overall	39.8	39.2	29.7

Table 3. Results for the final mapping algorithm on Testset2.

4.2 Results for Testset2

We created a second test set with 100 random Cyc categories, which six human subjects independently mapped to a Wikipedia articles. The instructions were to map only if both resources define the same concept, with the aid of the Wikipedia search function.

Interestingly, the number of mapped concepts varied across the subjects. All agreed that there is no mapping in only 22 cases. On average they mapped 56 Cyc terms, ranging from 47 to 65. The algorithm was again tested with and without the common-sense disambiguation, where the former mapped 58 and the latter only 39 terms. Note that the creator of Testeset1 did not include 'mappings' where a Cyc term had no corresponding Wikipedia article, whereas Testset2 was created randomly from all common-sense terms in Cyc. This is why both humans and the algorithm have lower coverage on this set.

To compute the agreement we compared mapped concepts between each pair of human subjects, and between each human subject and our algorithm. Table 3 summarizes the results. The overall agreement between our subjects is 39.8%. The surprisingly low coverage of the algorithm, when common-sense disambiguation is applied, results in very low agreement on this data set of only 29.7%. However, without this test the algorithm performs nearly as well as the human subjects (39.2%). In fact, it outperforms Subjects 1 and 6.

Error analysis shows that in some cases the algorithm picked a more general article than the humans, e.g. `#$Crop`→'Agriculture', instead of 'Crop (agriculture)', picked by all subjects, or `#$StarTrek-GameProgram` →'Star Trek', instead of 'Star Trek Games', as identified by one subject, or 'Star Trek Generation (video game)', by another, while the others failed to produce any mapping. In a few cases, the algorithm identified a mapping where most humans failed, e.g. `#$BurmesePerson`→'Bamar'.

5. Adding new information to Cyc

Now that the alignment has been performed, could any new information be added to Cyc from Wikipedia?

Synonyms: Despite the extensive work on its natural language interface, Cyc is weak at identifying its concepts. For instance, typing "Casino" into Cyc's search function does not retrieve `#$Casino-TheMovie`. Given 42,279 more accurate mappings, we can retrieve over 154,800 synonyms from Wikipedia (≈2.6 per term), of which only 8,390 are known by Cyc.

Translations: Currently, there are over 200 different language versions of Wikipedia. 15 versions have over 100,000 articles, and 75 have at least 10,000. We have estimated that given 42,836 mappings of Cyc terms to Wikipedia articles, we can retrieve over 500,000 translations of these terms in other languages, which is about 13 per term.

Glosses: For each mapping we can retrieve the first paragraph of Wikipedia's article, which would enrich Cyc's hand-written `#$comment` on that term.

URL Resources: Using triplets in DBpedia's infobox dump, we identified 1,475 links to URLs corresponding to the Wikipedia concepts that we have mapped to Cyc.

New Relations: Many other relations in the DBpedia dataset bear a significant similarity to Cyc predicates, e.g.:

```
keyPeople ↔ #$keyGroupMembers
capital ↔ #$capitalCity
```

However manual analysis has shown that much of the dumped data is of poor quality (e.g. `keyPeople` assertions of the form "CEO" or "Bob", `capital` assertions which name districts rather than cities). Much however could be done to automatically quality-control candidate assertions using Cyc's ontological constraints on the arguments of its predicates – thus for instance as Cyc knows that the first argument to `#$capitalCity` must be a `#$City`, it can reject the claim that the capital of Bahrain is `#$AlManamahDistrict`. We will explore this in future work.

6. Conclusions

We map 52,690 Cyc terms to Wikipedia articles, with a precision of 93%. Evaluation shows that this mapping technique achieves the same agreement with 6 human subjects as they do with each other. We also show how more accurate results can be achieved using Cyc's commonsense knowledge.

Our work opens up considerable possibilities for further enriching Cyc's ontological rigor with Wikipedia's folksonomic bounty.

Acknowledgements

We thank Mark Baltzegar, Sudarshan Palliyil, Tim Wilmot-Sitwell, Glen Meyer, Kendall Lister, Brett Summers, Marios Daoutis and Cycorp Inc. for their generous help with this research.

References

Auer, S.; Bizer, C.; Kobilarov, G.; and Lehmann, C. et al. 2007. DBPedia: A nucleus for a Web of open data. Aberer, K. et al (eds.) *ISWC/ASWC 2007, LNCS 4825*. Springer-Verlag, Berlin Heidelberg, pp. 722-35.

Gregorowicz, A.; and Kramer, M.A. 2006 Mining a large-scale term-concept network from Wikipedia. *Tech. report, The MITRE Corporation*.

Gruber, T.R. 1995. Toward principles for the design of ontologies used for knowledge-sharing. *Int. Journal of Human and Computer Studies,* 43 (5/6), 907-28.

Guarino, N. (1998). Formal ontology and information systems. *Proc. FOIS-98*, Trento, Italy.

Hepp, M.; Bachlechner, D.; and Siorpaes, K. 2006. Harvesting Wiki Consensus-Using Wikipedia Entries as Ontology. *Proc. ESWC-06 Workshop on Semantic Wikis*, pp.132-46.

Herbelot, A.; and Copestake, A. 2006. Acquiring Ontological Relationships from Wikipedia Using RMRS. *Proc. ISWC-06 Workshop on Web Content Mining with Human Language.*

Legg, C. 2007. Ontologies on the Semantic Web. *Annual Review of Information Science and Technology* 41, 407-52.

Lenat, D.B. 1995. Cyc: A Large-Scale Investment in Knowledge Infrastructure. *Communic. of the ACM* 38 (11).

Mann, G.S. 2002. Fine-grained Proper Noun Ontologies for Question Answering. *Proc. ICCL-02, SEMANET: Building and Using Semantic Networks*, Vol. 11, 1-7.

Matuszek, C.; Witbrock, M.; Kahlert, R.C. et al. 2005. Searching for Common Sense: Populating Cyc from the Web. *Proc. AAAI-05*. Pittsburgh, Penn., pp.1430-1435.

Milne, D.; and Witten, I.H. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. *Proc. AAAI-08 Workshop Wikipedia and the AI.*

Mihalcea, R.; and Csomai, A. 2007. Wikify!: Linking documents to encyclopedic knowledge. *Proc. CIKM-07*, pp. 233-242.

Niles, I.; Pease, A. 2003. Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. *Proc. IEEE IKE-03*, pp. 412-416

Reed, S.; and Lenat, D.B. 2002. Mapping ontologies into Cyc. *Proc. AAAI Workshop Ontologies for the Semantic Web*, Edmonton, Canada.

Rosse C.; and Mejino J.V.L. 2003. A reference ontology for biomedical informatics: the Foundational Model of Anatomy. *J Biomed Inform*. 36:478-500.

Ponzetto, S. P.; and Strube, M. 2007. Deriving a Large Scale Taxonomy from Wikipedia. *Proc. of AAAI-07*, pp. 1440-1445.

Topic Indexing with Wikipedia

Olena Medelyan, Ian H. Witten and David Milne

Computer Science Department
University of Waikato
New Zealand
{olena, ihw, dnk2}@cs.waikato.ac.nz

Abstract

Wikipedia can be utilized as a controlled vocabulary for identifying the main topics in a document, with article titles serving as index terms and redirect titles as their synonyms. Wikipedia contains over 4M such titles covering the terminology of nearly any document collection. This permits controlled indexing in the absence of manually created vocabularies. We combine state-of-the-art strategies for automatic controlled indexing with Wikipedia's unique property—a richly hyperlinked encyclopedia. We evaluate the scheme by comparing automatically assigned topics with those chosen manually by human indexers. Analysis of indexing consistency shows that our algorithm performs as well as the average person.

1. Introduction

The main topics of a document often indicate whether or not it is worth reading. In libraries of yore, professional human indexers were employed to manually categorize documents, and the result was offered to users along with other metadata. However, the explosion of information has made it infeasible to sustain such a labor-intensive process.

Automated indexing has been investigated from various angles. *Keyphrase extraction* weights word n-grams or syntactic phrases that appear in a document according to their statistical properties. The resulting index terms are restricted to phrases that occur in the document, and are prone to error because semantic relations are ignored. *Term assignment* uses text classification to create a model for every topic against which new documents are compared; but this needs a huge volume of training data. The inaccuracy of keyphrase extraction and the impracticality of term assignment have stimulated a new method, *keyphrase indexing*, which maps document phrases onto related terms of a controlled vocabulary that do not necessarily appear verbatim, and weights terms based on certain features. Problems of ambiguity and the need for a manually created vocabulary restrict this technique to narrow domains.

The online encyclopedia Wikipedia is tantamount to a huge controlled vocabulary whose structure and features resemble those of thesauri, which are commonly used as indexing vocabularies (Milne *et al*. 2006). As Figure 1 illustrates, the titles of Wikipedia articles (and redirects) correspond to terms. Its extensive coverage makes Wikipedia applicable to nearly any domain. However, its vast size creates new challenges.

This paper shows how Wikipedia can be utilized effectively for topical indexing. The scheme is evaluated on a set of 20 computer science articles, indexed by 15 teams of computer science students working independently, two per team. The automatic approach reaches the average performance of these teams, and needs very little training.

2. Related work

One of the largest controlled vocabularies used for indexing is the Medical Subject Heading (MeSH) thesaurus. It contains 25,000 concepts and has been applied to both term assignment and keyphrase indexing, individually and in combination. Markó *et al*. (2004) decompose document phrases into morphemes with a manually created dictionary and associate them with MeSH terms assigned to the documents. After training on 35,000 abstracts they assign MeSH terms to unseen documents with precision and recall of around 30% for the top 10 terms. However, only concepts that appear in the training data can be assigned to new documents, and the training corpus must be large.

Aronson *et al.* (2000) decompose candidate phrases into letter trigrams and use vector similarity to map them to concepts in the UMLS thesaurus. The UMLS structure allows these concepts to be converted to MeSH terms. The candidates are augmented by additional MeSH terms from the 100 closest documents in the manually indexed PubMed collection, and the terms are heuristically weighted. An experiment with 500 full text documents achieved 60% recall and 31% precision for the top 25 terms (Gay *et al.*, 2005). However, the process seems to involve the entire PubMed corpus, millions of manually indexed documents.

The key challenge is overcoming terminological differences between source documents and vocabulary terms. Wikipedia, with 2M articles and over 2M synonyms ("redirects"), extensively addresses spelling variations, grammatical variants and synonymy. The 4.7M anchor links offer additional clues to how human contributors refer to articles.

A second issue is the need for large amounts of training data in both the systems mentioned above. In contrast, Medelyan and Witten (2008) achieve good results with

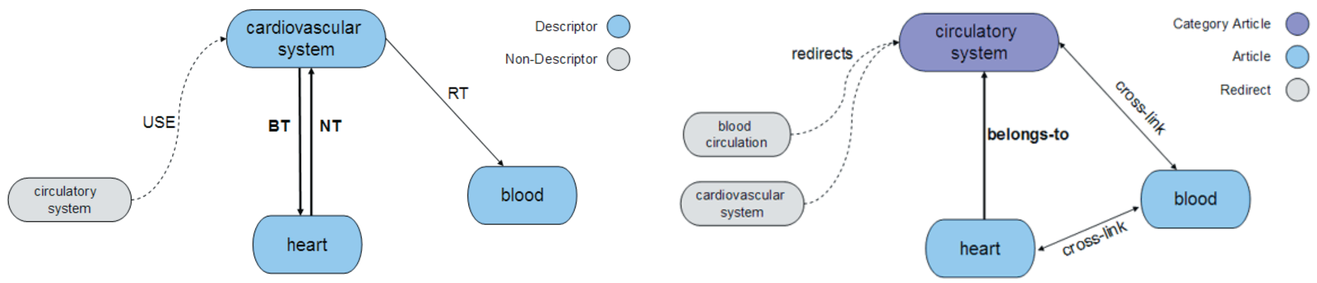


Figure 1. Excerpts from manually created Agrovoc thesaurus and the corresponding structure from Wikipedia

fewer than 100 training documents by learning typical properties of manually assigned terms in general, instead of associations between particular index terms and document phrases. To ensure semantic conflation they use synonymy links encoded in a manual thesaurus. Each candidate phrase is characterized by several features (see Section 3.4 below). A Naïve Bayes scheme is used to learn a model which is applied to unseen documents. Performance improves if "degree of correctness" data is available from multiple indexers: use the number of indexers who choose a term as a keyphrase instead of whether or not one indexer has been chosen it. The method yields 32% consistency with professional indexers, compared with a figure of 39% for human indexers. It is domain-independent but requires a manually created controlled vocabulary.

In this paper we distil a controlled vocabulary automatically from Wikipedia. Wikipedia has been used for similar tasks before. Gabrilovich and Markovich (2007) improve text classification by adding information from it to the bag-of-words document model. They build a vector space model of all Wikipedia articles, and, before classifying a new document, site it in the space and add the most similar articles' titles as new features. However, documents are classified into only a few hundred categories, whereas we treat every Wikipedia article title as a potential index term.

Mihalcea and Csomai (2007) describe the similar problem of "wikification". A document is "wikified" by linking it to Wikipedia articles, to emulate how Wikipedia articles cross-reference each other. For each n-gram that appears in Wikipedia they pre-compute the probability of it being a link—they call this its "keyphraseness." Then all phrases in a new document whose keyphraseness exceeds a certain threshold are chosen as keyphrases.

Their usage of the term "keyphrase" diverges from the conventional meaning. Keyphrases are terms that describe the main topics of a document; they describe concepts. Mihalcea and Csomai compute keyphraseness as property of n-grams rather than concepts. Furthermore, they compute it over the entire Wikipedia corpus: thus keyphraseness in their sense reflects the significance of a phrase for the document collection as a whole rather than for an individual document. For instance, the descriptor *Java (Programming language)* is more topical in a document that covers aspects of this language than in one that explains an algorithm that happens to be written in Java. Previously, to identify a document's topics, an analog of keyphraseness has been combined with document-specific features (Frank *et al*. 1999). We extend this to use the Wikipedia version of keyphraseness.

3. Indexing with Wikipedia as a controlled vocabulary

We follow the basic two-stage structure of most keyphrase indexing algorithms: first select many candidate terms for a document and then filter out all but the most promising. In keyphrase *extraction* candidates are plain document phrases, while in keyphrase *indexing* they are descriptors from the controlled vocabulary. We use Wikipedia articles as candidates and their titles as index terms. Figure 1 illustrates this: descriptors on the left map into articles on the right.

3.1 Selecting candidates

The *candidate selection* step extracts word n-grams from the document and matches them with terms in a controlled vocabulary. Current systems work within a particular domain and use a domain-specific vocabulary. Moving outside a specific domain by using a general controlled vocabulary presents significant difficulties. As noted earlier, we use Wikipedia article titles as index terms: a total vocabulary of 2M, along with a further 2M synonyms (i.e. redirect titles). Almost every document phrase can be mapped to at least one article; most phrases map to several. It is essential for success to avoid unnecessary mappings by disambiguating the word senses.

We perform candidate selection in two stages:

- What words and phrases are important?
- Which Wikipedia articles do they relate to?

The first stage excludes words that contribute little to identifying the document's topics—that is, words that can be changed without affecting the topics expressed. We adapt the "keyphraseness" feature and choose as candidates all phrases for which this exceeds a predefined threshold. Earlier, Frank *et al.* (1999) computed an analogous metric from a manually indexed corpus—but it had to be large to cover all sensible domain terminology. With Wikipedia this feature is defined within the vocabulary itself.

The second stage links each candidate phrase to a Wikipedia article that captures its meaning. Of course, the ambiguity of language and its wealth of synonyms are both reflected in Wikipedia, so word-sense disambiguation is necessary. For example, the word *tree* in a document about depth-first search should be linked to the article *Tree (Data structure)* rather than to any biological tree.

Mihalcea and Csomai (2007) analyze link annotations in Wikipedia. If the candidate *bar* appears in links annotated as [[bar (law)|bar]] and [[bar (establishment)|bar]], the two Wikipedia articles *Bar (law)* and *Bar (establishment)* are possible targets. The weakness of this technique is that links are often made to hyponyms or instances rather than synonyms of the anchor text. For example, the anchor *king* has 371 destinations, the majority of which are specific kings. We avoid these irrelevant senses with a more accurate technique, where n-grams are matched against titles of Wikipedia articles and their redirects.

If more than one article relates to a given n-gram, the next step is to disambiguate the n-gram's meaning. Mihalcea and Csomai investigate two approaches. Their *data-driven* method extracts local and topical features from the ambiguous n-gram, such as part-of-speech and context words, and computes the most probable mapping based on the distribution of these features in the training data. Their *knowledge-based* method computes the overlap of the paragraph in which the n-gram appears with the opening paragraph of the Wikipedia article. The first method is computationally challenging, requiring the entire Wikipedia corpus for training. The second performs significantly worse than a baseline that simply chooses the most likely mapping. We use a new disambiguation technique based on similarity of possible articles to context articles mined from the surrounding text. This is described in detail in the next section, and justified in Section 3.1.2.

3.1.1 Details of the candidate selection method

To identify important words and phrases in a document we first extract all word n-grams. For each n-gram *a*, we—like Mihalcea and Csomai—compute its probability of being a candidate (in other words, its keyphraseness) as follows:

$$Keyphraseness(a) \approx \frac{count(D_{Link})}{count(D_a)}$$

Here, $count(D_{Link})$ is the number of Wikipedia articles in which this n-gram appears as a link, and $count(D_a)$ is the total number of articles in which it appears.

The next step is to identify the articles corresponding to each candidate. Both Wikipedia titles and n-grams are case-folded, and parenthetical text (e.g *law* and *establisment* in the *bar* example given previously) is removed from the former. N-grams and titles can then be compared, so that matching articles are used as senses, as are the targets of matching redirects. From matching disambiguation pages we add all articles listed as meanings in the first position of each explanation.

This results in a set of possible article mappings for each significant n-gram. Articles for unambiguous n-grams (those with only one match) are collected and used to disambiguate the n-grams with more than one mapping. For this, we compute the average semantic similarity of each candidate article to all context articles identified for a given document. The semantic similarity of a pair of articles is computed from the links they make (Milne and Witten, 2008). For each pair of articles *x* and *y* we retrieve the sets of articles *X* and *Y* which link to them, and compute their overlap $X \cap Y$. Given the total number *N* of articles in Wikipedia, the similarity of *x* and *y* is:

$$SIM_{x,y} = 1 - \frac{\max(\log|X|, \log|Y|) - \log|X \cap Y|}{N - \min(\log|X|, \log|Y|)} .$$

Our disambiguation approach takes into account both this relatedness to context and the *commonness* of each sense: the extent to which they are well-known. The commonness of a sense (or article) *T* for an anchor (or n-gram) *a* is defined as:

$$Commonness_{a,T} = P(T \mid a) .$$

For example, the word *Jaguar* appears as a link anchor in Wikipedia 927 times. In 466 cases it links to the article *Jaguar cars*, thus the commonness of this mapping is 0.5. In 203 cases it links to the description of *Jaguar* as an animal, a commonness of 0.22. Mihalcea and Csomai (2007) use this information for one of their baselines, but seem to ignore it in the disambiguation process.

Finally, we multiply the article *T*'s average similarity to the context articles by its commonness given the n-gram *a*:

$$Score(a,T) = \frac{\sum_{c \in C} SIM_{T,c}}{|C|} \times Commonness_{a,T} ,$$

where $c \in C$ are the context articles for *T*. The highest-scoring article is chosen as the candidate term for the n-gram *a*.

3.1.2 Evaluation of candidate selection

To evaluate our disambiguation method we chose 100 random Wikipedia articles and used their manually annotated content as test documents. We iterate over the links in these articles, and use the above strategy to disambiguate them to Wikipedia articles. Table 1 compares the results with two baselines. The first one chooses an article at random from the set of candidate mappings. The second chooses the article whose commonness value is greatest. The results demonstrate that the new similarity-based disambiguation method covers almost as many candidates as the baselines (17,416 vs. 17,640) and is significantly more accurate than both, achieving an F-Measure of nearly 93%.

The baseline of choosing the most common sense provides a useful point of comparison between our disambiguation approach and Mihalcea and Csomai's work. Their knowledge-based approach performs significantly worse than this baseline, while ours is significantly better. Admittedly the comparison involves different versions of Wikipedia, but it seems unlikely that the previous approach would improve enough over the new data to outperform both the baseline and our technique. Instead it is more likely to degrade, since the task gets more difficult over time as more senses are added to Wikipedia. Section 5.2 contains further evaluation of our technique based on multiple-indexer data.

	A	C	P	R	F
Random	17,640	8,651	45.8	45.7	45.8
Most common	17,640	15,886	90.6	90.4	90.5
Similarity-based	17,416	16,220	93.3	92.3	92.9

Table 1. Disambiguation results: Attempted, Correct, Precision (%), Recall (%), F-measure (%)

3.2 Filtering

The *candidate selection* step is followed by a *filtering* step that characterizes each candidate term by statistical and semantic properties ("features") and determines the final score using a machine learning algorithm that calculates the importance of each feature from training data.

Earlier indexing schemes use features such as occurrence frequency, position in the document and keyphrase frequency (Frank *et al.* 1999). We adopt the first two and modify the third one to use "keyphraseness" (Feature 5 in Section 3.2.1). Furthermore, it is known that performance improves significantly if semantic relatedness of candidate phrases is taken into account (Turney, 2003; Medelyan and Witten, 2008). Although Wikipedia does not define semantic relations, articles can be seen as related if they contain many mutual hyperlinks (Milne and Witten, 2008).

3.2.1 Features for learning

For any given document, the candidate selection stage yields a list of Wikipedia article titles—terms—that describe the important concepts it mentions. Each term has a frequency that is the number of n-gram occurrences in the document that were mapped to it. Following earlier researchers (Frank *et al.* 1999; Turney, 2003; Medelyan and Witten, 2008), we define several features that indicate significance of a candidate term *T* in a document *D*.

1. **TF×IDF** = $\frac{\text{freq}(T,D)}{\text{size}(D)} \times -\log_2 \frac{\text{count}(T)}{N}$,

This compares the frequency of a term in the document with its occurrence in general use. Here, freq(*T*,*D*) is term *T*'s occurrence count in document *D*, size(*D*) is *D*'s word count, count(*T*) is the number of documents containing *T* in the training corpus, and *N* is the size of the corpus.

2. **Position of first occurrence** of *T* in *D*, measured in words and normalized by *D*'s word count. Phrases with extreme (high or low) values are more likely to be valid index terms because they appear either in the opening or closing parts of the document.

3. **Length** of *T* in words. Experiments have indicated that human indexers may prefer to assign multi-word terms.

4. **Node degree**, or how richly *T* is connected through thesaurus links to others that occur in the document. We define the degree of the Wikipedia article *T* as the number of hyperlinks that connect it to other articles in Wikipedia that have also been identified as candidate terms for the document. A document that describes a particular topic will cover many related concepts, so candidate articles with high node degree are more likely to be significant.

5. **Total keyphraseness**. For each candidate term *T* we define the document's *total keyphraseness* to be the sum of keyphraseness values for all unique n-grams *a* that were mapped to this term, times their document frequency:

$$total\_keyphraseness(T) = \sum_{a \Rightarrow T} keyphraseness(a) \times freq(a)$$

3.2.2 Using the features to identify the index terms

Given these features, a model is built from training data—that is, documents to which terms have been manually assigned. For each training document, candidate terms are identified and their feature values calculated. Because our data is independently indexed by several humans, we assign a "degree of correctness" to each candidate. This is the number of human indexers who have chosen the term divided by the total number of indexers: thus a term chosen by 3 out of 6 indexers receives the value 0.5.

From the training data, the learning algorithm creates a model from that predicts the class from the feature values. We use the Naïve Bayes classifier in WEKA (Witten and Frank, 2005). To deal with non-standard distributions of the feature values, we apply John and Langley's (1995) kernel estimation procedure.

To identify topics for a new document, all its terms (i.e., candidate articles) and their feature values are determined. The model built during training is applied to determine the overall probability that each candidate is an index term, and those with the greatest probabilities are selected.

4. Evaluation

Topic indexing is usually evaluated by asking two or more human indexers to assign topics to the same set of test documents. The higher their consistency with each other, the greater the quality of indexing (Rolling, 1981). Of course, indexing is subjective and consistency is seldom high. To reliably evaluate an automatic scheme it should be compared against several indexers, not just one—the goal being to achieve the same consistency with the group as group members achieve with one another.

4.1 Experimental data

We chose 20 technical research reports covering different aspects of computer science. Fifteen teams of senior computer science undergraduates independently assigned topics to each report using Wikipedia article names as the allowed vocabulary. Each team had two members who worked together in two 1½ hour sessions, striving to achieve high indexing consistency with the other teams; no collaboration was allowed. Teams were instructed to assign around 5 terms to each document; on average they assigned 5.7 terms. Each document received 35.5 different terms, so the overlap between teams was low.

We analyzed the group's performance using a standard measure of inter-indexer consistency:

$$Consistency = \frac{2C}{A+B}$$

Team ID	English?	Year	Consistency (%)
1	no	4.5	21.4
2	no	1	24.1
3	no	4	26.2
4	no	2.5	28.7
5	yes	4	30.2
6	mixed	4	30.8
7	yes	3	31.0
8	no	3	31.2
9	yes	4	31.6
10	yes	3.5	31.6
11	yes	4	31.6
12	mixed	3	32.4
13	yes	4	33.8
14	mixed	4	35.5
15	yes	4	37.1
		overall	**30.5**

Table 2. Consistency of each team with the others

where *A* and *B* are the total number of terms two indexers assign and *C* is the number they have in common (Rolling, 1981). This measure is equivalent to the F-measure, as well as to the Kappa statistic for indexing with very large vocabularies (Hripcsak and Rothschild, 2005).

Table 2 shows the consistency of each team with the other 14. It also indicates whether team members are native English speakers, foreign students, or mixed, and gives the average study year of team members. Consistency ranges from 21.1% to 37.1% with an average of 30.5%. In a similar experiment professional indexers achieved a consistency of 39% (Medelyan and Witten, 2008); however the vocabulary was far smaller (28,000 vs. 2M concepts).

4.2 Results

We first evaluate the performance of candidate selection, a crucial step in the indexing process that involves both phrase selection and word sense disambiguation. How many of the Wikipedia articles that people chose for each document are identified as candidates?

Table 3 shows the coverage of all manually chosen terms (Recall R). It also shows those that were chosen by at least 3 humans (best Recall, Rb), which we view as more important. The rows compare two disambiguation techniques: a simple one that chooses the most common sense, and the similarity-based approach.

The results are shown for extracting n-grams with keyphraseness exceeding 0.01, which covers a reasonable number of manually assigned terms (i.e. Wikipedia articles) and provides a sufficient number of context articles. An average of 473 candidate terms are identified for each document. The *similarity-based* disambiguation algorithm locates 78% of the terms chosen by at least 3 human indexers, 4.3 percentage points better than the *most common* baseline. Improvement in total recall is only 2.5 points, which indicates that the terms chosen by more indexers are more ambiguous, for example: *Tree (data structure), Inheritance (compute science), Index (search engine).*

	# terms per doc	P	R	Rb
most common	388	5.1	52.5	73.8
similarity-based	473	5.6	55.0	78.1

Table 3. Candidate selection results: Precision, Recall, best Recall (Rb) (%)

		Consistency (%)		
	Method	min	avg	max
1	human indexers	20.3	**30.5**	**38.4**
2	TF×IDF baseline	10.9	17.5	23.5
3	ML with 4 features	20.0	25.5	29.6
4	total keyphraseness	22.5	27.5	32.1
5	ML with 5 features	**24.5**	**30.5**	36.1

Table 4. Performance compared to human indexers

Table 4 compares the performance of the filtering technique of Section 3.2 with the index terms assigned by 15 human teams. As a baseline we extract for each document 5 terms with the highest TF×IDF values (row 2). This achieves an average consistency with humans of 17.5%. Next we evaluate the filtering strategy based on features previously used for automatic indexing: features 1–4 of Section 3.4 (row 3). We use “leave-one-out” evaluation, i.e. train on 19 documents and test on the remaining one, and repeat until all documents have been indexed. The result, 25.5%, is 8 points above the TF×IDF baseline.

Now we evaluate *total keyphraseness* (feature 5 in Section 3.4) (row 4). The consistency of the top 5 candidate terms is 27.5%, only 3 points less than consistency among humans. Finally we combine *total keyphraseness* with the other 4 features, bringing the average consistency to 30.5% (row 5). This is the same as the average over the 15 human teams (Table 2). The new method outperforms 5 teams, all in their 4th year of study in the same area as the test documents; one team consists of two English native speakers. These results are achieved after learning from only 19 manually indexed documents.

4.3 Examples

Figure 2 illustrates the terms assigned by humans (open circles) and our algorithm (filled circles). The 6 best human teams are shown in different colors; other teams are in black. Arrows between nodes show hyperlinks in the corresponding Wikipedia articles, and indicate the semantic relatedness of these concepts. The behavior of the algorithm is indistinguishable from that of the student teams.[1]

5. Conclusions

This paper combines research on linking textual documents into Wikipedia (Mihalcea and Csomai, 2007) with research on domain-specific topic indexing (Medelyan and Witten, 2008). We treat Wikipedia articles as topics and their titles as controlled terms, or descriptors.

[1] See *http://www.cs.waikato.ac.nz/~olena/wikipedia.html* for full results.

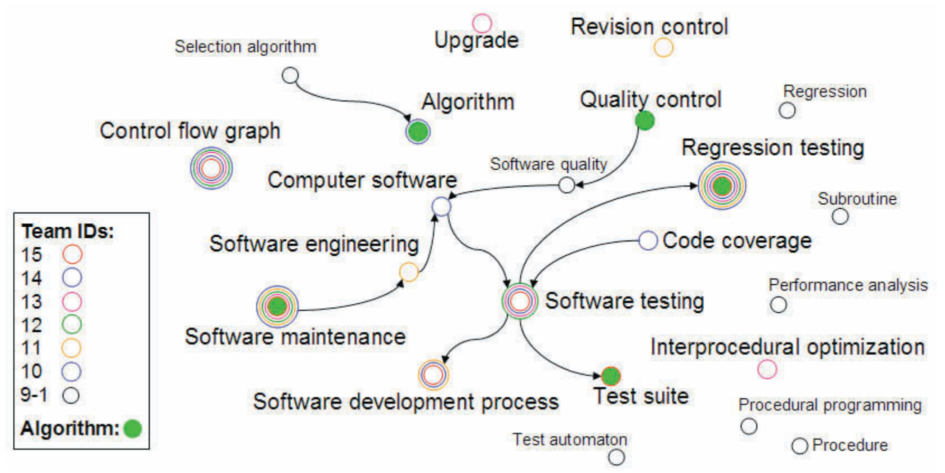


Figure 2. Topics assigned to a document entitled "A Safe, Efficient Regression Test Selection Technique" by human teams (outlined circles) and the new algorithm (filled circles)

We first link all important phrases in a document to Wikipedia articles by matching them to titles of articles, redirects and disambiguation pages. When multiple mappings exist, we apply an unsupervised disambiguation procedure based on semantic similarity.

Next, we restrict all linked Wikipedia articles to a handful of significant ones representing the document's main topics. One technique utilizes the knowledge in Wikipedia; a second uses training data to learn the distribution of properties typical for manually assigned topics. Evaluation on computer science reports indexed by human indexers shows that the former technique outperforms the latter, and a combination of the two yields the best results. The final approach has the same consistency with the 15 human teams as their average consistency with themselves.

Note that this performance is achieved with a very small training set of 19 documents, with 15 keyphrase sets each. Our new algorithm for efficient indexing with Wikipedia can assign topics to documents in nearly any domain and language, and we plan to capitalize on this by applying it to the multiply-indexed documents on social bookmarking sites like *del.icio.us* and *citeulike.org*.

Acknowledgements

We thank the students in our department who participated in the indexing competition, Andras Csomai for providing his data, and Eibe Frank for valuable comments.

References

Aronson, A. R.; Bodenreider, O.; Chang, H. F. et al. 2000. The NLM indexing initiative. *Proc. Fall Symp. of the American Medical Informatics Association*, LA, pp. 17–21.

Frank, E.; Paynter, G.W.; Witten, I.H.; Gutwin, C.; and Nevill-Manning, C.G. 1999. Domain-specific keyphrase extraction. *Proc. IJCAI*, Stockholm, Sweden, pp. 668–673.

Gabrilovich, E.; and Markovitch, S. 2006. Overcoming the brittleness bottleneck using Wikipedia. *Proc. National Conference on Artificial Intelligence*, Boston, MA.

Gay, C.W.; Kayaalp, M.; and Aronson, A. R. 2005. Semi-automatic indexing of full text biomedical articles. *Proc Fall Symp. of the American Medical Informatics Association*, Washington DC, USA, pp. 271–275.

Hripcsak, G.; and Rothschild, A.S. 2005. Agreement, the F-Measure, and reliability in information retrieval. *J. Am. Medical Information Association*, 12(3): 296–298.

Markó, K.; Hahn, U.; Schulz, S.; Daumke, P.; and Nohama, P. 2004. Interlingual indexing across different languages. *Proc. RIAO*, pp. 100–115.

Medelyan, O.; and Witten, I.H. 2008. Domain independent automatic keyphrase indexing with small training sets. To appear in *J. Am. Soc. Information Science and Technology*.

Mihalcea, R.; and Csomai, A. 2007. Wikify!: linking documents to encyclopedic knowledge. *Proc. CIKM*, pp. 233-242.

Milne, D.; Medelyan, O.; and Witten, I.H. 2006. Mining domain-specific thesauri from Wikipedia: A case study. *Proc. IEEE Int. Conf. on Web Intelligence*, Hong Kong.

Milne, D.; and Witten, I.H. 2008. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. Proc. *AAAI'08 Workshop on Wikipedia and Artificial Intelligence*, Chicago, IL.

Rolling, L. 1981. Indexing consistency, quality and efficiency. *Info. Processing and Management*, 17 (2): 69–76.

Turney, P. 2003. Coherent Keyphrase Extraction via Web Mining. Proc. *IJCAI-03*, Acapulco, Mexico, pp. 434-439.

Witten, I.H.; and Frank, E. 1999. *Data mining: Practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann, San Francisco, CA.

An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links

David Milne   **Ian H. Witten**

Department of Computer Science, University of Waikato
Private Bag 3105, Hamilton, New Zealand
{dnk2, ihw}@cs.waikato.ac.nz

Abstract

This paper describes a new technique for obtaining measures of semantic relatedness. Like other recent approaches, it uses Wikipedia to provide structured world knowledge about the terms of interest. Our approach is unique in that it does so using the hyperlink structure of Wikipedia rather than its category hierarchy or textual content. Evaluation with manually defined measures of semantic relatedness reveals this to be an effective compromise between the ease of computation of the former approach and the accuracy of the latter.

Introduction

How are *cars* related to *global warming*? What about *social networks* and *privacy*? Making judgments about the semantic relatedness of different terms is a routine yet deceptively complex task. To perform it, people draw on an immense amount of background knowledge about the concepts these terms represent. Any attempt to compute semantic relatedness automatically must also consult external sources of knowledge. Some techniques use statistical analysis of large corpora to provide this. Others use hand-crafted lexical structures such as taxonomies and thesauri. In either case it is the background knowledge that is the limiting factor; the former is unstructured and imprecise, and the latter is limited in scope and scalability.

These limitations are the motivation behind several new techniques which infer semantic relatedness from the structure and content of Wikipedia. With over two million articles and thousands of contributors, this massive online repository of knowledge is easily the largest and fastest growing encyclopedia in existence. With its extensive network of cross-references, portals and categories it also contains a wealth of explicitly defined semantics. This rare combination of scale and structure makes Wikipedia an attractive resource for this work (and for other NLP applications).

This paper describes a new technique—the Wikipedia Link-based Measure—which calculates semantic relatedness between terms using the links found within their corresponding Wikipedia articles. Unlike other techniques based on Wikipedia, WLM is able to provide accurate measures efficiently, using only the links between articles rather than their textual content. Before describing the details, we first outline the other systems to which it can be compared. This is followed by a description of the algorithm, and its evaluation against manually-defined ground truth. The paper concludes with a discussion of the strengths and weaknesses of the new approach.

Related Work

The purpose of semantic relatedness measures is to allow computers to reason about written text. They have many applications in natural language processing and artificial intelligence (Budanitsky, 1999), and have consequently received a lot of attention from the research community. Table 1 shows the performance of various semantic relatedness measures according to their correlation with a manually defined ground truth; namely Finkelstein *et al*'s (2002) WordSimilarity-353 collection.

The central point of difference between the various techniques is their source of background knowledge. For the first two entries in the table, this is obtained from manually created thesauri. WordNet and Roget have both been used for this purpose (McHale, 1998). Thesaurus-based techniques are limited in the vocabulary for which they can provide relatedness measures, since the structures they rely on must be built by hand.

Relatedness measure	Correlation with humans
Thesaurus based	
Wordnet	0.33-0.35
Roget	0.55
Corpus based	
Latent Semantic Analysis (LSA)	0.56
Wikipedia based	
WikiRelate	0.19-0.48
Explicit Semantic Analysis (ESA)	0.75

*Table 3***:** *Performance of existing semantic relatedness measures (from Gabrilovich and Markovitch, 2007)*

Corpus-based approaches obtain background knowledge by performing statistical analysis of large untagged document collections. The most successful and well known of these techniques is Latent Semantic Analysis (Landauer *et al.*, 1998), which relies on the tendency for related words to appear in similar contexts. LSA offers the same vocabulary as the corpus upon which it is built. Unfortunately it can only provide accurate judgments when the corpus is very large, and consequently the pre-processing effort required is significant.

Strube and Ponzetto (2006) were the first to compute measures of semantic relatedness using Wikipedia. Their approach—WikiRelate—took familiar techniques that had previously been applied to WordNet and modified them to suit Wikipedia. Their most accurate approach is based on Leacock & Chodorow's (1998) path-length measure, which takes into account the depth within WordNet at which the concepts are found. WikiRelate's implementation does much the same for Wikipedia's hierarchical category structure. While the results are similar in terms of accuracy to thesaurus based techniques, the collaborative nature of Wikipedia offers a much larger—and constantly evolving—vocabulary.

Gabrilovich and Markovitch (2007) achieve extremely accurate results with ESA, a technique that is somewhat reminiscent of the vector space model widely used in information retrieval. Instead of comparing vectors of term weights to evaluate the similarity between queries and documents, they compare weighted vectors of the Wikipedia articles related to each term. The name of the approach—Explicit Semantic Analysis—stems from the way these vectors are comprised of manually defined concepts, as opposed to the mathematically derived contexts used by Latent Semantic Analysis. The result is a measure which approaches the accuracy of humans. Additionally, it provides relatedness measures for any length of text: unlike WikiRelate, there is no restriction that the input be matched to article titles.

Obtaining Semantic Relatedness from Wikipedia Links

We have developed a new approach for extracting semantic relatedness measures from Wikipedia, which we call the Wikipedia Link-based Measure (WLM). The central difference between this and other Wikipedia based approaches is the use of Wikipedia's hyperlink structure to define relatedness. This theoretically offers a measure that is both cheaper and more accurate than ESA: cheaper, because Wikipedia's extensive textual content can largely be ignored, and more accurate, because it is more closely tied to the manually defined semantics of the resource.

Wikipedia's extensive network of cross-references, portals, categories and info-boxes provide a huge amount of explicitly defined semantics. Despite the name, Explicit Semantic Analysis takes advantage of only one property: the way in which Wikipedia's text is segmented into individual topics. It's central component—the weight between a term and an article—is automatically derived rather than explicitly specified. In contrast, the central component of our approach is the link: a manually-defined connection between two manually disambiguated concepts. Wikipedia provides millions of these connections, as

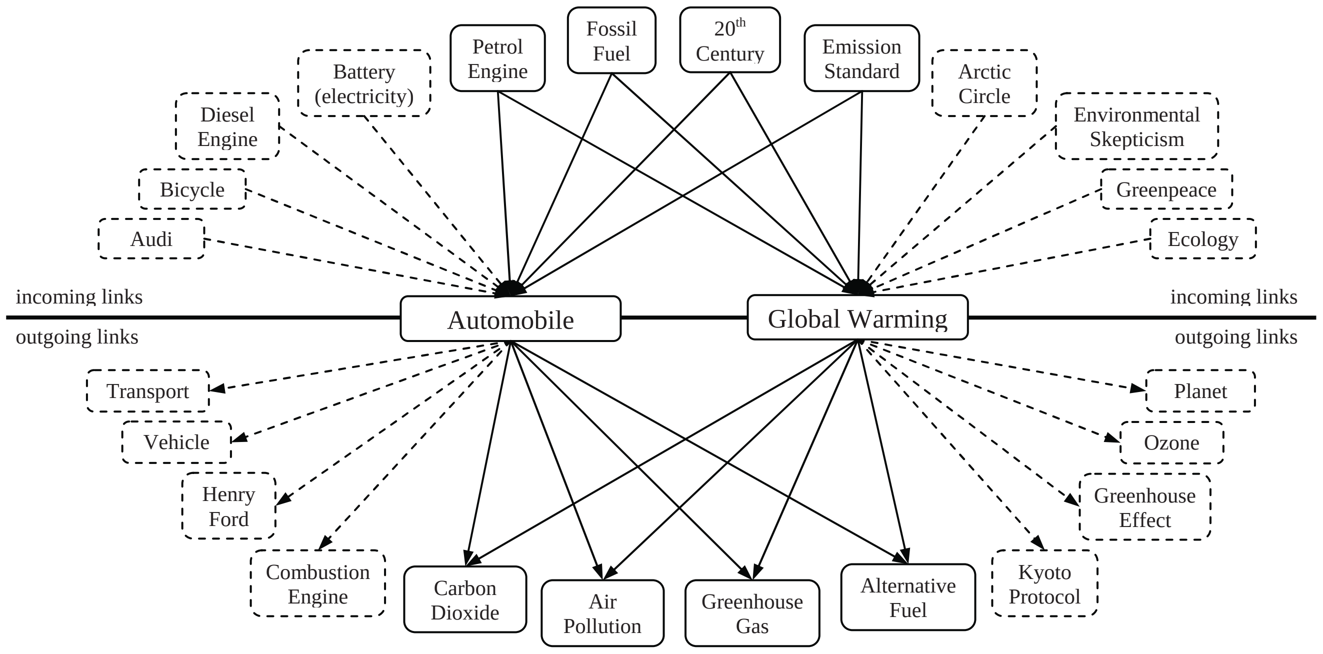


Figure 1: Obtaining a semantic relatedness measure between Automobile *and* Global Warming *from Wikipedia links.*

Figure 1 illustrates by attempting to answer the question posed at the start of the paper. It displays only a small sample—a mere 0.34%—of the links available for determining how *automobiles* are related to *global warming*. While the category links used by WikiRelate are also manually defined, they are far less numerous. On average, articles have 34 links out to other articles and receive another 34 links from them, but belong to only 3 categories.

The remainder of this section elaborates on our approach, and the various options we experimented with. It also assesses these individual components, in order to identify the best ones and define the final algorithm. Assessment of the algorithm as a whole—and comparison with related work—is left for the evaluation section. The testing reported in this section was done over a subset of 50 randomly selected term pairs from the WordSimilarity-353 collection, to avoid over-fitting the algorithm to the data.

Identifying candidate articles

The first step in measuring the relatedness between two terms is to identify the concepts they relate to: in Wikipedia's case, the articles which discuss them. This presents two problems: polysemy and synonymy.

Polysemy is the tendency for terms to relate to multiple concepts: for example *plane* might refer to a fixed-wing aircraft, a theoretical surface of infinite area and zero depth, or a tool for flattening wooden surfaces. The correct sense depends on the context of the term to which we are comparing it to; consider the relatedness of *plane* to *wing*, and *plane* to *surface*.

Synonymy is the tendency for concepts to be known by multiple names: a *plane* may also be referred to as *fixed wing aircraft*, *airplane* or *aeroplane*. It must be possible navigate to the appropriate article (and thus obtain the same relatedness measure) with any of these synonyms.

We use anchors—the terms or phrases in Wikipedia articles texts to which links are attached—to identify candidate articles for terms. Wikipedia's documentation dictates that any term or phrase that relates to a significant topic should be linked to the article that discusses it. Consequently it provides a vast number of anchor texts which capture both polysemy and synonymy: *plane* links to different articles depending on the context in which it is found, and *plane*, *airplane* and *aeroplane* are all used to link to the same article.

When testing this anchor-based approach on the WordSimilarity subset, we found that all of the 95 distinct terms in the collection were used as anchors in Wikipedia. In all cases the correct sense of the term was one of the anchor's destinations. All but one of the terms were ambiguous, with an average of 42 senses per term and a maximum of 371 senses (for *king*). This highlights one of the weaknesses of using anchors in this way: links often point to hyponyms of the anchor (in this case, specific kings) rather than senses.

Measuring relatedness between articles

Before the terms and candidate senses identified in the previous step can be disambiguated, we first judge the similarity between their representative articles. We have experimented with two measures. One is based on the links extending out of each article, the other on the links made to them. These correspond to the bottom and top halves of Figure 1.

The first measure is defined by the angle between the vectors of the links found within the two articles of interest. These are almost identical to the TF×IDF vectors used extensively within information retrieval. The only difference is that we use link counts weighted by the probability of each link occurring, instead of term counts weighted by the probability of the term occurring. This probability is defined by the total number of links to the target article over the total number of articles. Thus if *s* and *t* are the source and target articles, then the weight *w* of the link $s \to t$ is:

$$w(s \to t) = \log\left(\frac{|W|}{|T|}\right) \Bigg| \text{ if } s \in T \text{, 0 otherwise}$$

where *T* is the set of all articles that link to *t*, and *W* is the set of all articles in Wikipedia. In other words, the weight of a link is the inverse probability of any link being made to the target, or 0 if the link does not exist. Thus links are considered less significant for judging the similarity between articles if many other articles also link to the same target. The fact that two articles both link to *science* is much less significant than if they both link to a specific topic such as *atmospheric thermodynamics*.

These link weights are used to generate vectors to describe each of the two articles of interest. The set of links considered for the vectors is the union of all links made from either of the two source articles. The remainder of the approach is exactly the same as in the vector space model: the similarity of the articles is given by the angle (cosine similarity) between the vectors. This ranges from 0^o if the articles contain identical lists of links to 90^o if there is no overlap between them.

The second measure we use is modeled after the Normalized Google Distance (Cilibrasi and Vitanyi, 2007), which is based on term occurrences on web-pages. The name stems from the use of the Google search engine to obtain pages which mention the terms of interest. Pages that contain both terms indicate relatedness, while pages with only one of the terms suggest the opposite. Our measure is based on Wikipedia's links rather than Google's search results. Formally, the measure is:

$$sr(a,b) = \frac{\log(\max(|A|,|B|)) - \log(|A \cap B|)}{\log(|W|) - \log(\min(|A|,|B|))}$$

where *a* and *b* are the two articles of interest, *A* and *B* are the sets of all articles that link to *a* and *b* respectively, and—as before—*W* is the entire Wikipedia.

To evaluate these two measures independently from the disambiguation task, we manually identified the correct articles for each pair of terms in the WordSimilarity subset, and computed the correlation between manually defined measures and those provided by each approach. This provides a ground truth of *article* pairs—as opposed to *term* pairs—and manually defined measures of relatedness between them. Table 2 shows the results, and clearly identifies *Google Distance* as the more accurate measure. It also shows that a modest gain can be made by taking the average of the measures; this is the approach used in the remainder of the paper.

Measuring relatedness between terms

Once the candidate articles have been identified and the relatedness between them calculated, we resolve ambiguity by selecting one candidate article to represent each term. As with the previous step, there are several options, which we again assess (in Table 3) across the WordSimilarity subset.

First, one can make a snap decision by using the most common sense of each term. For example, when making a judgment between *Israel* and *Jerusalem*, one would consider only the nation and its capital city. The obscure but strong connection between two townships in Ohio with the same names would be completely ignored. The commonness of a sense is defined by the number of times the term is used to link to it: e.g. 95% of *Israel* anchors link to the nation, 2% to the football team, 1% to the ancient kingdom, and a mere 0.1% to the Ohio township. As shown in Table 3, merely selecting the *most common pair* of terms performs fairly well.

Another approach is to use the two terms involved to disambiguate each other. For example, when identifying the relatedness of *jaguar* and *car* it makes sense to use *car* to determine that we are talking about the automobile manufacturer *Jaguar Cars Ltd*, rather than the species of cat. This amounts to selecting the two candidate senses which most closely relate to each other. As shown in Table 3, selecting the *most closely related pair* of senses performs better than the most common sense heuristic, but is marred by the number of obscure senses available: there may be hundreds for each term. Consequently, for efficiency and accuracy's sake we only consider articles which receive at least 1% of the anchor's links. This theoretically leaves up to 100 candidates to be examined, but in practice the distribution of links for each anchor follows the power law, meaning that the vast majority are made to a handful of candidates. In the sample, the largest number of candidates examined for a term was 26.

The best results are obtained when both commonness and relatedness are considered. Evenly weighting the candidate senses by these variables and choosing the pair with the highest weight—*highest (commonness + relatedness)*—gives exactly the same results as with manual disambiguation (Table 2, row 3). Interestingly we can improve upon this by making a simple *sequential decision*, which first groups the most related pairs of candidates (within 40% of the most related pair) and then chooses the most common pair. This makes subtly different choices from those made manually. Given the term *environment* and the context *ecology*, for example, the system selects *ecosystem* as the representative article rather than *natural environment*, and consequently obtains a more accurate relatedness score.

Finally, we can also consider the case where two words are closely related because they belong together in the same phrase: e.g. *family planning* is a well-known phrase, and consequently *family* and *planning* are given a high semantic relatedness by humans even though their respective concepts are relatively disjoint. To identify these cases we simply concatenate the terms and see whether this is used as an anchor. If so, the frequency with which the anchor is used is normalized and added to the relatedness score of the original terms. This gives our *final relatedness measure*, which has a correlation of 0.78 with manual judgments over the WordSimilarity sample.

Evaluation

We evaluated the Wikipedia Link-based Measure on three datasets of term pairs and manually defined relatedness measures: Miller and Charles' (1991) list of 30 term pairs, Rubenstein and Goodenough's (1965) 65 pairs, and the WordSimilarity-353 dataset described under Related Work. The version of Wikipedia used to obtain our measures was released on November 20, 2007. At this point it contained approximately 13GB of uncompressed XML markup. This relates to just under two million articles, which constitute the various concepts for which semantic relatedness judgments were available. We also mined over five million distinct anchors, which defines the vocabulary of terms by which these concepts can be accessed.

Table 4 compares WLM with its two main competitors—WikiRelate and ESA—by their correlation with manually defined judgments. Only the best measures obtained by the different approaches are shown. It should be noted that the results were obtained with different

Relatedness measure	Correlation with humans
TF×IDF inspired	0.66
Google Distance inspired	0.72
combined (average)	0.74

Table 2: Performance of semantic relatedness measures between manually disambiguated articles

Relatedness measure	Correlation with humans
most common pair	0.68
most closely related pair	0.69
highest (commonness + relatedness)	0.74
sequential decision	0.75
final relatedness measure	0.78

*Table 3**:** Performance of relatedness measures (and disambiguation strategies) between original terms*

snapshots of Wikipedia, which may effect performance.

Across all three datasets, we see a consistent trend: WLM is better than WikiRelate but worse than ESA. The final row in the table combines the results across the three datasets, with correlations weighted by the size of each dataset. This shows WLM outperforming WikiRelate by 0.19, and in turn being outperformed by ESA by 0.08. The third row in Table 4 shows the performance of the algorithms over the WordSimilarity-353 collection. The accuracy of 0.69 for our system can be directly compared to the results in Table 1, which were obtained from the same dataset. Our algorithm outperforms all others except ESA by at least 0.13.

It is interesting to note the drop in WLM's performance between the WordSimilarity sample used in the previous section (0.78) and the full dataset used here (0.69). Much of the drop may be due to over-fitting the algorithm to the sample. Analysis of the results, however, reveals another reason: WLM differs most from the ground truth when the terms being compared cannot be resolved to suitable Wikipedia articles. For example, there is no article for the concept *defeat*; the anchor points only to specific military encounters. These cases are common in the full dataset but were, by chance, excluded from the sample.

Figure 2 plots the performance of WLM as successively more pairs are discarded from the WordSimilarity-353 collection so that only the most well defined terms are considered. We use anchor frequency as a simple indicator of how well a term is defined; if a term is not used to make a sufficient number of links, it is considered problematic. It is likely that ESA's performance would remain constant here, since it does not distinguish between terms used as anchors and those that appear in plain text. Thus Figure 2 shows how WLM's performance approaches the benchmark of 0.75 set by ESA when the terms involved are well defined as individual articles in Wikipedia.

Discussion

The previous section clearly identifies ESA as the best measure. It is less brittle, because it only requires that articles mention the terms involved. WLM, however, achieves competitive levels of accuracy when the terms involved correspond to distinct Wikipedia articles. Given Wikipedia's sheer size and rate of growth, one can expect this to hold true whenever the terms represent topics which one could reasonably write an article about. This is the case for most applications in the literature, which deal primarily with topics: named entities (Bunescu and Pasca, 2006; Cucerzan, 2007); key phrases (Mihalcea and Csomai, 2007); categories (Gabrilovich and Markovitch, 2006); or entries in existing ontologies (Medelyan and Legg, 2008) and thesauri (Ruiz-Casado *et al.*, 2005). In these applications, we can expect WLM to be competitive with the state of the art.

The advantage of our approach is that it requires far less data and resources. To obtain measures from ESA, one must preprocess a vast amount of textual data; 13Gb as of November 2007. Each term must be matched to the articles in which it is found, and each of the resulting lists of articles must be weighted, sorted, and pruned. One assumes (given the sorting requirement) that this is a log-linear at best. By comparison, WLM requires only the link structure of Wikipedia (450 Mb) and the statistics of how often anchors are used to link to different concepts (140Mb). No preprocessing is required other than to extract this information from Wikipedia's XML dumps. This is a straight-forward task that can be achieved in linear time (assuming constant hash-table operations).

Another advantage is the accessibility of our approach. At the time of writing, ESA is not publicly available. The only known re-implementation is based on the smaller German version of Wikipedia and a very restricted vocabulary of 10,000 terms (Jacobs, 2007). WLM in contrast is readily available as part of the open source WikipediaMiner toolkit.[1] This implementation provides measures for the full anchor vocabulary of whatever version of Wikipedia it is applied to: currently more than 5 million distinct phrases for the English language version. Other language versions have not yet been tested, but in principle the approach is language independent.

ESA is able to determine the relatedness of texts of any length as easily as individual terms, by gathering and merging the lists of articles related to each word. WLM and WikiRelate are not so easily extended: they require an additional step—such as Mihalcea and Csomai's (2007) wikification technique—to discover the topics mentioned in the texts. This requirement may well turn out to be an advantage, however, because both techniques would then be comparing collections of concepts and topics rather than

Dataset	WikiRelate	ESA	WLM
Miller and Charles	0.45	0.73	0.70
Rubenstein and Goodenough	0.52	0.82	0.64
WordSimilarity-353	0.49	0.75	0.69
Weighted average	*0.49*	*0.76*	*0.68*

Table 4: Performance of semantic relatedness measures for three standard datasets.

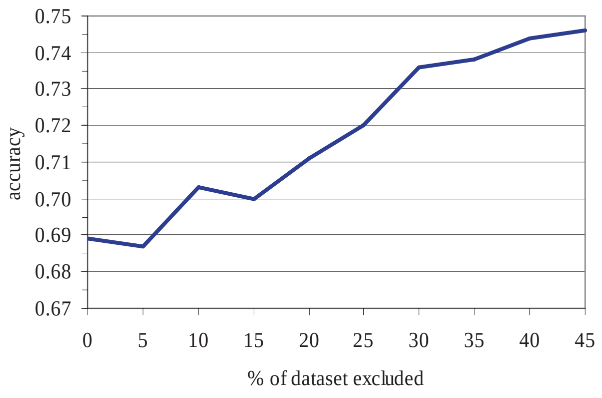


Figure 2: Accuracy of WLM with weakly defined terms excluded.

words. This highlights a fundamental difference between ESA and the other two approaches: that it disregards stop-words and word order. It considers, for example, *wind break* and *break wind* to be the same thing. It is unclear how much this affects the overall accuracy of the three techniques, as the datasets upon which they have so far been compared are restricted to individual words.

Conclusion

In this paper we propose and evaluate WLM, a novel approach to computing semantic relatedness with the aid of Wikipedia. Our approach is most similar to WikiRelate and ESA, which also exploit the online encyclopedia for this purpose. WLM forms a compromise between these very different methods by utilizing Wikipedia's network of inter-article links, rather than the comparatively small category hierarchy used by the former system, or the full textual content used by the latter.

Our measure consistently outperforms WikiRelate across all datasets. ESA remains the best measure in terms of robustness; however, we are able to match its accuracy when the terms involved are defined as individual articles in Wikipedia. Given the number of potential applications for which this requirement holds, we consider WLM to be a valuable contribution. For many tasks we expect it to be competitive with ESA, while using far less data and resources.

Future work will involve applying WLM to various tasks in order to investigate its utility more fully. Strube and Ponzetto (2006) have rightly pointed out the danger in using a few subjective and relatively small datasets for evaluation. Like them, we hope to apply our work to a host of NLP tasks that will require hundreds of thousands of relatedness judgments to be made, and thus provide a more reliable evaluation. Please, download the code,[1] apply it to your own NLP problems, and help us in this endeavor!

[1] An open-source implementation of WLM is available at www.sourceforge.net/WikipediaMiner

References

Budanitsky, A. (1999) *Lexical Semantic Relatedness and its Application in Natural Language Processing*. Ph.D. thesis, University of Toronto, Ontario.

Bunescu, R. and Pasca, M. (2006) Using Encyclopedic Knowledge for Named Entity Disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06)*, Trento, Italy.

Cilibrasi, R.L. and Vitanyi, P.M.B. (2007) The Google Similarity Distance. *IEEE Transactions on Knowledge and Data Engineering 19*(3), 370-383.

Cucerzan, S. (2007) Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP 2007)*, Prague, Czech Republic.

Finkelstein, L., Gabrilovich, Y.M., Rivlin, E., Solan, Z., Wolfman, G. and Ruppin, E. (2002) Placing search in context: The concept revisited. *ACM Transactions on Information Systems 20*(1).

Gabrilovich, E. and Markovitch, S. (2006) Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, Boston, MA.

Gabrilovich, E. and Markovich, S. (2007) Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, Hyderabad, India.

Jacobs, H. (2007) *Explicit Semantic Analysis (ESA) using Wikipedia*. Retrieved March 14, 2008 from http://www.srcco.de/v/wikipedia-esa

Landauer, T.K. and Foltz, P.W. and Laham, D. (1998) An introduction to latent semantic analysis. *Discourse Processes 25*(2-3), pp 259-284.

Leacock, C. & M. Chodorow (1998). Combining local context and WordNet similarity for word sense identification. In C. Fellbaum (Ed.), *WordNet. An Electronic Lexical Database*, Chp. 11, pp. 265.283. Cambridge, Mass.: MIT Press.

McHale, M. (1998) A Comparison of WordNet and Roget's Taxonomy for Measuring Semantic Similarity, In *Proceedings of COLING/ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, Montréal, Canada. pp. 115-120.

Medelyan, O. and Legg, C. (2008) Integrating CYC and Wikipedia: Folksonomy meets rigorously defined common-sense. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence (WIKIAI 2008)*.

Mihalcea, R. and Csomai, A. (2007) Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on Information and Knowledge Management (CIKM)*, Lisbon, Portugal. pp 233-242.

Miller, G. A. & W. G. Charles (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes, 6*(1):1-28.

Rubenstein, H. & J. Goodenough (1965). Contextual correlates of synonymy. *Communications of the ACM, 8*(10):627.633.

Ruiz-Casado, M., Alfonseca, E., Castells, P. (2005) Automatic assignment of Wikipedia Encyclopedic Entries to WordNet synsets. In *Proceedings of Advances in Web Intelligence*, Lodz, Poland.

Strube, M. and Ponzetto, S. P. (2006). WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, pp.1419-1424.

Mining Wikipedia's Article Revision History for Training Computational Linguistics Algorithms

Rani Nelken and **Elif Yamangil**
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138, USA
{nelken,elif}@eecs.harvard.edu

Abstract

We present a novel paradigm for obtaining large amounts of training data for computational linguistics tasks by mining Wikipedia's article revision history. By comparing adjacent versions of the same article, we extract voluminous training data for tasks for which data is usually scarce or costly to obtain. We illustrate this paradigm by applying it to three separate text processing tasks at various levels of linguistic granularity. We first apply this approach to the collection of textual errors and their correction, focusing on the specific type of lexical errors known as "eggcorns". Second, moving up to the sentential level, we show how to mine Wikipedia revisions for training sentence compression algorithms. By dramatically increasing the size of the available training data, we are able to create more discerning lexicalized models, providing improved compression results. Finally, moving up to the document level, we present some preliminary ideas on how to use the Wikipedia data to bootstrap text summarization systems. We propose to use a sentence's persistence throughout a document's evolution as an indicator of its fitness as part of an extractive summary.

Introduction

Much recent progress in natural language processing stems from successfully leveraging large-scale document corpora as a source of training data. Text documents are almost invariably found in fixed final form, a form which hides an often rich history of the documents' evolution from inception as a first draft to final published form. If we could somehow gain access to this information for a large document corpus, we could learn invaluable information from it.

Fortunately, Wikipedia provides just such a resource. Through Wikipedia's collaborative editing process, articles are iteratively amended and refined by multiple Web users. Wikipedia offers periodic snapshots of all of this historical data for its more than 7 million articles, thus providing a virtual paper trail of this collaborative editing process. It would not be an exaggeration to state that in the entire history of writing there has never been such a comprehensive resource containing the full history of so many documents.

We present a new paradigm for leveraging this data for training language processing algorithms. By comparing different versions of the same document, and repeating the process over a large collection of documents, we propose to collect users' editorial choices. We illustrate this process at different levels of linguistic granularity ranging from the level of the single word, through the sentence, to the document, using the data for three different tasks.

First, consider the problem of automated text correction. Text is often fraught with errors in spelling, style, and grammar, requiring subsequent correction and editing. Collecting common errors and their corrections is of obvious practical interest for text-correction algorithms as well as theoretical linguistic studies. Although modern word processors provide support for automated and semi-automated text correction, including context-sensitive spelling correction and grammar-checking, even the most sophisticated tools still fall short of catching all errors.[1] Given a supervised collection of typical errors, text correction algorithms can be accurately trained to identify and correct errors. For instance, Carlson, Rosen, and Roth (2001) presented a Winnow-based algorithm achieving accuracy levels at the 99% range for 265 lexical confusion sets, but how can we effectively collect such confusion sets automatically?

We propose that such errors and their corrections can be automatically harvested from Wikipedia article revisions. Since Wikipedia articles are viewed by many pairs of eyes, errors inadvertently added by one user are likely to be identified and corrected by subsequent users. By comparing temporally adjacent versions of the same article, we can metaphorically peer over users' shoulders just as they are making corrections. As a proof of principle, we illustrate this approach here on a very specific form of lexical error, known as "Eggcorns", a lexical error that retains both phonetic similarity and some level of semantic coherence. We show how such errors, which are of great interest to linguists—in addition to their practical interest for context-sensitive spelling correction—can be mined automatically from Wikipedia's revision history.

As a second application, we move up from the lexical level to the sentence level, presenting a new algorithm for sentence compression, the problem of shortening sentences

[1]See `http://faculty.washington.edu/sandeep/check/` for a critique of a popular commercial text editor's correction capabilities.

by dropping words in a way that preserves the most pertinent information and does not harm grammaticality. Such shortening is useful for instance for subtitle or caption generation or as part of a larger machine translation or summarization system. This problem has received much attention in the literature, but has suffered from a severe dearth of training data. We show how we can obtain training data up to 3 orders of magnitude larger than ever used for this task. Using only a fraction of this data, we train a novel statistical noisy channel model for sentence compression, showing improved compression rates and grammaticality with only a slight decrease in the importance of the retained information.

Finally, moving to the document level, we describe some early experiments with a new approach to taking advantage of Wikipedia revision data for training text summarization systems. Inspired by biological evolution, we hypothesize that the temporal persistence of a sentence throughout the revision history is a good indicator of its importance. We present some preliminary experiments lending credence to the idea that this approach can be used to help train text summarization systems using a corpus of unprecedented size.

Throughout the presentation of these three different tasks, we maintain the same theme of using Wikipedia's revision history to provide significant amounts of training data, which was previously either extremely sparse or very costly to acquire manually.

All our experiments were performed on the July 2006 full history version of the English Wikipedia, consisting at the time of 1.4 million articles. To efficiently process a dataset of such size (hundreds of GBs), we split it into multiple smaller chunks, and distribute all the processing among multiple processors. Note that at present Wikipedia includes almost 2.3 million English articles, significantly increasing the available data.

Harvesting Eggcorns

The term "eggcorn" was coined by Geoff Pullum on the popular "Language Log" Blog[2] for a particular type of error in English language usage. This error occurs when an expression or part of it is substituted by a homophone such that the resulting expression still makes sense semantically, even while deviating from common usage. The word "eggcorn" is itself an eggcorn for "acorn". The idea is that the error is not merely phonetic, but there are also semantic reasons behind the confusion, however misguided it is. For instance, an acorn and an egg share the same shape, and a grain of corn and an acorn are both seeds. Such usage errors are a source of interest (as well as undeniable amusement) for linguists, who may use them to learn interesting variations in common usage and their underlying semantics.

We denote eggcorns here as ordered pairs of the form ⟨*incorrect, correct⟩, using the linguists' star notation for the incorrect form. As a canonical running example we use the pair ⟨***full**proof, **fool**proof⟩. We are interested in automatically identifying occurrences of such eggcorns from Wikipedia revisions. A useful reference database of eggcorns which we use is the Eggcorn Database,[3] which documents 596 eggcorns. For simplicity, we focus here only on eggcorns consisting of a pair of single words (thus ignoring eggcorns of the form ⟨*a posable, opposable⟩ (as in "*a posable thumb"). We also exclude any eggcorns where one of the words is a stop-word, e.g., ⟨*and, ad⟩ (as in "*and hoc"), leading to a total of 348 reference eggcorns.[4]

[2]http://itre.cis.upenn.edu/~myl/languagelog/

Eggcorns are defined by a combination of semantic and phonetic similarity. The notion of semantic similarity underlying eggcorns is extremely nebulous, and non-uniform. For instance, we can detect and retrospectively rationalize some notions of semantic similarity in eggcorn pairs such as ⟨*crutch, crux⟩ and ⟨*isle, aisle⟩, or indeed in ⟨*eggcorn, acorn⟩, but it is much more difficult to formulate a general criterion that captures such similarity, much less automate it. For instance, it is clear that this type of similarity goes well beyond standard thesaural relations, and is unlikely to be easily amenable to co-occurrence based measures.

Phonetic similarity, on the other hand, is much more straightforward. We experimented with two measures:

- The classic Soundex algorithm, developed by Odell and Russell and patented as early as 1918 (Hall and Dowling 1980), which is based on mapping a word's consonants to one of 6 classes. Each word is then mapped to a concise 4-character code which consists of the first letter of the word and the first three distinct class indices of its letters. Two words are considered similar if they are mapped to the same code.
- Editex (Zobel and Dart 1996), which uses a similar though distinct method of classifying the letters into classes. Given two words, it then applies an edit-distance computation on the full encoding. Two words are considered similar if the edit-distance is below some threshold.

We tested these algorithms by applying them to the reference eggcorns, with the results in Table 1. Best results for Editex were achieved with an edit-distance threshold of 0.5. We therefore also used it for finding new eggcorns.

Table 1: Phonetic similarity of reference eggcorns

Algorithm	Precision	Recall
Soundex	0.95	0.64
Editex (threshold = 0.5)	0.90	0.96

Finding eggcorns

We are interested in all corrections of the form $\langle *w_1, w_2\rangle$, where the words are phonetically similar, are not morphologically related, and are not synonyms.

It is thus insufficient to search just for occurrences of w_1 or of w_2; we need to find genuine instances where w_1 is changed to w_2. We are interested in finding instances of

[3]http://eggcorns.lascribe.net/

[4]In addition to the officially sanctioned eggcorns, the user forums on the Eggcorn Database site include hundreds of additional user-contributed eggcorn candidates. For simplicity, we did not look for these.

the reference eggcorns as well as new, previously unreported eggcorns. To limit the search space, we first indexed all the articles that contain an occurrence of w_2, where w_2 is the correct member of one of the reference eggcorns. Over these articles we then searched for all cases where some word, w_1, is modified to a phonetically similar word, w_2. This approach ensures that we will traverse all the articles containing a potential occurrence of a reference eggcorn, and has the added potential of finding additional eggcorns.

We split each article into its set of revisions, $(r_1, r_2, \ldots, r_N)$ in chronological order. We find all pairs of adjacent revisions (r_n, r_{n+1}) where r_{n+1} included w_2, but r_n did not. Where did w_2 come from? Either it was typed correctly by the editor of r_{n+1}, or it already appeared in a possible incorrect form, w_1, in r_n, and was modified to w_2 in r_{n+1}, which is precisely the case we are interested in.

We seek the word w_1 that was w_2's predecessor in r_n by performing a double edit-distance operation. Using lines as a simple proxy for sentences, we split r_n and r_{n+1} into lines, and run an edit-distance operation, treating each line as an atomic token. We then examine where the line containing w_2 in r_{n+1} came from. If this line is a replacement of another line in r_n, then we run a second edit-distance on these two lines, this time at the resolution of words, to find where w_2 came from. If it is a replacement of a word, w_1, we check whether w_1 is phonetically similar to w_2. We further use WordNet (Fellbaum 1998) to filter out any pairs that share a morphological stem, or are potential synonyms.[5] This WordNet filter eliminates two reference eggcorns, ⟨*font, fount⟩, and ⟨*flare, flair⟩. In both cases, the words are synonyms under a dispreferred disambiguation. We output the resulting pairs $\langle *w_1, w_2 \rangle$ as eggcorn candidates.

We successfully found 31% of the reference eggcorns, such as ⟨*dandruff, dander⟩, ⟨*curtsey, courtesy⟩, and ⟨*isle, aisle⟩. Of course, there is no guarantee that the remaining eggcorns have ever occurred in Wikipedia. In addition, the procedure was able to identify many new and interesting eggcorns not previously listed in the Eggcorn Database. We list some examples in Table 2. Not surprisingly, less common words are more prone to error.

Table 2: Sampling of new eggcorns identified from Wikipedia

⟨*funder, founder⟩	⟨*rectify, ratify⟩
⟨*heaven, haven⟩	⟨*absorb, adsorb⟩
⟨*acerbate, exacerbate⟩	⟨*arrogate, abrogate⟩
⟨*birth, berth⟩	⟨*citing, sighting⟩
⟨*siege, seize⟩	⟨*ripe, rife⟩
⟨*rigid, rugged⟩	⟨*reverse, revert⟩
⟨*assume, resume⟩	⟨*restrain, refrain⟩

This procedure is geared towards high recall, and thus unsurprisingly has poor precision. False positives include typos ("bight" instead of "blight") pairs of words that are not in fact phonetically confusable (e.g., "ability" and "agility"), or not semantically related (e.g., "bacon" and "beacon"), or profanity. Future work is needed to filter these out effectively.

These results provide an encouraging proof of principle for mining such corrections automatically from Wikipedia revisions. We are currently working on a more comprehensive context-sensitive text-correction system trained on more general Wikipedia revision corrections.

Sentence compression

Moving from the lexical to the sentential level, we next explore sentence compression. We summarize the main contribution here, with fuller details presented in (Yamangil and Nelken 2008).

With the increasing success of machine translation in recent years, several researchers have suggested transferring similar methods for monolingual text rewriting tasks. In particular, Knight and Marcu (2000) (KM) applied a noisy channel model to the task of *sentence compression* — dropping words from an individual sentence while retaining its important information, and without sacrificing its grammaticality.

A well-recognized problem of sentence compression is data sparseness. While bilingual parallel corpora are relatively easy to obtain, collections of sentence compressions are quite rare. Indeed, most work on sentence compression has used the Ziff-Davis corpus (Knight and Marcu 2000), which consists of a mere 1067 sentence pairs. While data sparseness is a common problem of many computational linguistics tasks, the dearth of sentence compression data is a well recognized problem (Turner and Charniak 2005).

In recent years, there has been a growing interest in the related and more general tasks of sentence paraphrasing (Dolan, Quirk, and Brockett 2004), and textual entailment (Dagan, Glickman, and Magnini 2005), together with concentrated efforts to create common datasets for these problems. Theoretically, we could use this data for sentence compression as well, by filtering just those sentence pairs that are true compressions, as opposed to more general paraphrases. Unfortunately, however, these datasets are still only on the order of hundreds or thousands of sentences, only a fraction of which would be directly useful for compression, and hence they do not solve the sparseness problem.

Mining Wikipedia revisions from 250,000 articles, we have extracted over 380,000 full/compressed sentence pairs, 2 orders of magnitude more than the number of pairs in the Ziff-Davis corpus. Since Wikipedia currently has almost 2.3 million English articles and is constantly expanding, we can expect an increase of another order of magnitude. We thus can afford to be extremely selective of the sentence pairs we use. We make the simplifying assumption that *all* such sentence pairs also retain the core meaning, and are therefore valid training data for our purposes. This assumption is of course patently naïve, as there are many cases in which such revisions reverse sentence meaning, add or drop essential information, are part of a flame war, etc. Classifying these edits is an interesting task which we relegate to future work.

As with the eggcorns, we first extract all revisions, for each article. Here splitting into lines is no longer sufficient,

[5] We thank one of the reviewers for suggesting the WordNet filter.

so we split each revision into its sentences using a rule-based sentence splitter. [6]

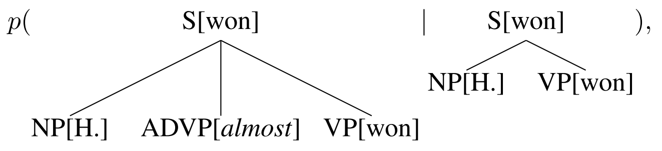
For each article, we run an edit-distance comparison between each temporally adjacent pair of revisions, treating each sentence as an atomic token. We look for all replacement operations of one sentence by another, and check whether one sentence is a compression of the other. We filter out ungrammatical sentences during preprocessing in which we run Collins' parser (1997), filtering out any sentences that score below a certain threshold.

We wish to apply KM's generative noisy channel model for the problem. Under this model, sentences start their life in short form, s, are ranked by a source language model, $p(s)$, and then probabilistically expanded to form the long sentence, $p(l|s)$. During decoding, given a long sentence, we seek the most likely short sentence that could have generated it. Using Bayes' rule, this is equivalent to seeking the short sentence s that maximizes $p(s) \cdot p(l|s)$.

This huge increase in training data enables not only a quantitive improvement in existing models, but a qualitative improvement as well. Whereas KM's original model was purely grammatical, we take advantage of the huge increase in data to lexicalize the model. Thus, sentence compression decisions can be made not only on the basis of grammatical information, but also taking into account the values of the lexical items. To illustrate why this is useful, consider the following sentence pair:

1. Hillary *barely* won the primaries.
2. Hillary *almost* won the primaries.

The validity of dropping the adverbial here clearly depends on the lexical value of the adverb. It is much more acceptable to drop the adverb in Sentence 1, since dropping it in Sentence 2 reverses the meaning. We learn probabilities of the form:



where we percolate the lexical head for each phrase up the tree. Our model makes compression decisions based on lexical dependencies between the compressed and retained parts of the parse tree.

We use Witten-Bell discounting (1991) to gradually back off from fully lexicalized probabilities to the purely grammatical probabilities, in cases where there is not enough lexical information. In addition to the lexicalized channel model, we also use a lexicalized probabilistic syntax-based source model, which we train from the parser's output on the short sentences of each sentence pair. Finally, decoding is done using the statistical sentence generator of Langkilde (2000), from which we extract the best scoring compression.

[6]We used a sentence splitter by Paul Clough, from `http://ir.shef.ac.uk/cloughie/software.html`

Evaluation

We evaluated our system using the same method as KM, using the same 32 sentences taken from the Ziff-Davis corpus.[7] We solicited judgments of *importance* (the value of the retained information), and *grammaticality* for our compression, the KM results, and human compressions from 8 native English speaking judges on a scale of 1 (worst) to 5 (best). Mean and standard deviation are shown in Table 3. Our method achieves an increase in compression rate as well as grammaticality over KM's method, with a slight decrease in importance.

Text summarization

As a final illustration of this new paradigm, we now move on to the document level, and in particular focus on text summarization, a task which has received much attention in the literature. The dominant approach to summarization is sentence extraction, in which a summary is created by extracting sentences from the full document, optimizing sentence salience and coherence, while avoiding redundancy.

We suggest that information in the revision history may be useful for summarizing Wikipedia articles. There are two motivations for this approach. First, there is independent interest in summarizing Wikipedia articles, due to their ubiquitous use. Although many Wikipedia articles include short abstracts, these abstracts have a fixed short length, and it would be useful to create summaries of flexible length.

More interestingly, if features extracted from the revision history are indeed significant indicators of importance, we could use these features to bootstrap further summarization research as follows. First, we would train a supervised or semi-supervised Wikipedia summarization system based in large part on the revision history. Second, we would apply the trained summarizer to Wikipedia articles, obtaining a huge collection of pairs of articles and their automatically generated summaries. Finally, we would use these pairs as training data for further summarization systems, which would no longer require the revision history.

What can we learn from the revision history about the importance of sentences for summaries? Following a biological intuition, we hypothesize that a sentence's persistence in an article is a good measure of its "fitness", i.e., the importance of it being included in the article. For all sentences in the most recent version of the article, we wish to track their lifespan from the moment they first appear in the document, through whatever changes they undergo, until the final version. Our hypothesis is that sentences that have persisted longer are more likely to be important than sentences that are only more recent additions, and are thus better candidates for extractive summaries.

Since sentences evolve, we have to be careful in tracking sentences. Viégas, Wattenberg, and Dave (2004) presented striking visualizations of sentence-level history of Wikipedia articles, using a notion of sentence identity up to the character level. This notion is too strict for our purposes (and arguable also for visualization). We therefore define a new notion of weak sentence identity up to an edit-distance

[7]We thank KM for sharing their dataset with us.

	KM	Our model	Humans
Compression	72.91%	67.38%	53.33%
Grammaticality	4.02±1.03	4.31±0.78	4.78±0.17
Importance	3.86±1.09	3.65±1.07	3.90±0.58

Table 3: Compression evaluation results

threshold, computed pairwise between the revisions. [8] For each sentence in the final version, we define its persistence at revision r_n to be the percentage of revisions in which it has survived.

$$\text{persistence}_n(s) = \frac{\#\text{revisions until } r_n \text{ including} s}{n}$$

We define the (final) persistence of a sentence as its persistence in the final revision. We discount large-scale spam revisions such as wholesale deletions. We also maintain the identity of sentences that are deleted and disappear from up to 50 revisions only to reappear in a subsequent revision.

To illustrate the effect of sentence identity on persistence, the average and standard deviation for persistence using both types of identity for a particular Wikipedia article, "World of Warcraft" (WoW), are as follows: 0.470 ± 0.365 (using weak identity), 0.162 ± 0.210 (using strict identity). As expected, the average using weak identity is higher, indicating that we can track sentences along a larger number of revisions, discounting small changes. Moreover, the standard deviation is higher, indicating more diversity in persistence, and thus a stronger differentiation between sentences. The Pearson correlation coefficient between the two persistence score vectors is 0.418, indicating a large qualitative difference.

To gauge the efficacy of this approach, we scored the persistence of sentences of two Wikipedia articles, "WoW" and "William Shakespeare" (WS) and plotted them shown in the heat-maps of Figures 1 and 2. In these figures, the (n, k) pixel corresponds to persistence$_n(s_k)$, where s_k is the k'th sentence of the final version. The color indicates the level of persistence, on a scale from blue (0) to red (1). Each sentence contributes a horizontal stripe of growing persistence, with highest persistence sentences tending towards the red end of the spectrum. Discontinuities indicate that the sentence was dropped from a small number of revisions only to return later on.

Examining the preserved sentences, we see that they often correspond to the first sentences in each section. We have marked these correspondences on the Y-axis of the figures. Since the lead sentences of a section are often the most important, this graph lends credence to our hypothesis that sentence persistence correlates with importance. Of course, much more extensive evaluation is required to validate this hypothesis. In addition, we see that structural Wikipedia markup is often maintained as well. This is illustrated in Figure 2 towards the bottom of the document. This document, WS, includes a structured section containing a collection of links, and these, not surprisingly well-preserved.

Obviously, our preliminary observation of the correlation between persistence and importance in no way constitutes a full summarization system. A true summarization system would not use this feature in isolation, but as part of a larger collection of features. Indeed, we plan to examine additional features of the revision history, such as user confidence, the number and extent of edits that a sentence underwent, number of revisions two sentences were adjacent, etc. The hope is to use such automatically extracted features to replace or augment human annotation in a method such as (Kupiec, Pederson, and Chen 1995). If successful, this approach can be applied to millions of Wikipedia articles, which could then be used as training data for further summarization algorithms.

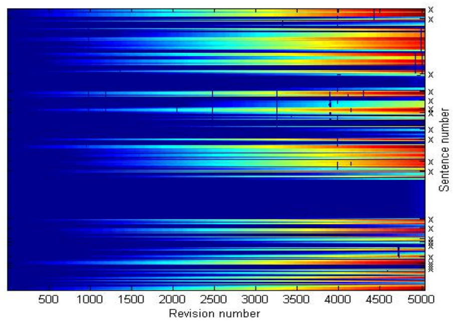


Figure 1: Persistence heat map for WoW

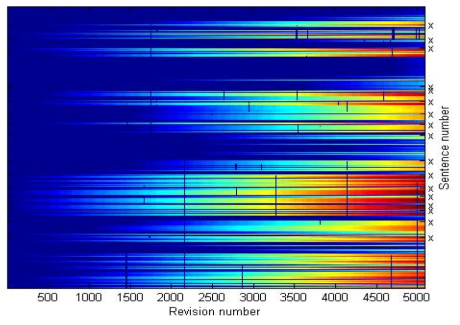


Figure 2: Persistence heat map for WS

[8]We chose the edit-distance threshold manually. In future work, we will train this threshold using cross-validation on a small hand-labeled training set. We thank one of the reviewers for this suggestion.

Conclusion

We have presented a new paradigm for obtaining large scale training data for training natural language processing algorithms from Wikipedia's article revision history, illustrating the approach on three specific tasks. Additional applications of this method abound at all levels of linguistic granularity. For instance, at the lexical level, our focus on eggcorns was more illustrative than exhaustive. For a full text correction application, one could mine the revisions for other types of lexical or phrasal replacements. One of the challenges of such an endeavor is filtering out the content-based replacements that would be meaningful only in the context of a particular article from the more general corrections that can be applied across the board. Hopefully, since content-related corrections would be specific to a particular document or domain context, the more general corrections would recur much more frequently in the corpus.

In addition to correction, one particularly interesting form of lexical substitution is the use of anaphoric pronouns. The revision history contains many instances of a noun phrase being replaced by an anaphoric pronoun or vice versa. If we make the plausible assumption that such replacements retain sentence meaning, we can use this data to train anaphora resolution systems using millions of articles.

Moving up to the sentence level, much work remains on extending the sentence compression method we have described. In particular, we are interested in learning better predictors of importance by classifying different types of sentence compressions/expansions appearing in the Wikipedia data, and moving from a generative to a discriminative model. Furthermore, we plan to use the data to learn other types of sentence rewriting such as more flexible paraphrases and grammatical reordering.

Finally, at the document level, our initial experiments on summarization are an encouraging first step of a fuller study of how to take advantage of Wikipedia's article revisions for training summarization algorithms.

Acknowledgments

This work was partially supported by a Google research award. We thank Stuart Shieber and the anonymous reviewers for their suggestions.

References

Carlson, A. J.; Rosen, J.; and Roth, D. 2001. Scaling up context sensitive text correction. In *IAAI2001*, 45–50.

Collins, M. 1997. Three generative, lexicalized models for statistical parsing. In Cohen, P. R., and Wahlster, W., eds., *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, 16–23. Somerset, New Jersey: Association for Computational Linguistics.

Dagan, I.; Glickman, O.; and Magnini, B. 2005. The PASCAL recognising textual entailment challenge. In Candela, J. Q.; Dagan, I.; Magnini, B.; and d'Alché Buc, F., eds., *MLCW*, volume 3944 of *Lecture Notes in Computer Science*, 177–190. Springer.

Dolan, B.; Quirk, C.; and Brockett, C. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*.

Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Hall, P. A. V., and Dowling, G. R. 1980. Approximate string matching. *ACM Comput. Surv.* 12(4):381–402.

Knight, K., and Marcu, D. 2000. Statistics-based summarization - step one: Sentence compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, 703–710. AAAI Press / The MIT Press.

Kupiec, J.; Pederson, J.; and Chen, F. 1995. A Trainable Document Summarizer. In *SIGIR 1995*, 68–73. Washington, WA, USA: ACM.

Langkilde, I. 2000. Forest-based statistical sentence generation. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, 170–177. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Turner, J., and Charniak, E. 2005. Supervised and unsupervised learning for sentence compression. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 290–297. Morristown, NJ, USA: Association for Computational Linguistics.

Viégas, F. B.; Wattenberg, M.; and Dave, K. 2004. Studying cooperation and conflict between authors with *history flow* visualizations. In Dykstra-Erickson, E., and Tscheligi, M., eds., *CHI*, 575–582. ACM.

Witten, I., and Bell, T. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory* 37(4).

Yamangil, E., and Nelken, R. 2008. Mining wikipedia revision histories for improving sentence compression. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Zobel, J., and Dart, P. W. 1996. Phonetic string matching: Lessons from information retrieval. In Frei, H.-P.; Harman, D.; Schäble, P.; and Wilkinson, R., eds., *Proceedings of the 19th International Conference on Research and Development in Information Retrieval*, 166–172. Zurich, Switzerland: ACM Press.

Okinet: Automatic Extraction of a Medical Ontology From Wikipedia

Vasco Calais Pedro[1], Radu Stefan Niculescu[2], Lucian Vlad Lita[2]
[1] Language Technologies Institute, Carnegie Mellon University
[2] Siemens Medical Solutions

Abstract

The medical domain provides a fertile ground for the application of ontological knowledge. Ontologies are an essential part of many approaches to medical text processing, understanding and reasoning. However, the creation of large, high quality medical ontologies is not trivial, requiring the analysis of domain sources, background knowledge, as well as obtaining consensus among experts. Current methods are labor intensive, prone to generate inconsistencies, and often require expert knowledge. Fortunately, semi structured information repositories, like Wikipedia, provide a valuable resource from which to mine structured information. In this paper we propose a novel framework for automatically creating medical ontologies from semi-structured data. As part of this framework, we present a Directional Feedback Edge Labeling (DFEL) algorithm. We successfully demonstrate the effectiveness of the DFEL algorithm on the task of labeling the relations of Okinet, a Wikipedia based medical ontology. Current results demonstrate the high performance, utility, and flexibility of our approach. We conclude by describing ROSE, an application that combines Okinet with other medical ontologies.

Motivation

In the last decades the number of available medical ontologies has grown considerably. These ontologies enable the use of previous medical knowledge in a structured way. Applications of medical ontologies include: more effective search of patient records, hospital quality improvement programs, (semi)automatic ICD-9 coding for insurance reimbursement, preliminary symptom-based diagnosis, ambiguity reduction when choosing medical tests, and classification of diseases, symptoms, and other medical concepts. For example, when trying to answer whether a patient was prescribed *Aspirin* (for hospital quality improvement measures), one needs to consider similar terms (such as *Ecotrin*, *Bayer pain reliever*, etc). Also, when performing (semi)automatic patient ICD-9 coding, it is useful to map conditions that can be described in various ways (*Heart Attack* can be also stated as *AMI* or *MI* or *Myocardial Infarction* or simply *Infarction*). For preliminary diagnosis at the point of care, ontologies can help by quickly returning diseases that have a given set of symptoms (instances of symptoms and diseases are concepts related by the "symptom of" relationship).

Several proprietary and public efforts such as MESH(Lipscomb 2000) and SNOMED(Spackman, Campbell et al. 1997) have become available and UMLS(Bodenreider and Journals) is rapidly becoming a de facto standard for medical ontologies, containing more than 100 dictionaries. Other medical ontologies include: RadLex (Radiology Information Resource)**,** OMIM (Online Mendelian Inheritance in Man)**,** MEDCIN (medical terminology), LOINC (Logical Observation Identifiers Names and Codes) and ICD-9/ICD-10 Codes.

At the same time, large sources of encyclopedic knowledge are becoming readily available in wiki like form. Resources such as Wikipedia(Denoyer and Gallinari 2006), the largest collaboratively edited source of encyclopedic knowledge(Krotzsch, Vrandecic et al. 2005; Völkel, Krötzsch et al. 2006), Scholarpedia(Izhikevich 2007), Citizendium(Sanger 2007) and the recently launched, incipient Google Knols Project are examples of semi-structured encyclopedic knowledge bases that provide a natural way to collect human knowledge(Lih 2003), with the advantage of naturally solving, to a large degree, the problem of consensus.

These resources represent an intermediate step between unstructured text and structured knowledge and are seen as potential viable sources of knowledge for automatic construction of medical ontologies.

In this paper we propose a general framework to mine structured knowledge from Wikipedia and apply it to the creation of a medical ontology. The paper proceeds as follows: we first discuss related work, and then describe the general framework for building a medical ontology from Wikipedia. We demonstrate our Directional Feedback Edge Labeling algorithm on a task of labeling the relations in Okinet, a Wikipedia based medical ontology. We conclude with a description of the Okinet browser as well as some interesting and promising ideas for future work.

Related Work

Maedche(Maedche and Staab 2002; Maedche 2002) and Navligli et al.(Navigli, Velardi et al. 2003) explored semi-automatic methods for concept and relation extraction, focusing on building ontologies from broad domain documents. Blake and Pratt(Blake and Pratt 2002) worked on extracting relationships between concepts from medical texts. Khoo et al.(Khoo, Chan et al. 2002) matched graphical patterns in syntactic parse trees in order to look for causal relations.

Several pieces of previous work focused on the link structure of Wikipedia to derive structure. Kozlova(Kozlova 2005) mined the link structure in Wikipedia for document classification. Milne et al.(Milne, Medelyan et al. 2006) used the basic link structure to construct domain specific thesauri and applied it to the agriculture domain. Bhole et al.(Bhole, Fortuna et al. 2007) used document classification techniques to determine appropriate documents in Wikipedia that were later mined for social information (people, places, organizations and events).

The Wikipedia Structure

Wikipedia general structure consists of an *article name*, which is unique within the particular wiki structure and thus suitable for a *concept name*, and *links* connecting articles, which are suggestive of semantic relations between them. Each article is typically divided in sections and sometimes contains tables that synthesize information pertinent to the article.

Within the different types of inter-article links, we often find *redirects* (articles that consist solely of a link to another article) and when we find this type of link we can interpret the two concepts described by those articles as synonyms. Each article is normally inserted into one or more categories, thus creating a set of hierarchical relations.

Even though each link seems to carry semantic information between two concepts, only a small percentage is typically used in mining Wikipedia, namely the *redirects* and *categories*. The main challenge of this work is to assign the correct semantic label to the extracted links deemed of interest, when the link is not a redirect.

General Methodology

We propose that we should take an inclusive approach rather than a selective approach to create a medical ontology, where we start by including all the article names as concepts and all the existing links as potential relations. We subsequently rely on extracted features to assign labels, finally discarding links without labels.

The goal is to first create a directed unlabeled graph that mimics the link structure, use the extracted features to generate a small amount of labeled data and run a Directional Feedback Edge Labeling Algorithm to extend the labels to the rest of the links, discarding the links with confidence below a preset threshold.

Feature Extraction

For every link extracted we store a set of features that are associated with that link. The set of features consists of the following:

Document Title

The title of the document where the link was found. This corresponds to the source concept.

Section Header Path

The path composed of the sections up to the section where the link was found. E.g. Diagnosis → Symptoms.

Context

The context surrounding the link. This consists of the 3 words before and after the link.

Link Type

The type of link. This can be *redirect*, *anchor*, *category* or *regular*.

Part of List

Binary feature that is positive if the link was found within a list, such as – *Fatigue*, *Headache*, *Nausea*, *Vomiting*.
In Table 1 we show an example of the information that the extraction of one link generates.

Sample Feature Extraction	
Concept	**fever**
Document Title	**Influenza**
Header Path	**Symptoms and diagnosis > Symptoms**
Context	**Extreme coldness and fever**
Link Type	**regular**
Part of List	**yes**

Table 1. Sample Feature Extraction

Even though we extracted five features, for the purposes of this work, we used only three features. We expect to use the remaining features in future work for the purpose of increasing performance.

Generating Labeled Data

Once we process the entire Wikipedia, we have a directed unlabeled graph where each edge represents a relation between two concepts. For each edge we also have a set of associated features.

After we decide the set of labels we are interested in, we use a combination of heuristics to bootstrap the labeling process. Besides using the redirect anchor and category links to label synonyms and hypernyms, we rely on the following two strategies.

List Based Labeling

Uses articles that list concepts and assigns labels to the instances of those lists that are under corresponding sections. E.g. if we find fever under the section symptoms in article flu and fever is also in the list of medical symptoms article, then we assign symptom as label for the link between flu and fever.

Context Based Labeling

Assigns the section title as label if the context shows that the link is displayed within a list. E.g. If we find *–fever, headache and nausea* under the section *symptoms* under article *flu*, we assign *symptom* as a label for the link between *flu* and *fever*.

After the bootstrapping process, we have a directed graph with a partially labeled relation set. In the next section we introduce the Directional Feedback Edge Labeling Algorithm which starts with a small such set of labeled links and uses graph probability propagation to label the remaining links/relations in the ontology.

Directional Feedback Edge Labeling Algorithm

The Directional Labeling Algorithm relies on neighboring edge trends and directionality to update probabilities of possible labels that can be assigned to an unlabeled relation. The steps of this algorithm are described in Algorithm 1.

Each unlabeled edge starts with equal probability of label assignment. At each iteration, in STEP 1, for each node we update the probabilities of the labels of the outgoing edges by smoothing them with the overall probability distribution of labels over the outgoing edges of that node (essentially multiplying the two probability distributions). This assures we take into account both our current belief about that edge and the overall information contained in the edges going out of that node. To give an intuition why both types of information are important, consider the example in Figure 1. The dashed and the dotted edges represent edges which were labeled during the bootstrapping phase. The dashed edges represent label *SymptomOf* and the dotted edges represent label *Treats*. The solid edges are unlabeled and therefore it is natural to assume that, in the absence of other information, each label is equally likely. However, based on the already labeled outgoing edges at C_1, the unlabeled edge (C_1,C_{10}) has a 2/3 probability to have label *SymptomOf* and 1/3 probability to have label *Treats*. Therefore, our initial belief of the edge (C_1,C_{10}) needs to be updated by incorporating this new information.

For each node C, perform Steps 1 and 2, then repeat until convergence.

STEP 1. Let p_{ik} be the probability of the i^{th} outgoing edge (out of n possible) from node C to have the k^{th} label (out of m possible labels). Update the outgoing edge probabilities:

$$P_{ik} \leftarrow \frac{P_{ik} \times \sum_{j=1}^{n} P_{jk}}{\sum_{l=1}^{m} \left(P_{il} \times \sum_{j=1}^{n} P_{jl} \right)}$$

STEP 2. Update the incoming edge probabilities similar to the previous step.

STEP 3. Once convergence is reached via the above two steps, assign the maximum probability label to an edge as long as this probability is higher than a predefined threshold.

Algorithm 1. Directional Feedback Edge Labeling.

In STEP 2, we then perform the same procedure for each node, but based on incoming edges. Because an edge is an incoming edge for a node and an outgoing edge for another, the label probability distribution for that edge is influenced by the label distributions in both its endpoints. Therefore, after a number of iterations, the label probabilities can be influenced by other label probabilities at any distance in the graph.

Back to the example in Figure 1, the edge (C_1,C_{10}) has a 2/3 probability to be labeled *SymptomOf* if we look only at the outgoing edges from C_1 whereas it has a probability of 1 to be labeled *Treats* if we look only at the incoming edges to C_{10}. This justifies the need to perform the same operation for both incoming and outgoing edges. The need to perform both steps iteratively is twofold: to assure convergence and to allow knowledge to propagate across the network.

After convergence, we select only the edges with labels above a predefined threshold and discard the rest as unreliably labeled.

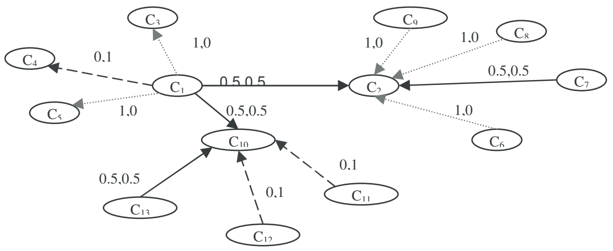


Figure 1. Example of Directional Labeling

UMLS and Okinet

UMLS is perhaps the most important medical ontology currently available. It uses a semantic network to combine the knowledge contained in the set of available dictionaries and allows for easy access to a set of standard ontological relations. The work of mapping the vocabularies demands a large human effort and is very time consuming. Due to the structure of UMLS, certain semantic relations exist only at the semantic network level. This means that in UMLS we are not able to determine symptoms of particular diseases, but rather between classes of concepts. For example, we are not able to find out what are the *symptoms_of flu*, but rather what categories of concepts could represent symptoms of *flu*, which is a problem.

Okinet thus exist as complement to UMLS, allowing the rapid and automatic creation of relations at the instance level, which enables the use of inference processes using both ontologies.

Experimental Setting

In order to test our approach, we used Wikipedia as a test case, even though the methodology could be applied to any other wiki like resource. Our goal is to create an ontology of causes, treatments, symptoms, diagnoses and side effects.

We started by selecting all the concepts contained in the *list of diseases* article, which contains 4000+ diseases and syndromes. We then expanded our article set to include all the articles that linked or were linked to by any of the articles contained in the current set.

Next we performed the feature extraction process followed by the bootstrapping procedure. The results were manually checked to create a gold standard set. This resulted in an ontology with 4308 concepts and 7465 relations divided as depicted in Figure 2.

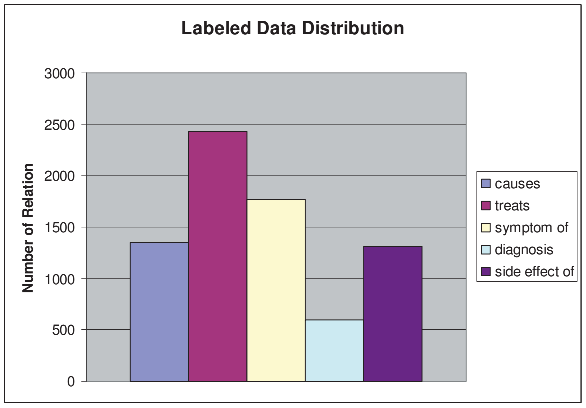


Figure 2. Distribution of relation labels.

Results

We experimented using small percentages of the labeled data as a training seed for the Directional Edge Feedback Labeling algorithm while considering the remaining edges unlabeled. The results of the labeling algorithm were then compared with the original labels. In our experiments, we varied both the percentage of the labeled training data (seed size) as well as the threshold above which we assign a label. We evaluated the results using precision and recall: Precision The percentage of label assignments that were correctly assigned to the proper class.

Recall The percentage of possible labels that were correctly assigned.

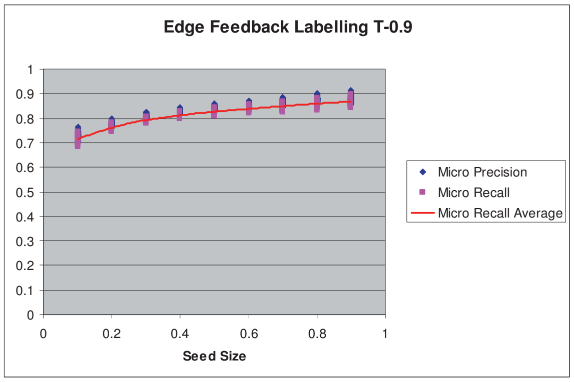


Figure 3. Algorithm performance with threshold 0.9 and variation on the seed size

In Figure 3 we can see the results of varying the size of labeled seed set at a threshold for label assignment at 0.9, which means that we only assign a label with high confidence. Even though we are only showing micro precision and micro recall, the results for macro precision and recall were very similar and were thus not presented for simplicity purposes. Each point in the average line represents 100 hundred runs with a labeled seed size of the indicated value. The precision and recall average vary between 70% and 90% while seeds vary from 10% to 90% of the total labeled set.

Even though the results are very promising, we explored ways to boost the results at small seed sizes. Due to the propagation nature of this algorithm, by stopping after a few iterations, we are in fact preventing long-range labeled edge influences and therefore we can restrict the process of labeling an edge to local neighborhood in the graph. Figure 4 shows the results of stopping the labeling algorithm after two iterations. Given the number of iterations, only edges with a fast convergence rate will update the probabilities distribution enough to get assigned a label. This means that the higher the threshold, the more accuracy we get, even though the recall is sharply reduced. This variation is particularly useful in situations where the precision is more important than recall. Using this technique we would be able to extend the labeled data set with highly accurate labels.

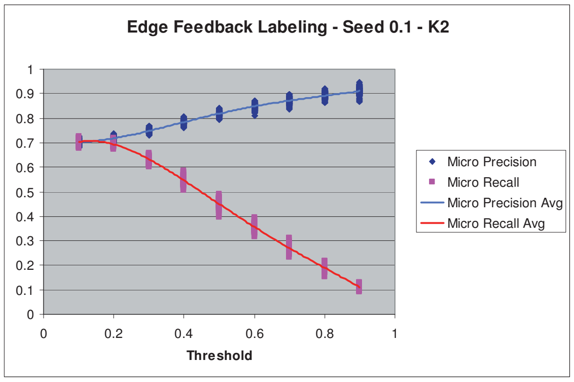


Figure 4. Precision and recall with 10% seed size, algorithm stopped after two iterations and varying the assignment threshold.

Finally we looked at our algorithm as way to reduce uncertainty. Figure 5 shows the results of taking the two highest confidence labels for each edge and considering as correct if either of the assigned labels is correct.

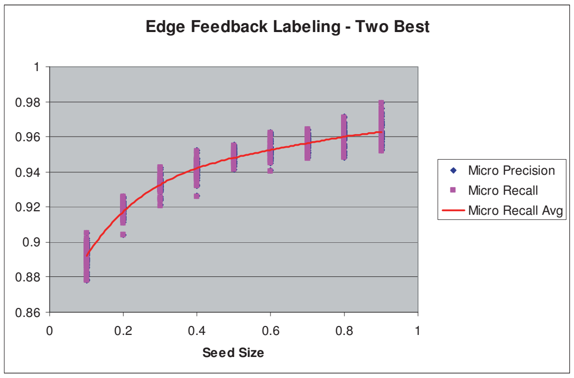


Figure 5. Precision and Recall when considering looking at the two labels with the highest probability

The Rose System

The current focus is on integration of multiple ontologies in the medical domain for immediate use. We currently have integrated UMLS , Okinet and Wordnet (Miller 1995) as a proof of concept application. We have built an ontology search system that uses a federated approach to ontology search as described in (Pedro, Lita et al. 2007) and produced a simple interface for ontology search. We call this integrated system ROSE, the Remind Ontology Search Engine. The goal of ROSE is to allow for rapid access to the ontological knowledge in the ontologies contained therein. In particular, the target domain for ROSE is the medical domain, where there are multiple ontologies, lexicons, and general graph-based resources covering specific parts of the medical knowledge space. In this scenario, a federated ontology search engine is desperately needed.

Web Based Interface

Although ROSE works in a typical server fashion, which can be queried directly using xml, we have also created a visualization tool for ontology browsing. Given the fact that the server is querying several ontologies, possible using resources span several servers, having a localized copy of everything is unfeasible. We therefore opted for the creation of a web interface to visualize the results of ontology queries.

The web interface allows for querying ROSE for synonyms and medical relations from all the available ontologies. It enables the user to visualize the desired relations across ontologies in a graph display and browse subsequent relations with ease and simplicity. It uses AJAX and JSP with JSON as a communication protocol. Below is a snapshot of the result of querying rose for Congestive Heart Failure. We can see the results from Wordnet, UMLS and Okinet represented by different colors.

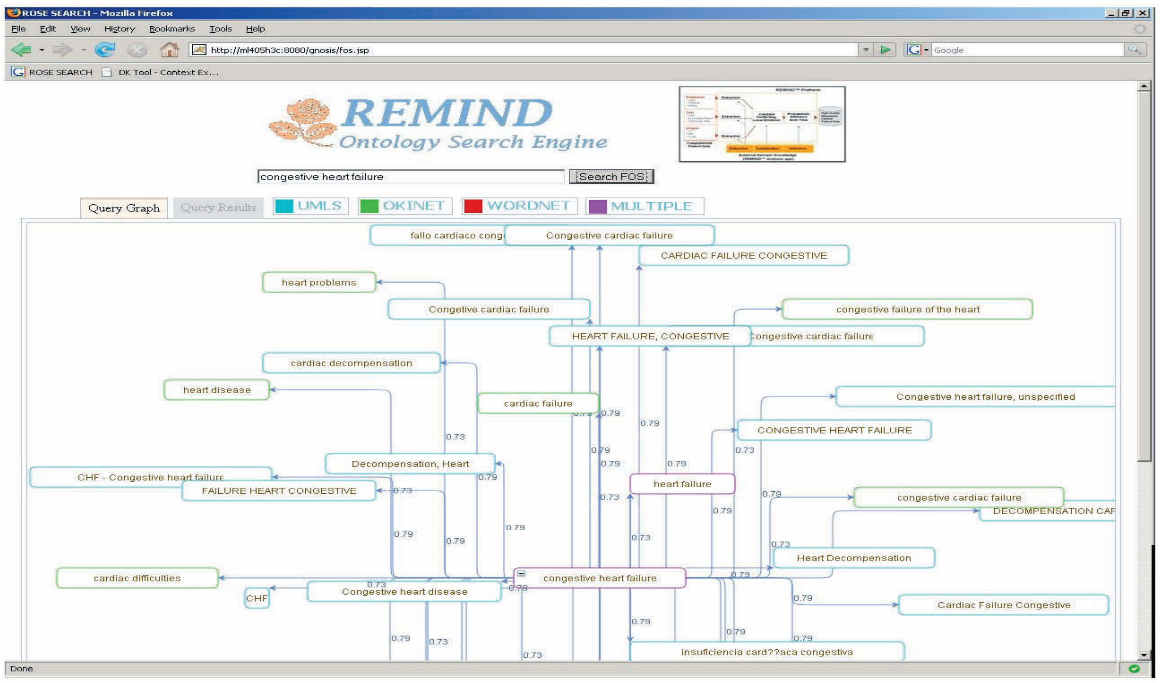
Summary and Future Work

The creation of medical ontologies is a complex and challenging task, in that it requires the analysis of domain sources, background knowledge and obtaining consensus among creators. The current methods are labor intensive and prone to generate inconsistencies. In this paper we propose a novel methodology for creating medical ontologies automatically from Wikipedia. We have measured the precision and recall of our method using a set of experiments and demonstrated the flexibility and utility of our approach. Given that our experiments were performed on a small dataset obtaining high performance, we believe that the described method has the potential to achieve even higher performance on larger datasets, further alleviating the amount of work required for ontology creation and further speeding up the process of knowledge acquisition.

Multiple research avenues remain open for domain-specific ontologies extracted from semi-structured data. In particular for the medical domain, Wikipedia is proving to be a continuously updated source of knowledge. In future work, we would like to incorporate additional features we have already extracted into the edge labeling algorithm. These features have the potential of boosting performance, especially with smaller seed sets.

We also intend to extend our medical ontology to additional concepts and relation types. Complex relations in the medical domain are abundant and higher coverage would allow sophisticated user to explore Okinet beyond frequent, often encountered medical phenomena.

Of particular interest studying how applicable is our method to automatic ontology mapping. Graph-based algorithms are particular well suited to this task, provided sufficient training data or sufficiently large seeds. We are



interested in reducing these seed sets while maintaining high performance.

Another interesting direction for future work is the question of whether seed localization has an impact on the performance of the labeling algorithm.

Finally, since our Okinet generation method does not rely on rule-based components, nor has been infused with medical expert knowledge, a promising research direction consists of extending our approach to non-medical domains such as finance, education, physics, which have at least a moderate wiki presence.

References

Bhole, A., B. Fortuna, et al. 2007. *Mining Wikipedia and relating named entities over time*. Bohanec, M., Gams, M., Rajkovic, V., Urbancic, T., Bernik, M., Mladenic, D., Grobelnik, M., Hericko, M., Kordeš, U., Markic, O. Proceedings of the 10 th International Multiconference on Information Societz IS 8: 12.

Blake, C. and W. Pratt 2002. *Automatically Identifying Candidate Treatments from Existing Medical Literature*. AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases.

Bodenreider, O. and O. Journals. *The Unified Medical Language System (UMLS): integrating biomedical terminology*. Nucleic Acids Research 32(90001): 267-270.

Denoyer, L. and P. Gallinari 2006. *The Wikipedia XML corpus*. ACM SIGIR Forum 40(1): 64-69.

Izhikevich, E. M. 2007. *Scholarpedia, the free peer-reviewed encyclopedia*. from http://www.scholarpedia.org/.

Khoo, C., S. Chan, et al. 2002. *Chapter 4*. The Semantics of Relationships: An Interdisciplinary Perspective.

Kozlova, N. 2005. *Automatic ontology extraction for document classification*, Saarland University.

Krotzsch, M., D. Vrandecic, et al. 2005. *Wikipedia and the Semantic Web-The Missing Links*. Proceedings of Wikimania.

Lih, A. 2003. *Wikipedia as Participatory Journalism: Reliable Sources? Metrics for evaluating collaborative media as a news resource*. Nature 2004.

Lipscomb, C. E. 2000. *Medical Subject Headings (MeSH)*. Bull Med Libr Assoc 88(3): 265-266.

Maedche, A. and S. Staab 2002. *Measuring similarity between ontologies*. Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW): 251–263.

Maedche, A. D. 2002. *Ontology Learning for the Semantic Web*, Kluwer Academic Publishers.

Milne, D., O. Medelyan, et al. 2006. *Mining Domain-Specific Thesauri from Wikipedia: A Case Study*. Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence: 442-448.

Navigli, R., P. Velardi, et al. 2003. *Ontology learning and its application to automated terminology translation*. Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications] 18(1): 22-31.

Sanger, L. 2007. *Citizendium*. from www.citizendium.org.

Spackman, K. A., K. E. Campbell, et al. 1997. *SNOMED RT: A reference terminology for health care*. Proc AMIA Annu Fall Symp 640(4): 503-512.

Völkel, M., M. Krötzsch, et al. 2006. *Semantic Wikipedia*. Proceedings of the 15th international conference on World Wide Web: 585-594.

Automatic Vandalism Detection in Wikipedia: Towards a Machine Learning Approach

Koen Smets and **Bart Goethals** and **Brigitte Verdonk**
Department of Mathematics and Computer Science
University of Antwerp, Antwerp, Belgium
{koen.smets,bart.goethals,brigitte.verdonk}@ua.ac.be

Abstract

Since the end of 2006 several autonomous bots are, or have been, running on Wikipedia to keep the encyclopedia free from vandalism and other damaging edits. These expert systems, however, are far from optimal and should be improved to relieve the human editors from the burden of manually reverting such edits. We investigate the possibility of using machine learning techniques to build an autonomous system capable to distinguish vandalism from legitimate edits. We highlight the results of a small but important step in this direction by applying commonly known machine learning algorithms using a straightforward feature representation. Despite the promising results, this study reveals that elementary features, which are also used by the current approaches to fight vandalism, are not sufficient to build such a system. They will need to be accompanied by additional information which, among other things, incorporates the semantics of a revision.

Introduction

Since the inception of Wikipedia in 2001, the free encyclopedia, which is editable by everyone, has grown rapidly to become what it is today: one of the largest sources of adequate information on the Internet. This popularity translates itself to an ever growing large amount of articles, readers consulting them, editors improving and extending them … and unfortunately also in the number of acts of vandalism committed a day. By vandalism we understand every edit that damages the reputation of articles and/or users of Wikipedia. Priedhorsky et al. (2007) provide a survey of the typical categories of damages together with an empirically determined likeliness of occurrence. We list them here in decreasing order of appearance: introducing nonsense, offenses or misinformation; the partial deletion of content; adding spam (links); mass deletion of an article …

To fight vandalism, Wikipedia relies on the good faith of its users that accidentally discover damaged articles and, as in practice turns out, on the time-consuming efforts of its administrators and power users. To ease their job, they use special tools like Vandal Fighter to monitor the recent changes and which allow quick reverts of modifications matching regular expressions that define bad content or are performed by users on a blacklist. Since the end of 2006 some vandal bots, computer programs designed to detect and revert vandalism have seen the light on Wikipedia. Nowadays the most prominent of them are ClueBot and VoABot II. These tools are built around the same primitives that are included in Vandal Fighter. They use lists of regular expressions and consult databases with blocked users or IP addresses to keep legitimate edits apart from vandalism. The major drawback of these approaches is the fact that these bots utilize static lists of obscenities and 'grammar' rules which are hard to maintain and easy to deceive. As we will show, they only detect 30% of the committed vandalism. So there is certainly need for improvement.

We believe this improvement can be achieved by applying machine learning and natural language processing (NLP) techniques. Not in the very least because machine learning algorithms have already proven their usefulness for related tasks such as intrusion detection and spam filtering for email as well as for weblogs.

The remainder of this paper is as follows. First, we give a brief overview of related work, followed by a motivation for using machine learning to solve the problem. Next, we complement the most recent vandalism studies by discussing the performance results of the bots currently active on Wikipedia. Thereafter, we present the preliminary results of using a Naive Bayes classifier and a compression based classifier on the same features that serve as raw input for those bots. Finally, we formulate conclusions and outline the approach we plan to investigate next.

Related Work

Wikipedia has been subject to a statistical analysis in several research studies. Viégas, Wattenberg, and Dave (2004) make use of a visualization tool to analyze the history of Wikipedia articles. With respect to vandalism in particular, the authors are able to (manually) identify mass addition and mass deletion as jumps in the history flow of a page. Buriol et al. (2006) describe the results of a temporal analysis of the Wikigraph and state that 6 percent of all edits are reverts and likely vandalism. This number is confirmed by Kittur et al. (2007) in a study investigating the use of reverting as the key mechanism to fight vandalism. They also point out that only looking for reverts explicitly signaling vandalism is not strict enough to find evidence for most of the vandalism in the history of articles. The most recent study, to the best of our knowledge, by Priedhorsky et al. (2007) categorizes the different types of vandalism and their occurrence rate in a subset of 676 revision chains that were reverted. They confirm that reverts explicitly commented form a good approximation to spot damages, with a precision and recall of respectively 77% and 62%. Our work complements this last one, as we investigate a yet more recent version of the English Wikipedia history, and also analyse the decisions made by two bots. We also try to respond to the authors' request to investigate the automatic detection of damage. The authors believe in intelligent routing tasks, where automation directs humans to potential damage incidents but where humans still make the final decision.

There is strong cross-pollination possible between Wikipedia and several research areas. Wikipedia can benefit from techniques from the machine learning, information retrieval and NLP domains in order to improve the quality of the articles. Adler and de Alfaro (2007) build a content-driven system to compute and reflect the reputation of authors and their edits based on the time span modifications remain inside an article. Priedhorsky et al. (2007) use a closely related measure but they do not take into account the lifetime but the expected viewing time to rate the value of words. Rassbach, Pincock, and Mingus (2007) explore the feasibility of automatically rating the quality of articles. They use a maximum entropy classifier to distinguish six quality classes combining length measures, wiki specific measures (number of images, in/out links . . .) and commonly used features to solve NLP problems (part-of-speech usage and readability metrics). The problem to detect damages is related to ours in the sense that we need to rate the quality of a single revision instead of the whole article. The cross-pollination also holds for the other way around as machine learning, information retrieval and NLP can benefit from the use of Wikipedia. Gabrilovich and Markovitch (2007) use an explicit semantic interpreter built using articles from Wikipedia which is capable of measuring the semantic relatedness between text documents.

Recently, Potthast, Stein, and Gerling (2008) also use machine learning to detect vandalism in Wikipedia. Compared to their work, we have a larger labeled data set, use different classifiers, and most importantly, use different features. We aim to summarize an edit by focusing on the difference between the new and old version of an article, while Potthast, Stein, and Gerling use a set of 15 features that quantify the characteristics of vandalism.

Vandalism Detection and Machine Learning

The particular task to detect vandalism is closely related to problems in computer security: intrusion detection or filtering out spam from mailboxes and weblogs. It is a specific kind of web-defacement, but as the accessibility allows everyone to contribute, there is no need for crackers breaking into systems. We can see it as a form of content-based access control, where the integrity constraint on Wikipedia enforces that "All article modifications must be factual and relevant" as stated by Hart, Johnson, and Stent (2007). The problem also shares characteristics intrinsic to computer security problems. We need to deal with a skew and ever changing class distribution as the normal edits outnumber vandalism and both vandalism and legitimate edits are likely to change, due to respectively the adversarial environment and the rise of new articles or formatting languages.

Machine learning provides state of the art solutions to closely related problems. We put two techniques from the world of spam detection to the test. On one hand we use a well-known Naive Bayes classifier and on the other hand, as results from Naive Bayes models are significantly improved by state-of-the-art statistical compression models, a classifier based on probabilistic sequence modeling provided by Bratko et al. (2006).

Although we are aware that we will not be capable of identifying all types of vandalism (e.g. detecting misinformation in the pure sense is regarded as impossible without consulting external sources of information), we believe that machine learning might cope with this interesting, but far from trivial, problem.

Performance Analysis of Bots on Wikipedia

In this section we complement the work done by Priedhorsky et al. (2007) by analysing the results of the bots on one hour of data from the English version of Wikipedia. We show that there is still significant room for improvement in the automatic detection of vandalism. Furthermore, we provide additional evidence that the labeling procedure based on edit reverts, is quite sound. Next, we introduce the Simple English Wikipedia and present the results of a modified version of ClueBot on this data set, which we also use in our machine learning experiments later on. We start however with a short introduction to ClueBot's inner working.

ClueBot

ClueBot, written by Carter (2007), uses a number of simple heuristics to detect a subset of the types of vandalism mentioned above. First, it detects page replaces and page blanks relying on an auto-summary feature of MedaWiki software. Next, it categorizes mass delete, mass addition and small changes based on absolute difference in length. For the last three types, vandalism is determined by using a manually crafted static score list with regular expressions specifying the obscenities and defining some grammar rules which are hard to maintain and easy to by-pass. Negative scores are given to words or syntactical constructions that seem impossible in good articles, while wiki links and wiki transcludes are considered as positive. The difference between the current and the last revision is calculated using a standard *diff* algorithm. Thereafter, the inserted and deleted sentences are analysed using the score list and if this value exceeds a certain threshold vandalism is signaled. ClueBot further relies on the user whitelist for trusted users and increases its precision by only reverting edits done by anonymous or new users.

English Wikipedia (enwiki)

We analyse one hour of data from the first of March 2008 (00:00:00 - 00:59:59), restricting ourselves to the recent

	legitimate	reverted	mislabeled
1 hour	6944	323	26 (8.00%)
5 hours	28 312	1 926	

	ClueBot	VoABot II
1 hour	68 (22.89%)	33 (11.11%)
5 hours	349 (18.12%)	154 (8.00%)

Table 1: Edit statistics on English Wikipedia (2008.03.01).

changes of pages from the main namespace (0), the true encyclopedic articles, and ignore revisions from user or talk and discussion pages.

The data is automatically labeled by matching revision comments to regular expressions that signal a revert action, i.e. an action which restores a page to a previous version. This approach closely resembles the identification of the set of revisions denoted by Priedhorsky et al. (2007) as Damaged-Loose, a superset of the revisions explicitly marked as vandalism (Damaged-Strict).

While labeling based on commented revert actions is a good first order approximation, mislabeling cannot be excluded. If we regard vandalism as the positive class throughout this paper, then there will be both false positives and false negatives. The former arises when reverts are misused for other purposes than fighting vandalism like undoing changes without proper references or prior discussion. The latter occurs when vandalism is corrected but not marked as reverted in the comment, or when vandalism remains undetected for a long time. Estimating the number of mislabelings is very hard and manual labeling is out of question, considering the vast amount of data.

From the total of 6944 revisions, 4.65% are considered vandalism. Manual inspection demonstrates that of these 323, 11 are mislabeled as vandalism and for 15 others we are in doubt. So in the worst case we have to cope with a false positive rate of 8%.

Of the correctly labeled acts of vandalism 68 are identified by ClueBot and 33 by VoABot II, the two active vandal fighting bots on Wikipedia nowadays. Together this corresponds to a recall of 33%. Hence the bulk of the work is still done by power users and administrators. All vandalism identified by the two bots is true vandalism so the precision during this one hour is 100%.

Priedhorsky et al. (2007) identify that around 20% of their labeled data is misinformation, a number confirmed by our manual inspection. Even disregarding those, the above analysis reveals there is much room for improvement with respect to the recall. Numerical analysis on a data set including the next four hours, see Table 1, shows that these numbers remain invariant, as they are only multiplied by a factor of 5.

Simple English Wikipedia (simplewiki)

As a proof of concept and because of storage and time constraints, we run the preliminary machine learning experiments on Simple English Wikipedia, a user-contributed online encyclopedia intended for people whose first language is not English. This encyclopedia is much smaller in size compared to the standard English Wikipedia as shown in Table 2. There are no bots in operation that try to remove spam or vandalism. Nevertheless the articles are also subject to vandalism, which often last longer as fewer readers and users are watching the pages.

	pages	revs	xml.bz2
simplewiki	53 449	499 395	88.7 MB
enwiki	11 405 052	167 464 014	133.0 GB

Table 2: Size (Simple) English Wikipedia expressed in terms of the total number of pages, revisions and the compressed file size of the pages-meta-history files available at `http://download.wikimedia.org`.

period	nr (%) of vandalism	revs	pages
2003 - 2004	21 (1.12)	1 870	784
2004 - 2005	276 (2.03)	13 624	2541
2005 - 2006	2 194 (5.60)	39 170	6626
2006 - 2007	12 061 (8.33)	144 865	17 157
2007 - ...	12 322 (6.96)	177 165	22 488
2003 - ...	26 874 (7.13)	376 694	28 272

Table 3: Estimated vandalism statistics of the revisions together with the number of revisions and pages from the main namespace in Simple English Wikipedia (2007.09.27).

We work with the dump from 2007.09.27 and again we only consider the main articles disregarding pages from other namespaces. Labeling using the same procedure shows that the amount of vandalism, as we see in Table 3, is fairly stable and comparable with the percentage on enwiki.

As a reference, we provide the performance of a modified version of ClueBot on the simplewiki data set in Table 4. We use our own implementation based on the source code of the one running at enwiki, with that difference that we only consider the heuristics to detect vandalism and do not take into account the dynamic user whitelist.

We notice in Table 4 a drop in both precision and recall. The former can possibly be explained by not using the dynamic user white list, while the fact that the static score list of the ClueBot is manually tailored towards the English Wikipedia could explain the drop in recall. A more thorough study, including manually analysing the decisions of the ClueBot, is required before we can further explain the decreased performance.

	ACC	PRE	REC	F_1
2003 - 2004	0.9752	0.1250	0.0476	0.0689
2004 - 2005	0.9722	0.2577	0.0905	0.1340
2005 - 2006	0.9346	0.4185	0.0761	0.1288
2006 - 2007	0.9144	0.6207	0.1306	0.2158
2007 - ...	0.9320	0.6381	0.1774	0.2777
2003 - ...	0.9270	0.6114	0.1472	0.2372

Table 4: Performance of ClueBot (without user whitelist) on Simple English Wikipedia.

delete	Vandalism is almost always a crime; different types of vandalism include: graffiti, smashing the windows of cars and houses, and rioting. {{stub}}
insert	Being *** is almost always a crime; different types of **** *** include: doggy style.
change delete	Vandalism property vandal graffiti website vandals funny attention vandal Vandals
change insert	******* of as *** **** *** **** site **** *** *** *** ****** ***_*******
comment	
user group	anonymous

Table 5: Censored feature list of revision 29853 from the Vandalism page in Simple Wiki English.

Experiments

In this section, we will discuss the setting for our machine learning experiment conducted on simplewiki, the Simple English version of Wikipedia. We first consider the data representation. Thereafter we give a brief description of two learning algorithms put to test: a Naive Bayes classifier on bags of words (BOW) and a combined classifier built using probabilistic sequence modeling (Bratko et al. 2006), also referred to in the literature as statistical compression models.

Revision Representation

In this case study we use the simplest possible data representation. As for ClueBot and VoABot II, we extract raw data from the current revision and from the history of previous edits. This first step could be seen as making the static scoring list of ClueBot dynamic. This should provide a baseline for future work. In particular, for each revision we use its text, the text of the previous revision, the user groups (anonymous, bureaucrat, administrator . . .) and the revision comment. We also experimented with including the lengths of the revisions as extra features. The effect on overall performance is however minimal and thus we discard them in this analysis. Hence the focus lies here more on the content of an edit.

As the modified revision and the one preceding it differ slightly, it makes sense to summarize an edit. Like ClueBot, we calculate the difference using the standard *diff* tool. Processing the output gives us three types of text: lines that were inserted, deleted or changed. As the changed lines only differ in some words or characters from each other, we again compare these using *wdiff*. Basically, this is the same as what users see when they compare revisions visually using the MediaWiki software. Table 5 gives an example of the feature representation used throughout this paper, applied to a vandalized revision.

To evaluate our machine learning experiments we use 60% of the labeled data for training and the remaining 40% for evaluation purposes. We do not aim to statistically analyse the different approaches but use it more as a guide to conduct our search towards a machine learning based vandalism detection tool.

BOW + Naive Bayes

As a first attempt we use the Naive Bayes implementation from the ‘Bow’ toolkit (McCallum 1996) as learning mechanism to tackle the problem. This tool treats each feature as a bag of words and uses Porter’s stemming algorithm and stop word removal to decrease the size of the feature space. Next, we train a Naive Bayes classifier on each of the features separately. Our final classifier combines the results of the individual classifiers by multiplying the obtained probability scores.

Probabilistic Sequence Modeling

Probabilisitic sequence modeling (PSM) forms the foundation of statistical compression algorithms. The key strength of compression-based methods is that they allow constructing robust probabilistic text classifiers based on character-level or binary sequences, and thus omit tokenization and other error-prone pre-processing steps. Nevertheless, as clearly stated by Sculley and Brodley (2006), they are not a “parameter free” silver bullet for feature selection and data representation. In fact they are concrete similarity measures within defined feature spaces. Commonly used statistical compression algorithms are dynamic Markov compression (DMC) and prediction by partial matching (PPM), both described in detail by Bratko et al. (2006). Basically these are n-gram models where weights are implicitly assigned to the coordinates during compression. Empirical tests, in above references, show that compression by DMC and PPM outperforms the explicit n-gram vector space model due to this inherent feature weighting procedure. For the implementation we use PSMSlib (Bratko 2006), which uses the PPM algorithm.

During the training phase a compression model M_c^f is built (Bratko et al. 2006) for each feature f in Table 5 and for each class c (vandalism or legitimate). The main idea is that sequences of characters generated by a particular class will be compressed better using the corresponding model. In theory, an optimal compression can be achieved if one knows the entropy given that model. In order to classify a revision r, we estimate for each of its feature values x the entropy H by calculating,

$$H_c^f(r) = \frac{1}{|x|} \log \prod_{i=1}^{|x|} p(x_i | x_{i-k}^{i-1}, M_c^f),$$

where $p(x_i | x_{i-k}^{i-1}, M_c^f)$ is the probability assigned by model M_c^f to symbol x_i given its k predecessors. In order to score the revision, we combine all features by summing over the entropies,

$$S_c(r) = \sum_f H_c^f(r)$$

and then calculating the log ratio

$$S(r) = \log \frac{S_{van}(r)}{S_{leg}(r)}.$$

If the value S exceeds a prespecified threshold, default 0, we assign the revision to the vandalism class otherwise we consider it as legitimate. The threshold parameter trades off the precision and the recall.

Analysis and Discussion

In this section we discuss the results of the two attempts to put machine learning to work on the Simple English data set.

BOW + Naive Bayes

Table 6 shows the results on the test set of the final Naive Bayes classifier only taking into account the revision diff features as bags of words. Table 7 shows the same, this time including the user group information together with revision comments. While the precision in these tables is almost the same as in Table 4, a significant increase can be noticed in terms of recall and F_1, especially when including user group information and comment.

Table 8 shows the results on the whole data set of the classifiers based on a single feature ignoring the probability of the class priors. This provides more insight in the influence of the different features.

As expected, we see that the '(change) delete'-feature contributes little more than noise, while the 'change insert' is the most decisive factor. Next, we observe a seemingly important contribution of the 'change delete'-feature with respect to the recall. This may be due to the fact that some pages are vandalised more than others. It is, however, not a decisive feature as it contributes little to the overall result in terms of precision.

The domination of the 'user group'-feature on the recall can be easily explained by combining the facts that anonymous users commit most of the vandalism, but that their overall legitimate contribution to Wikipedia is rather small.

Note that when ignoring the probability of the class prior in the Naive Bayes classifier on all features, as shown by the last line in Table 8, the recall is higher but at the same time there is a drop in the precision.

	ACC	PRE	REC	F_1
2003 - 2004	0.9748	0.4000	0.4444	0.4210
2004 - 2005	0.9648	0.3007	0.3603	0.3278
2005 - 2006	0.9235	0.3701	0.2941	0.3278
2006 - 2007	0.9266	0.6975	0.3266	0.4449
2007 - ...	0.9310	0.5949	0.1960	0.2948
2003 - ...	0.9303	0.6166	0.2503	0.3561

Table 6: Results Naive Bayes using the revision diff features in a BOW.

Probabilistic Sequence Modeling

Table 9 shows the overall performance together with the results of the individual models on the same test set. Interesting to note is that the recall is much higher, but that the precision drops unexpectedly. We lack a plausible explanation for this strange behaviour, but the effect can be diminished by setting the threshold parameter to a score higher than zero. This is shown in Figure 1, where we plot the precision/recall curves for varying thresholds for the probabilistic sequence models and for the Naive Bayes models, both with and without user groups and comments. The marks show the results when the log ratio threshold is equal to 0. The tendency is that, despite the worse behavior shown in Table 9, the overall accuracy measured in term of precision and recall is better for the compression based models than for the bag of words model using Naive Bayes.

	ACC	PRE	REC	F_1
2003 - 2004	0.9794	0.5000	0.4444	0.4705
2004 - 2005	0.9635	0.2937	0.3783	0.3307
2005 - 2006	0.9165	0.3427	0.3439	0.3433
2006 - 2007	0.9261	0.6161	0.4800	0.5396
2007 - ...	0.9342	0.5911	0.3453	0.4359
2003 - ...	0.9314	0.5882	0.3694	0.4538

Table 7: Results Naive Bayes including user group information and revision comments.

2003 - ...	ACC	PRE	REC	F_1
delete	0.8618	0.1476	0.2813	0.1936
insert	0.9585	0.2636	0.2670	0.2653
change delete	0.5002	0.1079	0.5307	0.1794
change insert	0.9068	0.6486	0.2068	0.3136
comment	0.8729	0.2360	0.2894	0.2600
user groups	0.8444	0.3102	0.8319	0.4520
	0.9057	0.4181	0.5667	0.4812

Table 8: Results individual Naive Bayes classifiers (ignoring class priors).

2003 - ...	ACC	PRE	REC	F_1
delete	0.1568	0.0809	0.9567	0.1493
insert	0.5031	0.1274	0.9281	0.2241
change delete	0.2891	0.0805	0.7867	0.1461
change insert	0.5028	0.1177	0.8362	0.2064
	0.8554	0.3117	0.7201	0.4351
comment	0.7978	0.2667	0.9233	0.4138
user groups	0.8460	0.3171	0.8598	0.4633
	0.8436	0.3209	0.9171	0.4755

Table 9: Results Probabilistic Sequence Modeling classifiers.

To boost the overall performance we will need additional information. We believe that incorporating weighted semantics derived from explicit semantic analysis, as described by Gabrilovich and Markovitch (2007), is necessary. The intuition is that the semantics of offenses, nonsense and spam are likely to differ from the semantics of the revised article and hence are an important feature for classification. Moreover, we believe that the 'text deleted'-feature contains more

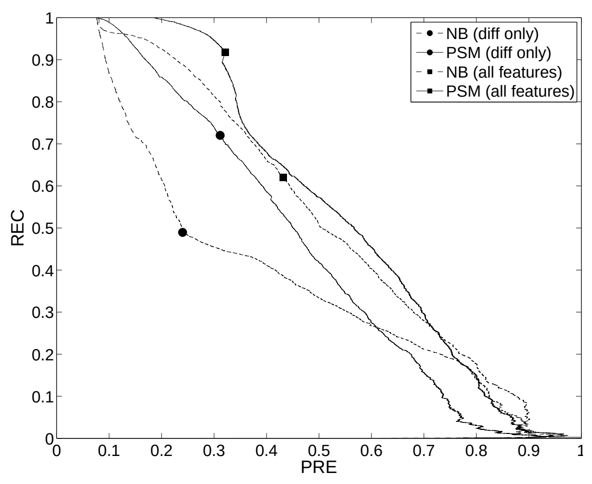


Figure 1: Precision/Recall curves: Naive Bayes versus Probabilistic Sequence Modeling for revision diff features with(out) user groups and comment.

information than is apparent from the current results, where it appears to be merely a noise factor. To exploit the usefulness of this feature, we will take into account its effect on the semantic level by measuring the text life, i.e. the value of the deleted words, as suggested by Adler and de Alfaro (2007).

Conclusions and Future Work

As far as we know, we are among the first to try machine learning techniques to answer the need of improving the recall of current expert systems, which are only capable of identifying 30% of all vandalism. We demonstrate that, by applying two machine learning algorithms, a straight forward feature representation and using a set of noisy labeled examples, the accuracy of the actual running bots can be improved. We feel confident that this study is merely a starting point and that there is much room for improvement. In the end almost all vandalism that is not related to misinformation should be detectable automatically, without consulting third-party information.

For future work, we will combine the ideas from Gabrilovich and Markovitch (2007) and Adler and de Alfaro (2007) to enhance the feature representation. We aim to rebuild their explicit semantic interpreter and use it for semantic comparison between the current modified revision and the previous versions of an article. We will compare the concepts related to text inserted and deleted, and weight these features using respectively the authority of authors and the value of words expressed in text life or expected viewing rate. In this context, we plan to compare our effort to the work of Potthast, Stein, and Gerling (2008).

Acknowledgements

We thank Walter Daelemans for suggesting statistical compression models, Jacobi Carter and Aaron Schulz for providing the source code of ClueBot and VoAbot II. Koen Smets is supported by a Ph. D. fellowship of the Research Foundation - Flanders (FWO).

References

Adler, B. T., and de Alfaro, L. 2007. A Content-Driven Reputation System for the Wikipedia. In *Proceedings of the 16th International World Wide Web Conference (WWW)*. ACM Press.

Bratko, A.; Cormack, G. V.; Filipič, B.; Lynam, T. R.; and Zupan, B. 2006. Spam Filtering using Statistical Data Compression Models. *Journal of Machine Learning Research* 7(Dec):2673–2698.

Bratko, A. 2006. PSMSLib: Probabilistic Sequence Modeling Shared Library. Available at `http://ai.ijs.si/andrej/psmslib.html`.

Buriol, L. S.; Castillo, C.; Donato, D.; Leonardi, S.; and Stefano, M. 2006. Temporal Analysis of the Wikigraph. In *Proceedings of the IEEE/WCIC/ACM International Conference on Web Intelligence (WI)*, 45–51.

Carter, J. 2007. ClueBot and Vandalism on Wikipedia. Unpublished. Available at `http://24.40.131.153/ClueBot.pdf`.

Gabrilovich, E., and Markovitch, S. 2007. Computing Semantic Relatedness using Wikipedia-Based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 1606–1611.

Hart, M.; Johnson, R.; and Stent, A. 2007. Content-Based Access Control. Submitted to the IEEE Symposium on Privacy and Security.

Kittur, A.; Suh, B.; Pendleton, B. A.; and Chi, E. H. 2007. He Says, She Says: Conflict and Coordination in Wikipedia. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

McCallum, A. K. 1996. Bow: a Toolkit for Statistical Language Modeling, Text Retrieval, Classification and Clustering. Available at `http://www.cs.cmu.edu/~mccallum/bow`.

Potthast, M.; Stein, B.; and Gerling, R. 2008. Automatic Vandalism Detection in Wikipedia. In *Proceedings of the 30th European Conference on IR Research (ECIR)*, 663–668.

Priedhorsky, R.; Chen, J.; Lam, S. T. K.; Panciera, K.; Terveen, L.; and Riedl, J. 2007. Creating, Destroying, and Restoring Value in Wikipedia. In *Proceedings of the International ACM Conference on Supporting Group Work*.

Rassbach, L.; Pincock, T.; and Mingus, B. 2007. Exploring the Feasibility of Automatically Rating Online Article Quality. In *Proceedings of the International Wikimedia Conference (Wikimania)*. Wikimedia.

Sculley, D., and Brodley, C. E. 2006. Compression and Machine Learning: a New Perspective on Feature Space Vectors. In *Proceedings of the Data Compression Conference (DCC)*, 332–341.

Viégas, F. B.; Wattenberg, M.; and Dave, K. 2004. Studying Cooperation and Conflict between Authors with *history flow* Visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Enriching the Crosslingual Link Structure of Wikipedia - A Classification-Based Approach -

Philipp Sorg and **Philipp Cimiano**
Institute AIFB, University of Karlsruhe,
D-76128 Karlsruhe, Germany
{sorg,cimiano}@aifb.uni-karlsruhe.de

Abstract

The crosslingual link structure of Wikipedia represents a valuable resource which can be exploited for crosslingual natural language processing applications. However, this requires that it has a reasonable coverage and is furthermore accurate. For the specific language pair German/English that we consider in our experiments, we show that roughly 50% of the articles are linked from German to English and only 14% from English to German. These figures clearly corroborate the need for an approach to automatically induce new cross-language links, especially in the light of such a dynamically growing resource such as Wikipedia. In this paper we present a classification-based approach with the goal of inferring new cross-language links. Our experiments show that this approach has a recall of 70% with a precision of 94% for the task of learning cross-language links on a test dataset.

Introduction

From the natural language processing perspective, a very interesting feature of Wikipedia, besides the overwhelming amount of content created daily, is the fact that information is linked across languages. This is accomplished via so called *cross-language links* mapping articles in one language to equivalent articles in another language. Obviously, such links have a natural application in cross-lingual natural language processing, e.g. in machine translation, cross-lingual information retrieval, projection of information across languages, alignment etc.

However, if natural language processing applications are expected to exploit the cross-language link structure, it should have enough coverage. A first analysis of the coverage for one language pair, i.e. German/English, shows that only a percentage of the pages are connected via such cross-language links. Thus, in this article we present a novel method for learning additional cross-language links in order to enrich Wikipedia. The method is based on a classification-based approach which classifies pairs of articles of two different languages as connected by a cross-language link or not. The features used by the classifier range from a simple calculation of the edit distance between the title of the articles over word overlap counts through to more complex link patterns as features. The results of the approach are encouraging as they show a prediction recall of 70% with a precision of 94% on the task of finding the corresponding article in another language.

Given our encouraging results, we have started processing the German Wikipedia. We will provide the additional links as download for the research community.

The remainder of this article is structured as follows: First we motivate our approach and analyse the availability of cross-language links for the language pair German/English. The core of our approach is explained and its assumptions motivated quantitatively. Then the classification-based approach and the used features are described in detail. Afterwards we present the experimental results. Before concluding we discuss related work.

Motivation

As stated above, the cross-language links in Wikipedia can be used for various cross-lingual NLP tasks. But in order to be able to perform these tasks, the cross-language link structure should be consistent and needs to have enough coverage.

In the context of this paper, we have chosen the German and English Wikipedia and computed statistics about the German/English cross-lingual link structure to get a clear picture about its consistency and coverage.

These findings motivate our approach to learning new cross-language links in Wikipedia.

Statistics about German/English Cross-Language Links

For the analyis of the German and English Wikipedia we counted the absolute number of articles in the English and German Wikipedia, the number of cross-language links between the English and German Wikipedia and classified these links into bidirectional links, links with no backlink and links with backlink to another article[1]. Articles are defined as Wikipedia pages that are not redirect[2] pages and are in the default namespace. Cross-language links ending

[1]E.g.: "3D rendering" to "3D-Computergrafik" back to "3D computer graphics"

[2]Redirect Pages are used to disambiguate different surface forms, denominations and morphological variants of a given unambiguous NE or concept to a unique form or ID.

	Articles	Cross-Language Links		
English Wikipedia	2,293,194	English $\longrightarrow$ German (EN2DE)	321,498	14.0%
German Wikipedia	703,769	German $\longrightarrow$ English (DE2EN)	322,900	45.9%

	EN2DE C.-L. Links		DE2EN C.-L. Links	
Bidirectional links	303,684	94.5%	303,684	94.1%
No backlink	9,753	3.0%	12,303	3.8%
Backlink to another article	7,845	2.4%	6,132	1.9%

Table 1: Statistics on the English (October 18, 2007) and German (October 09, 2007) Wikipedia Corpus.

in redirect pages were resolved to the corresponding article. All the results of the analysis are presented in Table 1.

The results show that only a small fraction (14%) of articles in the English Wikipedia is linked to articles in the German Wikipedia. The fraction of German articles linked to English articles is much bigger, but with 45.9% it is still less than half of all articles in the German Wikipedia. For some articles there may not be a corresponding article in another language due to the local context of the specific country. But as this is probably not the case for half of the German Wikipedia, there is still a big margin to learn new meaningful cross-language links.

As the fraction of bidirectional links is around 95% in the English and German Wikipedia, the consistency of cross-language links seems to be good. This motivates to use them in a bootstrapping manner to find new cross-language links.

Chain Link Hypothesis

One problem in learning new cross-language links between the German and English Wikipedia is the huge number of pages (see number of articles in Table 1). It will surely not be possible to use a classifier on all article pairs, such that a preselection of candidate articles seems appropriate.

In order to preselect a number of relevant articles, we rely on the *chain link hypothesis*. This hypothesis builds on the notion of a chain link:

Definition 1 *For two Wikipedia databases WP$_\alpha$, WP$_\beta$ with corresponding languages α, β, a* **chain link (CL)** *between two articles $A_\alpha \in$ WP$_\alpha$ and $A_\beta \in$ WP$_\beta$ is defined as the following link structure:*

$$A_\alpha \xrightarrow{pl} B_\alpha \xrightarrow{ll} B_\beta \xleftarrow{pl} A_\beta$$

with $B_\alpha \in$ WP$_\alpha$ and $B_\beta \in$ WP$_\beta$. Pagelinks between articles are displayed as $\xrightarrow{pl}$ and cross-language links between articles in different languages as $\xrightarrow{ll}$. The articles B_α and B_β are called **chain link intermediate articles (CLIA)**.

An example for such a chain link between a German and an English article is visualized in Figure 1. The article “Horse” ($= A_\alpha$) in the English Wikipedia is connected through the displayed chain link to the article “Hauspferd” ($= A_\beta$) in the German Wikipedia. The articles “Mammal” ($= B_\alpha$) and “Säugetiere” ($= B_\beta$) are CLIAs of this chain link that is formed by the pagelink from “Horse” to “Mammal”, the cross-language link from “Mammal” to “Säugetiere” and the pagelink from “Hauspferd” to “Säugetiere”.

Based on chain links we formulate the chain link hypothesis, the basic hypothesis for the selection of candidates for new cross-language links: *Every article is linked to its corresponding article in another language through at least one chain link.*

In order to empirically verify the plausibility of the above hypothesis, we have generated the RAND1000 dataset containing 1000 random articles of the German Wikipedia with existing cross-language links to the English Wikipedia. For all articles in the RAND1000 dataset, we have checked if the hypothesis is indeed fulfilled. For an article A_α in the dataset, connected to the article A_β in the English Wikipedia through a cross-language link, this means that we have to check if A_β is in the *candidate set* $\mathcal{C}(A_\alpha)$. The candidate set of an article A_α are all articles that are connected to A_α through at least one chain link.

However, we noticed that on average the number of articles in each candidate set is still to big. In case of the RAND1000 dataset the mean size of the candidate set is 153,402. This means that an approach to find a cross-language link for an article A, that considers all articles in $\mathcal{C}(A)$ as potential candidates, can be very expensive from a computational point of view.

Thus, we also consider a reduction of the number of candidates. Therefore we define the *support* of a candidate C in respect to an article A in the dataset as the number of existing chain links between A and C. For each article A, we limit the number of candidates to less than 1000 by requiring a minimal support via an appropriate threshold. For each article, we call the set of these candidates the *restricted candidate set* $\mathcal{C}'(A)$, which is restricted by definition to at most 1000 candidates. The following table contains the percentage of articles for which the chain link hypothesis is fulfilled using the full candidate set and the restricted candidate set:

	Percentage
Full candidate set	95.7 %
Restricted candidate set	86.5 %

This means that for 95.7% of pages in the RAND1000 dataset the corresponding article in the English Wikipedia is included in the full candidate set. For the restricted candidate set the hypothesis holds for 86.5% of the pages. With respect to the decrease in performance time by processing at most 1000 instead of 153,402 articles on average, this decrease in terms of best case accuracy seems a good trade-off.

Overall, the chain link hypothesis is therefore strongly supported by this evaluation on the RAND1000 dataset,

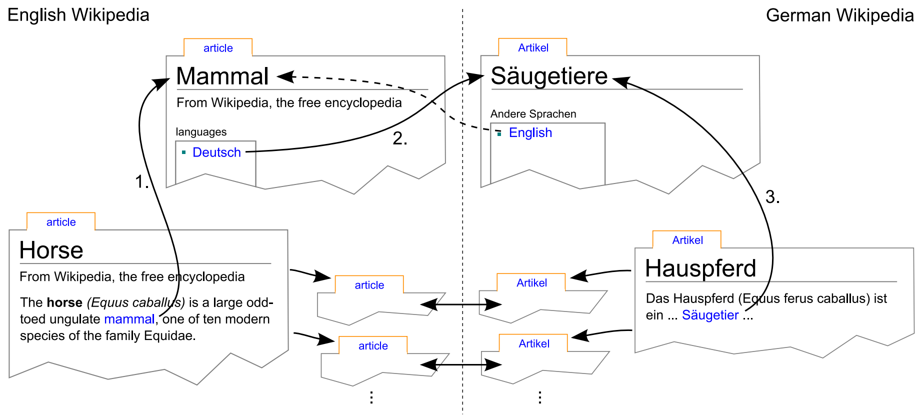

English Wikipedia
German Wikipedia
article
Mammal
From Wikipedia, the free encyclopedia
languages
Deutsch
2.
Artikel
Säugetiere
Andere Sprachen
English
1.
3.
article
Horse
From Wikipedia, the free encyclopedia
The horse (Equus caballus) is a large odd-toed ungulate mammal, one of ten modern species of the family Equidae.
article
Artikel
Artikel
Hauspferd
Das Hauspferd (Equus ferus caballus) ist ein ... Säugetier ...
article
Artikel

Figure 1: Visualisation of a chain link that is used to find candidate pages for new cross-language links. 1. is a pagelink in the English Wikipedia, 3. a pagelink in the German Wikipedia and 2. a cross-language link from the English to the German Wikipedia.

even after restricting the candidate set to at most 1000 candidates for each article. Based on these findings the usage of the chain link hypothesis to restrict the set of candidate articles for new cross-language links seems to be promising. The approach presented in the remainder of this paper strongly relies on the chain link hypothesis as a feature for training a classifier which is able to predict whether a pair of articles in two languages (German/English in our case) should be connected via a cross-language link or not. Having motivated our approach and the underlying hypothesis empirically, we describe the approach in more detail in the next section.

Classification-based Approach

The main idea behind our approach to learn new cross-language links is to train a classifier which is able to predict whether a pair of articles (A, B) where $A \in \text{WP}_\alpha$ and $A \in \text{WP}_\beta$ should be cross-linked. As it is not feasible to apply the articles to all pairs in two languages, for the article A we only consider the candidates $\mathcal{C}'(A) \subset \text{WP}_\beta$ as potential cross-links.

As classifier we used the popular Support Vector Machine (SVM) implementation *SVMlight* by Joachims (1999) with a linear kernel function. The classifier is trained with a number of features which we describe below in more detail. Features are defined on article-candidate pairs $(A, C) \in \text{WP}_\alpha \times \text{WP}_\beta$ with $C \in \mathcal{C}'(A)$ and are based on different information sources. Based on our chain link hypothesis, the support of C in respect to A, defined above as the number of chain links between these articles, is considered and the link structure of the CLIAs is exploited. In addition, the categories of A and C are also considered. As categories are also linked by language links it is possible to align categories across languages. Finally, we also use simple features based on the title and text of articles.

Feature Design

The features can be classified into two classes: graph-based and text-based features. The former are based on different link types in Wikipedia, i.e. pagelinks, category links and language links. The latter are based on the title and text of the Wikipedia articles.

For the definition of graph-based features, we need to define the number of inlinks of an article. Inlinks of an article $A \in \text{WP}_\alpha$ are pagelinks from another article that are targeted to A. The number of inlinks of A is therefore defined as $\text{INLINKS}(A) = |\{B \in \text{WP}_\alpha \mid B \xrightarrow{pl} A\}|$.

For the definition of text-based features we need to introduce the *Levenshtein Distance* (Levenshtein, 1966), a string metric that is based on the edit distance between two strings. The edit distance is defined as the minimal number of insert, delete and replace operations that is needed to transform one string to another. We use a version of the Levenshtein Distance that is normalized by the string lengths.

As described above, features are based on article-candidate pairs. In the following, we will refer to the article as A and to the candidate as C with $C \in \mathcal{C}'(A)$.

Graph-based Features:

Feature 1 (*Chain Link Count Feature*)
This feature is equal to the support of C with respect to A.

Feature 2 (*Normalized Chain Link Count Feature*)
This feature is the value of Feature 1 normalized by the support threshold that was used to restrict the candidate set for A.

Featureset 3 (*Chain Link Inlink Intervals*)
Given an article A and a candidate C we compute all the chain links between these and classify them into 20 intervals defined over the number of inlinks that the CLIA of the chain link has, i.e. we classify a CLIA B into a bucket according to the value $\text{INLINKS}(B)$. Thus, we yield 20 features corresponding to the 20 intervals.
The motivation behind this classification is the assumption that chain links containing CLIAs with fewer inlinks are probably more specific for a topic and therefore more important for the choice of the correct article.

By classifying the chain links into different classes using the number of inlinks of the CLIAs this assumption can be explored by the classifier.

Feature 4 (*Common Categories Feature*)
The output of this feature is the number of common categories of two articles in different languages. Common category means that both articles are member of categories that are linked through existing cross-language links.

Feature 5 (*CLIA Graph Feature*)
This feature is based on a similarity measure on graphs. Given two graphs G_α and G_β on the same set of vertices, the similarity is defined as the number of common edges of these graphs normalized by the number of vertices. For the article A and the candidate C, the graphs G_α and G_β are defined on the set of chain links between A and C as vertices. Edges in G_α between two chain links exist if the CLIAs in WP_α of these chain links are linked by a pagelink in WP_α. Analogous, edges in G_β between two chain links exist, if the CLIAs in WP_β of these chain links are linked by pagelink in the WP_β. The value of this feature is the value of the defined similarity measure between G_α and G_β.

Text-based Features:

Feature 6 (*Editing Distance Feature*)
The output of this feature is the normalized Levenshtein Distance on the titles of the candidate articles pair.

Feature 7 (*Text Overlap Feature*)
This feature computes the text overlap between the text of the candidate article pair. To remain independent of lexical resources there is no translation involved. This feature will be useful if the articles for example share many named entities.

Evaluation

The evaluation is based on the RAND1000 dataset. As described above, this dataset consists of 1000 articles of the German Wikipedia with an existing language link to an article in the English Wikipedia.

In the following we first analyse this dataset to get a lower bound for the classification experiment. Afterwards we describe the experimental setup. Finally we present further results on articles without an existing language link.

Baseline

In order to find a lower bound for recall, we define a simple method to find language links by matching the titles of articles. The recall of this method on the RAND1000 dataset is equal to the percentage of articles that are linked to English articles with identical title. The analysis of the RAND1000 dataset showed that 47.0% of the articles in this dataset are linked to English articles with identical title. The reason for this high share is the fact that many Wikipedia articles describe named entities and thus have the same title in different languages. This value defines a lower bound for recall as this method to find new language links is very simple and straightforward. Any other method should exceed the results of this baseline.

Evaluation of the RAND1000 Dataset

In the experiments we used a random 3-1-split of the RAND1000 dataset. The first part containing 750 articles was used for training the classifier. The remaining 250 articles were used for the evaluation.

In order to evaluate the correctness of our approach, we consider the TOP-k with $k \in \{1..5\}$ candidates with respect to a ranking determined on the basis of the example's (directed) distance from the SVM-induced hyperplane. The larger the distance, the higher is the classifier's certainty that it is a positive example. Hereby, we do not distinguish between positive examples, which have a positive distance to the margin and negative examples, which have a negative one. Thus, it is possible that in absence of positive examples, also negative examples appear at the top of the ranking.

TOP-k Evaluation As quality measure for the TOP-k evaluation we defined TOP-k-Accuracy as the share of articles in the test set for which the correct linked article was part of the k top ranked candidates [3].

One important problem in learning the classifier is the discrepancy between positive and negative training data. For every article in the training set there exists at most one positive example but up to 1000 negative examples. Using all this training data will most likely yield a classifier which always predicts new examples to belong to the majority class, the negative examples in our case (compare Provost (2000)). In order to avoid this, the training data has to be balanced, such that we only used a portion of the negative examples in order to train the classifier. For each article in the training set, 2, 5 and 10 negative examples were randomly selected and together with all positive examples were used to train the classifier.

To be able to measure the quality of different features we trained the classifier with different feature sets. First we used only the *Chain Link Count Feature*. In this case candidate articles with a higher number of chain links are ranked higher. The purpose of the results of this experiment is to support the hypothesis that chain links are a prominent clue for language links between articles. In another set of experiments we used the text features only as well as the graph features only, respectively. This allows to assess the influence of each of the different features. Finally, the classifier was trained with all features to find out if it is indeed worth considering all the features together.

Results of the experiments are shown in Table 2. The table shows the accuracy with respect to the top k candidates with varying sizes of negative examples considered. Overall it seems that the choice of negative/positive ratio does not have a strong impact on the results. However further experiments showed that using too many negative examples leads to learning a trivial classifier as is the case when using the chain link count feature alone for a negative/positive ratio of 10:1. A negative/positive ratio of 5:1 seems therefore reasonable and will be used in the further experiments described below. The accuracy of the prediction, when considering only the chain link features, ranges from 42.4% (TOP-

[3] TOP-k-Accur. = $\frac{|\{A_\alpha \in \text{RAND1000} \mid \exists A_\beta \in \text{TOP-}k(A_\alpha) : A_\alpha \xrightarrow{ll} A_\beta\}|}{|\text{RAND1000}|}$

Ratio -/+ data	Feature selection		TOP-k-Accuracy TOP-1	TOP-2	TOP-3	TOP-4	TOP-5
2:1	1	(*Chain Link Count Feature*)	42.4%	51.2%	60.0%	62.8%	64.8%
	6-7	(*Text features*)	68.4%	71.2%	73.6%	74.8%	75.2%
	1-5	(*Graph features*)	54.8%	64.0%	68.4%	70.8%	72.0%
	1-7	(*All features*)	71.2%	76.0%	78.8%	79.6%	80.0%
5:1	1	(*Chain Link Count Feature*)	42.4%	51.2%	60.0%	63.2%	64.8%
	6-7	(*Text features*)	68.8%	72.8%	74.4%	74.8%	75.2%
	1-5	(*Graph features*)	55.2%	62.8%	67.6%	68.8%	70.0%
	1-7	(*All features*)	74.8%	**79.2%**	**79.2%**	80.0%	80.4%
10:1	1	(*Chain Link Count Feature*)	0.0%	0.4%	0.4%	0.4%	0.4%
	6-7	(*Text features*)	68.4%	72.4%	74.4%	74.8%	75.2%
	1-5	(*Graph features*)	55.6%	62.4%	67.6%	69.2%	70.4%
	1-7	(*All features*)	**76.0%**	78.4%	78.8%	**80.4%**	**81.2%**

Table 2: Results of the evaluation on the RAND1000 dataset. The first column describes the negative/positive ratio of training examples. The second column describes the feature selection. TOP-k-Accuracy is used as quality measure.

1) to 64.8% (Top-5). Considering the TOP-1 results, we conclude that the classifier trained with the chain link features alone does not improve with respect to our baseline of 47% consisting of considering articles with the same title. The text and graph features alone yield results in terms of accuracy between 68.8% (TOP-1) and 75.2% (TOP-5) as well as 55.2% (TOP-1) and 70% (TOP-5). Both types of features thus allow to train a classifier which outperforms the naive baseline. Considering all features yields indeed the best results, leading to a prediction accuracy of between 76% (TOP-1) and 81.2% (TOP-5). Thus, we have shown that the number of chain links seems to be the weakest predictor for a cross-language link between two articles in isolation. When considering all features, the results certainly improve, showing that the number of chain links crucially contributes towards making a good decision in combination with the other features used. As we use articles from the English and German Wikipedia as test data, the text features based on text overlap and similarity are strong features with good classification results. However, even using only graph features, thus operating on a completely language-independent level, the results exceed the trivial baseline. Thus, we can assume that our method will produce reasonable results for any language pair of Wikipedia, even if they use different alphabets or if their languages are from different linguistic families. In those cases the text based features will play a negligible role.

Best Candidate Retrieval In order to automatically induce new language links, it is necessary to choose exactly one candidate for each source article and to decide whether this candidate is the corresponding article or not. To achieve these goals we define Best Candidate Retrieval as a modified TOP-1-Retrieval which selects that positive example which has the largest (positive) margin with respect to the SVM-induced hyperplane. This differs from the TOP-k retrieval introduced above in that the latter one performs a ranking on the basis of distance to the discriminating hyperplane, also considering examples on the "wrong side" of the plane. The Best Candidate Retrieval produced the following results:

Ratio -/+ data	Feature selection	Recall	Precision
10:1	All features	69.6%	93.5%

The recall of this experiment is 22.6% higher than the lower bound. Due to the preselection of candidates, the maximum recall is 86.5%. It is important to note that a recall of 69.6% means that we find 80% of the cross-language links that can be found at all given our preselection on the basis of the candidates' support.

As our aim is to learn correct links, high precision is a requirement. In this sense our approach seems very promising as new language links are learned with high precision of 93.5% and a reasonable recall. It could therefore be used to enrich the Wikipedia database with new language links.

Learning New Language Links

In order to test our approach in a "real scenario" with the aim of inducing new cross-language links instead of merely reproducing the existing ones, we have started processing the German Wikipedia, considering all those articles which do not have an existing cross-language link to the English Wikipedia. As our algorithms are still in a state of research prototype and as we do not have the computational power it was not possible for us to process all of these articles. Because of that we defined a relevance ranking on the articles based on the number of incoming pagelinks and sorted the articles according to this ranking. We processed the first 12,000 articles resulting in more than 5,000 new cross-language links according to best candidate retrieval as described above. The file with the results can be downloaded from our website [4].

The first 3,000 links were manually evaluated. As for 2,198 links the titles were identic, these links were assumed to be correct. The remaining 802 links were evaluated by 3 independent persons. They annotated them as correct links, wrong links and links between related articles. The annotator's correlation was reasonable with a Pearson's product-moment correlation coefficient between 0.80 and 0.84. As

[4] `http://www.aifb.uni-karlsruhe.de/WBS/pso/learned_language_links_(German-English).tsv`

overall result we got a precision of 81.9% for learning correct cross-language links. Further, the manual evaluation showed that 92.2% of the links connected at least related articles. These are very satisfactory results.

Related Work

Several authors have considered exploiting the cross-language link structure of Wikipedia for cross-lingual natural language applications. Adafre & de Rijke (2006) have for example used the language links to find similar sentences across languages. They have also considered discovering additional links in Wikipedia (Adafre & de Rijke, 2005). However, the latter approach only aimed to add additional links to articles within the same language. Based on earlier results showing that multilingual resources such as EuroWordNet can be used for cross-language Question Answering (see Ferrández & Ferrández (2006)), the same authors have shown that using Wikipedia in addition to EuroWordnet can even improve results on the cross-language Question Answering task (see Ferrández *et al.* (2007)). The reason is that Wikipedia contains more complete and up-to-date information about named entities. Other researchers have shown that the multilingual information in Wikipedia can be successfully used to improve a cross-lingual information retrieval system (see Schönhofen *et al.* (2007)). Very recently, Wentland et al. have considered the cross-lingual link structure of Wikipedia to extract multilingual contexts for named entities contained in Wikipedia. Such multilingual contexts can then be used for the disambiguation of named entities across multiple languages (Wentland *et al.*, 2008).

To the best of our knowledge, we are not aware of any approach aiming at finding new cross-language links in Wikipedia. However, such an approach would be beneficial for all of the cross-lingual applications mentioned above.

Conclusion

We have presented an approach for inducing new cross-language links for Wikipedia. Such links can be beneficial for any cross-language natural language processing task exploiting Wikipedia as source of multilingual knowledge. Our approach works for language pairs for which a number of cross-language links are already available and bootstraps on the basis of these existing links to discover new ones. No other lexical resources are needed. We have shown that our method achieves a satisfactory level of recall of around 70% and a high level of precision of around 94%. These results hold for that subset of Wikipedia pages which have been already linked across languages. To get a better estimate of the accuracy of the approach, we started to induce new cross-language links for articles in the German Wikipedia without a cross-language link to an article in the English Wikipedia and manually evaluated the first 3000 learned links. The results of this evaluation show that around 82% of the links are correct and that 92% of the links connect at least related articles. For a productive use of our methods, the algorithm needs to be optimized from a computational point of view. On a standard dual core computer using a MySQL database, the extraction of the candidates for the RAND1000 dataset and the computation of all features took 26 hours. Most expensive are the selection of candidates and the computation of graph features. The computational costs could therefore possibly be reduced by optimizing the database and by identifying the most relevant graph features. However this remains future work.

Acknowledgements

This work was funded by the Multipla project sponsored by the German Research Foundation (DFG) under grant number 38457858 and by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978.

References

Adafre, S., and de Rijke, M. 2005. Discovering missing links in wikipedia. In *Proceedings of the 3rd International Workshop on Link Discovery*.

Adafre, S., and de Rijke, M. 2006. Finding similar sentences across multiple languages in wikipedia. In *Proceedings of the EACL Workshop on New Text, Wikis, Blogs and Other Dynamic Text Sources*.

Ferrández, S., and Ferrández, A. 2006. Cross-lingual question answering using inter lingual index module of eurowordnet. In *Advances in Natural Language Processing. Research in Computing Science*.

Ferrández, S.; Toral, A.; Ferrández, .; Ferrández, A.; and Muñoz, R. 2007. Applying wikipedia's multilingual knowledge to cross-lingual question answering. In *Proccedings of the 12th International Conference on Applications of Natural Language to Information Systems*.

Joachims, T. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods - Support Vector Learning*.

Levenshtein, V. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*.

Provost, F. 2000. Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI Workshop on Imbalanced Data Sets*.

Schönhofen, P.; Benczúr, A.; Bíró, I.; and Csalogány, K. 2007. Performing cross-language retrieval with wikipedia. In *Working Notes of the Cross Language Retrieval Forum (CLEF)*.

Wentland, W.; Knopp, J.; Silberer, C.; and Hartung, M. 2008. Building a multilingual corpus for named entity disambiguation, translation and transliteration. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. To appear.

Augmenting Wikipedia-Extraction with Results from the Web

Fei Wu and **Raphael Hoffmann** and **Daniel S. Weld**
CSE Department
University of Washington
Seattle, USA
{wufei,raphaelh,weld}@cs.washington.edu

Abstract

Not only is Wikipedia a comprehensive source of quality information, it has several kinds of internal structure (e.g., relational summaries known as *infoboxes*), which enable self-supervised information extraction. While previous efforts at extraction from Wikipedia achieve high precision and recall on well-populated classes of articles, they fail in a larger number of cases, largely because incomplete articles and infrequent use of infoboxes lead to insufficient training data. This paper explains and evaluates a method for improving recall by extracting from the broader Web. There are two key advances necessary to make Web supplementation effective: 1) a method to filter promising sentences from Web pages, and 2) a novel *retraining* technique to broaden extractor recall. Experiments show that, used in concert with shrinkage, our techniques increase recall by a factor of up to 8 while maintaining or increasing precision.

Introduction

Like many others at the workshop, we wish to convert as many facts in Wikipedia as possible into semantic form. Such a system could be useful for next-generation search, question answering and much more. Performing this process autonomously is crucial, since the scale of available knowledge is vast. In many ways our vision is shared with those working on general-purpose information extraction, such as Snowball (Agichtein & Gravano 2000), KnowItAll (Etzioni *et al.* 2005) and Textrunner (Banko *et al.* 2007), but in contrast to systems which seek to extract from arbitrary Web text, we focus on Wikipedia and hope to expand from that base.

The Long-Tailed Challenge: While focusing on Wikipedia helps solve the problem of inaccurate and unreliable source data (Giles 2005), it introduces new challenges. For example, many previous systems (e.g., Mulder (Kwok, Etzioni, & Weld 2001), AskMSR (Brill, Dumais, & Banko 2002), and KnowItAll) exploit the presence of redundant information on the Web, enabling powerful statistical techniques. The Wikipedia corpus, however, has greatly reduced duplication. Fortunately, Wikipedia has several attributes that significantly facilitate extraction: 1) Infoboxes, tabular summaries of an object's key attributes, may be used as a source of training data, allowing for self-supervised learning. 2) Wikipedia gives important concepts their own unique identifier — the URI of a definitional page. The first reference to such a concept often includes a link which can be used for disambiguation. As a result, homonyms are much less of a problem than in unstructured text. 3) Wikipedia *lists* and *categories* provide valuable features for classifying pages.

This paper reports on K2, which extends Wu and Weld's available Kylin system — a self-supervised Wikipedia information extractor (Wu & Weld 2007). Like Kylin, K2 looks for sets of pages with similar infoboxes, determines common attributes for each class, creates training examples, learns extractors, and runs them on each page — creating new infoboxes and completing others. Kylin, itself, works quite well for *popular* infobox classes where users have previously created enough infoboxes to train an effective extractor model. For example, in the "U.S. County" class Kylin has 97.3% precision with 95.9% recall. Unfortunately, most classes contain only a *small number* of infobox-containing articles. Specifically, 1442 of 1756 (82%) classes have fewer than 100 articles, and 42% have 10 or fewer. For classes sitting on this long tail, Kylin can't get enough training data and its extraction performance may be unsatisfactory.

Furthermore, even when Kylin does learn an effective extractor there are numerous cases where a Wikipedia article simply doesn't have much information to be extracted. Indeed, another long-tailed distribution governs the *length* of articles; among the 1.8 million pages,[1] many are short articles and almost 800,000 (44.2%) are marked as *stub* pages, indicating that much-needed information is missing.

Contributions: Thus, in order to create a comprehensive semantic knowledge base summarizing Wikipedia topics, we must confront the problems of these long-tailed distributions. This paper presents K2, which extends Kylin with novel techniques for increasing recall.

- By mapping the contents of known Wikipedia infobox data to TextRunner, a state-of-the-art open information extraction system (Banko *et al.* 2007), K2 creates a larger and cleaner training dataset for learning more robust extractors.
- When it is unable to extract necessary information from a Wikipedia page, K2 retrieves relevant sentences from

[1]Unless noted otherwise, all statistics are taken from the 07/16/2007 snapshot of Wikipedia's English language version.

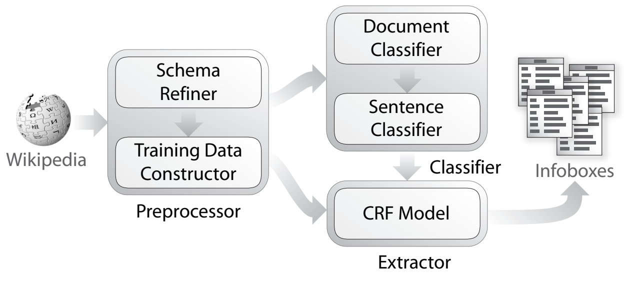


Figure 1: Kylin performs self-supervised information extraction, using Wikipedia inforboxes for training data.

the greater Web. The key to this method is a process for tightly filtering which non-Wikipedia sentences are given to the K2 extractors.

Our techniques work best in concert. Together with shrinkage, they improve the area under the P/R curve by as much as 8, compared with baseline Kylin.

Background: Extraction in Kylin

Following (Wu & Weld 2007) we are interested in the problem of infobox completion. An infobox is a relational summary of an article: a set of attribute / value pairs describing the article's subject (see (Wu & Weld 2007) for an example). Not every article has an infobox and some infoboxes are only partially instantiated with values. We seek to create or complete infoboxes whenever possible. Before explaining how K2 extracts data from the general Web to supplement that found in Wikipedia, we review the basic Kylin architecture (Figure 1), upon which we build.

Preprocessor: The preprocessor selects and refines infobox schemata, choosing relevant attributes; it then generates machine-learning datasets for training sentence classifiers and extractors. Refinement is necessary for several reasons. For example, *schema drift* occurs when authors create an infobox by copying one from a similar article and changing attribute values. If a new attribute is needed, they just make up a name, leading to schema and attribute duplication.

Next, the preprocessor constructs two types of training datasets — those for sentence classifiers, and CRF attribute extractors. For each article with an infobox mentioning one or more target attributes, Kylin tries to find a unique sentence in the article that mentions that attribute's value. The resulting labelled sentences form positive training examples for each attribute; other sentences form negative training examples. If the attribute value is mentioned in several sentences, then one is selected heuristically.

Generating Classifiers: Kylin learns two types of classifiers. For each class of article being processed, a heuristic *document classifier* is used to recognize members of the infobox class. For each target attribute within a class a *sentence classifier* is trained in order to predict whether a given sentence is likely to contain the attribute's value. For this, Kylin uses a maximum entropy model (Nigam, Lafferty, & McCallum 1999) with bagging. Features include a bag of words, augmented with part of speech tags.

Learning Extractors: Extracting attribute values from a sentence is best viewed as a sequential data-labelling problem. Kylin uses conditional random fields (CRFs) (Lafferty, McCallum, & Pereira 2001) with a wide variety of features (e.g., POS tags, position in the sentence, capitalization, presence of digits or special characters, relation to anchor text, etc.). Instead of training a single master extractor to clip all attributes, Kylin trains a different CRF extractor for each attribute, ensuring simplicity and fast retraining.

Shrinkage: Although Kylin performs well when it can find enough training data, it flounders on sparsely populated infobox classes — the majority of cases. We partially mitigated this problem by refining Wikipedia's infobox ontology (Wu & Weld 2008) and improving Kylin's performance using shrinkage, a general statistical technique for improving estimators in the case of limited training data (McCallum *et al.* 1998). K2 uses shrinkage when training an extractor of a instance-sparse infobox class by aggregating data from its parent and children classes.

Shrinkage improves Kylin's precision, but more importantly, it increases recall, when extracting from the long tail of sparse infobox classes. Because this technique leads to extractors with improved robustness, we use shrinkage in K2, when extracting information from the general Web.

Retraining

We now consider how to improve extractor robustness by harvesting additional training data from the *outside* Web. Leveraging information outside Wikipedia to help training extractors, could improve Kylin's recall. To see why, we note that the wording of texts from the greater Web are more diverse than the relatively strict expressions used in many places in Wikipedia.[2] Training on a wider variety of sentences would improve the robustness of Kylin's extractors, which would potentially improve the recall.

The trick here is determining how to automatically identify relevant sentences given the sea of Web data. For this purpose, K2 utilizes TextRunner, an open information extraction system (Banko *et al.* 2007), which extracts semantic relations $\{r|r = \langle \textit{obj}_1\textit{, predicate, obj}_2\rangle\}$ from a crawl of about 10 million Web pages. Importantly for our purposes, TextRunner's crawl includes the top ten pages returned by Google when queried on the title of every Wikipedia article. In the next subsection, we explain the details of our retraining process; then we follow with an experimental evaluation.

Using TextRunner for Retraining: Recall that each Wikipedia infobox implicitly defines a set of semantic triples $\{t|t = \langle \textit{subject, attribute, value}\rangle\}$ where the subject corresponds to the entity which is the article's title. These triples have the same underlying schema as the semantic relations extracted by TextRunner and this allows us to generate new training data.

The retrainer iterates through each infobox class C and again through each attribute, $C.a$, of that class collecting a set of triples from existing Wikipedia infoboxes: $T =$

[2]It is possible that Wikipedia's inbred style stems from a pattern where one article is copied and modified to form another. A general desire for stylistic consistency is another explanation.

$\{t|t.attribute = C.a\}$.[3] The retrainer next iterates through T, issuing TextRunner queries to get a set of potential matches $R(C.a) = \{r|\exists t : r.obj_1 = t.subject, r.obj_2 = t.value\}$, together with the corresponding sentences which were used by TextRunner for extraction. The K2 retrainer uses this mapped set $R_{C.a}$ to augment and clean the training data set for C's extractors in two ways: by providing additional positive examples for the learner, and by eliminating false negative examples which were mistakenly generated by Kylin from the Wikipedia data.

Adding Positive Examples: Unfortunately, TextRunner's raw mappings, $R(C.a)$, are too noisy to be used as positive training examples. There are two causes for the noise. The most obvious cause is the imperfect precision of TextRunner's extractor. But false positive examples can also be generated when there are multiple interpretations for a query. Consider the TextRunner query $\langle r.obj_1 = A, r.predicate =?, r.obj_2 = B\rangle$, where A is a person and B is his birthplace. Since many people die in the same place that they were born, TextRunner may well return the sentence "Bob died in Seattle." which would be a poor training example for birthplace.

Since false positives could greatly impair training, the K2 retrainer morphologically clusters the predicates which are returned by TextRunner (e.g., "is married to" and "was married to" are grouped). We discard any predicate that is returned in response to a query about more than one infobox attribute. Only the k most common remaining predicates are then used for positive training examples; in our experiments we set $k = 1$ to ensure high precision.

Filtering Negative Examples: As explained in (Wu & Weld 2007), Kylin considers a sentence to be a negative example unless it is known to be positive or the *sentence classifier* labels it as potentially positive. This approach eliminates many false negatives, but some remain. A natural idea is to remove a sentence from the set of negative examples if it contains the word denoting the relation itself. Unfortunately, this technique is ineffective if based soley on Wikipedia content. To see why, consider the "Person.spouse" attribute which denotes the "marriage" relation —because the word "spouse" seldom appears in natural sentences, few false negatives are excluded. But by using TextRunner, we can better identify the phrases (predicates) which are harbingers of the relation in question. The most common are used to eliminate negative examples. By adding new positive examples and excluding sentences which might be false negatives, retraining generates an improved training set, whose benefit we now quantify.

Retraining Experiments: We pose two questions: 1) Does K2's retraining improve over Kylin's extractors? 2) Do the benefits from retraining combine synergistically with those from shrinkage? Before addressing those questions we experimented with different retraining alternatives (e.g., just adding positive examples and just filtering negatives). While both approaches improved extractor performance, the combination worked best, so the combined method was used in the subsequent study.

We evaluate retraining in two different cases. In the first case, we use nothing but the target class' infobox data to prime TextRunner for training data. In the second case, we first used uniform-weight shrinkage to create a training set which was then used to query TextRunner. To compute precision and recall, we manually verified the attribute values contained in the articles. This made it necessary to limit the evaluation set and so we tested on four randomly picked classes of different sizes. We got improved results on each; Figure 2 shows the results on the sparsest and on the most popular class.

We note that in most cases retraining improves the performance, in both precision and recall. When compared with shrinkage, retraining provides less benefit for sparse classes but helps more on the popular class "writer." This makes sense because without many tuples to use for querying TextRunner, retraining has little effect. We suspect that full cotraining would be more effective on sparse classes when shrinkage was unavailable. Finally, we observe that the combination of shrinkage and retraining is synergistic, always leading to the biggest improvement. Particularly, on the two sparsest classes "Irish Newspaper" and "Performer", it substantially improved recall by 590% and 73.3% respectively, with remarkable improvement in precision as well; and the areas under the precision and recall curve improve 1753% and 66% respectively. For the more popular classes "Baseball Stadium" and "Writer" recall improved by 41% and 11% respectively.

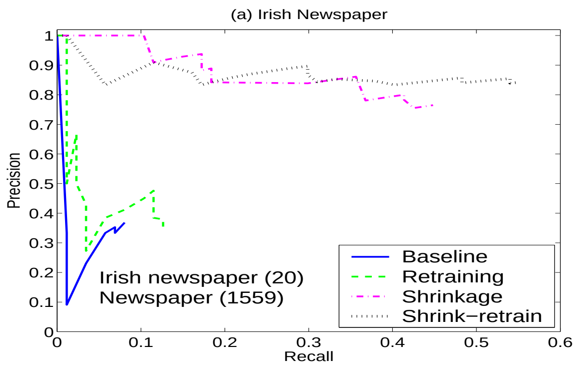
Extracting from the Web

While shrinkage and retraining improve the quality of Kylin's extractors, the lack of redundancy of Wikipedia's content makes it increasingly difficult to extract additional information. Facts that are stated using uncommon or ambiguous sentence structures often hide from the extractors.

In order to retrieve facts which can't be extracted from Wikipedia, we extract from the general Web: We train extractors on Wikipedia articles and then apply them to relevant Web pages. The challenge — as one might expect — is maintaining high precision. Since the extractors have been trained on a very selective corpus, they are unlikely to discriminate irrelevant information. For example, an extractor for a person's birthdate will have been trained on a set of pages all of which have that person's life as their primary subject. Such extractors become inaccurate when applied to a page which compares the lives of several people — even if the person in question is one of those mentioned.

To ensure extraction quality, it is crucial to carefully select which content is to be processed by K2's extractors. We viewed this task as an information retrieval problem, and solved it in the following steps: K2 generates a set of queries and utilizes a general Web search engine, namely Google, to identify a set of pages which are likely to contain the desired information. The top-k pages are then downloaded, and the

[3] We note that another way of generating the set, T, would be to collect baseline Kylin extractions for $C.a$ instead of using existing infoboxes. This would lead to a *cotraining* approach rather than simple retraining. One could iterate the process of getting more training date from TextRunner with improvements to the Kylin extractor (Blum & Mitchell 1998).



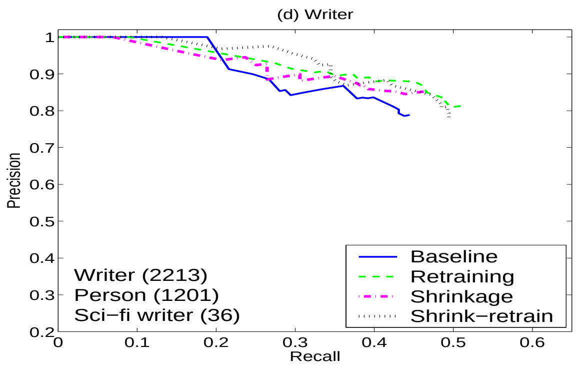


Figure 2: Used in isolation, retraining enables a modest but marked improvement in recall. And combining retraining with shrinkage yields substantially improved extractors with dramatic improvements to precision in the sparse Irish Newspaper domain (only 20 infoboxes) and improved recall in both domains. Note that Irish Newspaper used shrinkage from the paper Newspaper class (1559 infoboxes), while Writer used shrinkage from both a parent and a child class.

text on each page is split into sentences, which K2 processes in turn. Finally, each extraction is weighted using a combination of factors which we will explain shortly.

Choosing Search Engine Queries: The first step is ensuring that the search engine returns highly relevant pages. A simple approach is to use the article title as a query. Suppose we are interested in finding the birth date of Andrew Murray, a writer, whose Wikipedia page is titled "Andrew Murray (minister)". Wikipedia uses information in parentheses to resolve ambiguities, but K2 removes it to increase recall. To improve result relevance, quotes are placed around the remaining string, here `''andrew murray''`.

Although such a query might retrieve many pages about Murray, it is possible that none of the top k contains the attribute value in question. K2 therefore runs several more restrictive queries which contain additional keywords to better target the search.

One such query is the quoted article title followed by the attribute name, as in `''andrew murray'' birth date`. While this increases the chance that a returned page contains the desired information, it also greatly reduces recall, because the terms 'birth date' might not actually appear on a relevant page. For example, consider the sentence "Andrew Murray was born in 1828.".

But note that K2 has already computed a list of predicates which are indicative of each attribute (e.g. 'was born in' for the birth date), as explained in our section on retraining. Thus, K2 generates appropriate queries for each predicate, which combines the quoted title with these predicates. The combined results of all queries (title only, title and attribute name, as well as title and any attribute predicate) are retrieved for further processing.

Weighing Extractions: Pages which do not contain the preprocessed article title, here 'Andrew Murray', are discarded. Then, formatting commands and scripts are removed, and sentences in the remaining text are identified.

Since most sentences are still irrelevant, running Kylin's extractors on these directly would result in many false positives. Recall that unlike Wikipedia's articles, web pages often compare multiple related concepts, and so we would

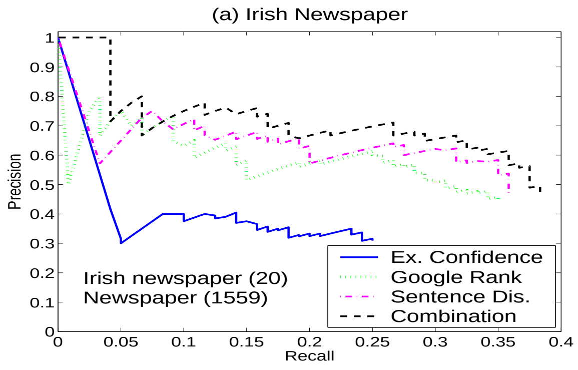


Figure 3: When applying K2 to Web pages, the CRF extractor's confidence is a poor choice for scoring competing extractions of the same attribute. By factoring in IR features, performance improves substantially.

like to capture the likeliness that a sentence or extraction is relevant to a concept. A variety of features may be indicative of content relevance, but K2 uses two in particular:

- The number of sentences δ_s between the current sentence and the closest sentence containing the (preprocessed) title of the article.
- The rank of the page δ_r on Google's results lists returned in response to our queries.

Each retrieved sentence is processed by Kylin's extractors, and for each extraction a combined score is computed. This score takes into account both factors δ_s and δ_r as well as the confidence δ_c reported by Kylin's extractors. The combined score is obtained in the following way: First, each of the three parameters $\delta_s, \delta_r, \delta_c$ is normalized by applying a linear mapping into the intervals $[\alpha_s, 1]$, $[\alpha_r, 1]$, and $[\alpha_c, 1]$ respectively, where 1 corresponds to the optimal value and α_s, α_r, and α_c are user-defined parameters. With δ_s^*, δ_r^*, and δ_c^* denoting the normalized weights, the combined score is then obtained as $score_{web} := \delta_s^* * \delta_r^* * \delta_c^*$.

Combining Wikipedia and Web Extractions: Finally, K2 combines extraction results from Wikipedia and Web pages. Extractions from Wikipedia are ranked by extractor confidence $score_{wiki} := \delta_c$ and extractions from the Web by

$score_{web}$ as defined above. But what is the overall best extraction? We expect that extractions from Wikipedia tend to be more precise (a given Wikipedia article is known to be relevant, of high quality, and of consistent structure for which Kylin's extractors have been trained), but fewer.

K2 applies a simple normalization and always returns the extraction with highest score. To be able to balance the weight of one extractor versus the other, K2 adjusts the score of extractions from the Web to $1 - (1 - score_{web})^{\lambda}$, where λ is a new parameter. If $\lambda = 0$, extractions from the Web are not considered, and if $\lambda = 1$, $score_{wiki}$ and $score_{web}$ are compared directly.

Web Experiments: In our experiments we investigated 1) how to best weigh extractions from the Web, and 2) if our techniques for combining extractions from Wikipedia and Web pages improve recall while maintaining precision.

We assume that there exists some correct value for each attribute contained in the infobox template for an article and define recall to be the proportion of correct attribute values relative to all attributes. Note that most infoboxes in Wikipedia do not provide a value for each attribute contained in the corresponding infobox template. For example, the attribute "spouse" does not make sense for people who are not married, and "death date" for people who are still alive. Therefore, our recall estimates are conservative, but enable a relative comparison of our proposed techniques.

For all experiments, we queried Google for the top-100 pages containing the article title, and the top-10 pages containing the article title and the attribute name or any associated predicate. Each new extraction — for which no ground truth existed in Wikipedia — was manually verified for correctness by visiting the source page.

In our first series of experiments (Figure 3), we used Shrink-Retrain — the best extractors trained on Wikipedia — and applied different weighting functions to select the best extraction for an attribute. The CRF extractor's reported confidence performed poorly in isolation. Giving priority to extractions from pages at a higher position in Google's returned result lists and resolving ties by confidence, yielded a substantial improvement. Similarly, we tried giving priority to extractions which were fewer sentences apart from the occurrence of the Wikipedia article title on a page, again resolving ties by extractor confidence. The large improvements in precision and recall (as highlighted in Figure 3) show that much of the returned text is irrelevant, but can be re-weighted using simple heuristics. Finally, we were interested if a weighted combination of these factors would lead to synergies. We set $\alpha_s = .1$, $\alpha_r = .7$, $\alpha_c = .9$, so that each factor was roughly weighted by our observed improvement (results were not sensitive to minor variations). On all datasets, performance was comparable or better than the best factor taken in isolation.

In our second series of experiments (Figure 4), we combined extractions from Wikipedia and the Web. In both cases, we applied the Shrink-Retrain extractor, but scored extractions from the Web using the weighted factor combination with $\lambda = .4$. The results, shown in Figure 4, show large improvements in recall at higher precision for the popular "Writer" (42%) dataset, and at moderately reduced precision for the sparse "Irish Newspaper" dataset. The area under the curve was substantially expanded in both cases, by 58% and 15% respectively. Compared to the original baseline system, the area has expanded 93% and 771% respectively. On the "Baseball Stadium" and "Performer" classes, the area has expanded 91% and 102% respectively.

In future work, we would like to automatically optimize the parameters α_s, α_r, α_c, λ based on comparing the extractions with values in the infobox.

Related Work

In the preceding sections we have discussed how our work relates to co-training. In this section, we discuss the broader context of previous work on unsupervised information extraction and other Wikipedia-based systems.

Unsupervised Information Extraction: Since the Web is large and highly heterogeneous, unsupervised and self-super-vised learning is necessary for scaling. Several systems of this form have been proposed. SNOWBALL (Agichtein & Gravano 2000) iteratively generates extraction patterns based on occurrences of known tuples in documents to extract new tuples from plain texts. MULDER (Kwok, Etzioni, & Weld 2001) and AskMSR (Brill, Dumais, & Banko 2002) use the Web to answer questions, exploiting the fact that most important facts are stated multiple times in different ways, which licenses the use of simple syntactic processing. Instead of utilizing redundancy, K2 exploits Wikipedia's unique structure and the presence of user-tagged data to train machine learners. Patwardhan et al. proposed a decoupled information extraction system by first creating a self-trained relevant sentence classifier to identify relevant regions, and using a semantic affinity measure to automatically learn domain-relevant extraction patterns (Patwardhan & Riloff 2007). K2 uses the similar idea of decoupling when applying extractors to the general Web. However, K2 uses IR-based techniques to select relevant sentences and trains CRF extractors.

Other Wikipedia-Based Systems: Bunescu and Pasca utilized Wikipedia to detect and disambiguate named entities in open domain documents (Bunescu & Pasca 2006). Ponzetto et al. derived a large scale taxonomy based on the Wikipedia category system by identifying the IS-A relationships among category tags (Ponzetto & Strube 2007). Auer and Lehmann developed the DBpedia (Auer & Lehmann 2007) system which extracts information from existing infoboxes within articles and encapsulate them in a semantic form for query. In contrast, K2 populates infoboxes with *new* attribute values. Suchanek et al. implement the YAGO system (Suchanek, Kasneci, & Weikum 2007) which extends WordNet using facts extracted from Wikipedia's category tags. But in contrast to K2, which can learn to extract values for *any* attribute, YAGO only extracts values for a limited number of predefined relations.

Conclusion

Wu and Weld's Kylin system demonstrated the ability to perform self-supervised information extraction from Wikipedia (Wu & Weld 2007). While Kylin achieved high precision and reasonable recall on popular infobox classes, most

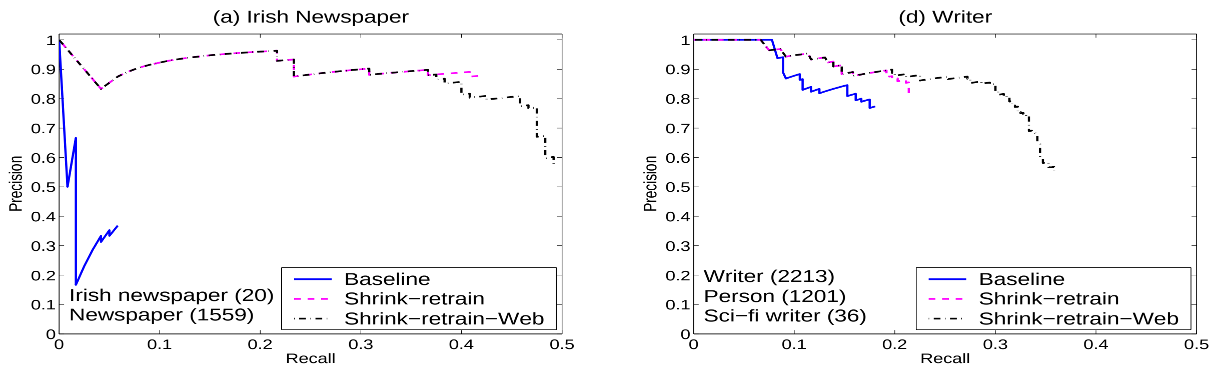


Figure 4: Combining Kylin's extractions from Wikipedia and the Web yields a substantial improvement in recall without compromising precision. Already, shrink-retrain improved recall over the original Kylin system, here the baseline, but the combination of extractions from Wikipedia and the Web, shrink-retrain-Web, performs even better. Note that recall is substantially improved, even for the Writer class, which has many infoboxes (2213) for training.

classes (i.e., 82%) provide fewer than 100 training examples; on these classes, Kylin's performance is unacceptable.

This paper describes the K2 system, which extends Kylin by supplementing Wikipedia extractions with those from the Web. There are two keys to effective (self-supervised) Web extraction: 1) careful filtering to ensure that only the best sentences are considered for extraction and 2) a novel retraining technique which generates more robust extractors. While these techniques are useful individually, their combination is synergistic (Figure 4):

- Precision is modestly improved in most classes, with larger gains if sparsity is extreme ("Irish Newspaper").
- Recall sees extraordinary improvement with gains from 0.06% to 0.49% (a factor of 8.4) in extremely sparse classes such as "Irish Newspaper." Even though the "Writer" class is populated with over 2000 infoboxes, its recall improves from 18% to 32% (a factor of 1.77) at equivalent levels of precision.
- Calculating the area under the precision / recall curve also demonstrates substantial improvement, with an improvement factor of 16.71, 2.02, 1.91, and 1.93 for "Irish Newspaper," "Performer," "Baseball Stadium," and "Writer," respectively.

Much remains to be done. For example, we wish to extend our retraining technique to full cotraining. There are ways to better integrate extraction of Web content with that of Wikipedia, ranging from improved querying policies to DIRT-style analysis of extraction patterns (Lin & Pantel 2001).

References

Agichtein, E., and Gravano, L. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*.

Auer, S., and Lehmann, J. 2007. What have Innsbruck and Leipzig in common? Extracting semantics from wiki content. In *ESWC07*.

Banko, M.; Cafarella, M. J.; Soderland, S.; Broadhead, M.; and Etzioni, O. 2007. Open information extraction from the Web. In *Proceedings of IJCAI07*.

Blum, A., and Mitchell, T. 1998. Combining Labeled and Unlabeled Data with Co-Training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, 92–100.

Brill, E.; Dumais, S.; and Banko, M. 2002. An analysis of the AskMSR question-answering system. In *Proceedings of EMNLP02*.

Bunescu, R., and Pasca, M. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL06*.

Etzioni, O.; Cafarella, M.; Downey, D.; Kok, S.; Popescu, A.; Shaked, T.; Soderland, S.; Weld, D.; and Yates, A. 2005. Unsupervised named-entity extraction from the Web: An experimental study. *Artificial Intelligence* 165(1):91–134.

Giles, J. 2005. Internet encyclopaedias go head to head. *Nature* 438:900–901.

Kwok, C. T.; Etzioni, O.; and Weld, D. 2001. Scaling question answering to the Web. *ACM Transactions on Information Systems (TOIS)* 19(3):242–262.

Lafferty, J.; McCallum, A.; and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of WWW01*.

Lin, D., and Pantel, P. 2001. DIRT @SBT@discovery of inference rules from text. In *Knowledge Discovery and Data Mining*, 323–328.

McCallum, A. K.; Rosenfeld, R.; Mitchell, T. M.; and Ng, A. Y. 1998. Improving text classification by shrinkage in a hierarchy of classes. In Shavlik, J. W., ed., *Proceedings of ICML-98, 15th International Conference on Machine Learning*, 359–367. Madison, US: Morgan Kaufmann Publishers, San Francisco, US.

Nigam, K.; Lafferty, J.; and McCallum, A. 1999. Using maximum entropy for text classification. In *Proceedings of Workshop on Machine Learning for Information Filtering, IJCAI99*.

Patwardhan, S., and Riloff, E. 2007. Effective information extraction with semantic affinity patterns and relevant regions. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-07*.

Ponzetto, S. P., and Strube, M. 2007. Deriving a large scale taxonomy from wikipedia. In *Proceedings of AAAI07*.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: A core of semantic knowledge - unifying WordNet and Wikipedia. In *Proceedings of WWW07*.

Wu, F., and Weld, D. 2007. Autonomouslly semantifying wikipedia. In *Proceedings of CIKM07*.

Wu, F., and Weld, D. 2008. Automatically refining the wikipedia infobox ontology. In *Proceedings of WWW08*.

Using Wikipedia Links to Construct Word Segmentation Corpora

David Gabay and **Ziv Ben-Eliahu** and **Michael Elhadad**
Ben Gurion University of the Negev
Department of Computer Science
POB 653 Be'er Sheva, 84105, Israel
(gabayd|ben-elia|elhadad)@cs.bgu.ac.il

Abstract

Tagged corpora are essential for evaluating and training natural language processing tools. The cost of constructing large enough manually tagged corpora is high, even when the annotation level is shallow. This article describes a simple method to automatically create a partially tagged corpus, using Wikipedia hyperlinks. The resulting corpus contains information about the correct segmentation of 523,599 non-consecutive words in 363,090 sentences. We used our method to construct a corpus of Modern Hebrew (which we have made available at http://www.cs.bgu.ac.il/~nlpproj). The method can also be applied to other languages where word segmentation is difficult to determine, such as East and South-East Asian languages.

Word Segmentation in Hebrew

Automatic detection of word boundaries is a non-trivial task in a number of languages. Ambiguities arise in writing systems that do not contain a word-end mark, most notably East-Asian logographic writing systems and South-East Asian alphabets. Ambiguities also appear in alphabets that contain a word-end mark, but sometimes allow agglutination of words or insertion of a word-end mark inside a single word. A discussion of the definition of "word" in general can be found, for example, in (Sciullo and Williams 1987). We focus in this work on word segmentation in unvocalized Modern Hebrew. According to common definitions (see (Adler 2007) Chapter 2 for a recent review), a Hebrew word may consist of the following elements: a proclitic, a stem, an inflectional suffix and a pronominal suffix (enclitic). In the official standard defined by the Mila Knowledge Center for Hebrew,[1], as well as other work in parts of speech (POS) tagging and morphological analysis of Hebrew, inflectional suffixes are referred to as attributes of the stem. The problem of word segmentation in Hebrew concerns, therefore, the identification of the proclitics, stem and enclitics of a word, while POS tagging refers to assigning the correct part of speech to each part. Morphological disambiguation refers, one step further, to the complete analysis of all the morphological attributes of each word part. Proclitics include conjunctions, prepositions, complementizers and the definite article. They are composed of one or two letters and follow a strict order. The segmentation of a given word is often ambiguous. In a corpus of 40 million tokens, we found that there are, on average, 1.26 different possible segmentations per token, even when only proclitics are being considered. For example, the word[2] *$btw* may be segmented, among other options, as:
$-b-tw, meaning "that in a note"
$-bt-w, meaning "that his daughter"
$btw, meaning "(they) went on strike"
The major cause for ambiguity is proclitics, as enclitics are rare in Modern Hebrew. When performing POS-tagging or full morphological analysis, word segmentation can be performed as a separate first step (Bar-Haim, Sima'an, and Winter 2005), alongside POS-tagging (Adler and Elhadad 2006) or even in joint inference with syntactic analysis (Cohen and Smith 2007), following (Tsarfati 2006). Word segmentation may also be considered a separate task, easier than full morphological analysis but still far from trivial. As a separate task, it has practical value on its own - narrowing search results, for example. Current work in POS tagging and morphological analysis reports success rate of 97.05 percent in word segmentation for supervised learning (Bar-Haim, Sima'an, and Winter 2008). In the case of unsupervised learning, 92.32 percent accuracy is reported by (Adler and Elhadad 2006) in segmentation and simple POS tagging, without full morphological analysis. The lack of annotated corpora is one of the problems in assessing NLP tools for modern Hebrew. In this work, we propose an original method that exploits Wikipedia data to obtain high-quality word segmentation data.

Wikipedia Links and Word Segmentation

Using Wikipedia as a data source for NLP and AI tasks has become common in recent years, as work in different fields makes use of the attractive Wikipedia qualities: it is easily accessible, large and constantly growing, multilingual, highly structured, and deals with a considerable number of topics. In this work, we focus on the form of hyperlinks in Wikipedia. Wikipedia hyperlinks, together with a man-

[1]http://www.mila.cs.technion.ac.il

[2]For the sake of simplicity, we use only transliterations in this article. Translitatetion follows ISO standard.

ual mapping of article names into WordNet labels, have already been used by (Mihalcea 2007) to generate sense-tagged corpora. We follow a similar intuition to address word segmentation. We make use of the structure of hyperlinks within Wikipedia, that is, hyperlinks between different articles within Wikipedia. Internal Wikipedia hyperlinks consist of a surface form, visible to the reader, and the name of the Wikipedia article to which the hyperlink leads, which is a unique identifier. The syntax of internal hyperlinks in Wikitext - the markup language in which Wikipedia is written - is as follows: hyperlinks are surrounded by double brackets. They may contain a pipe (|) sign, in which case the text preceding the pipe is the name of the linked article, and the part following it is the text that is visible to the reader.

If no pipe appears, then the name of the linked article will be the visible text. For example, the Wikitext sentence (taken from the English Wikipedia): *"During the [[Great Depression in the United States|Great Depression]] of the 1930s, Roosevelt created the [[New Deal]]"* will be parsed to generate the sentence *"During the Great Depression of the 1930s, Roosevelt created the New Deal"*. The first hyperlink will lead to the article *"Great Depression in the United States"* and the second to the article *"New Deal"*. Text that is adjacent immediately before or after the double brackets will be adjacent to the visible text of the hyperlink.

Constructing a Word Segmentation Corpus from the Hebrew Wikipedia

The format of internal hyperlinks in Wikipedia makes it a source of information on word segmentation. We describe how we exploit the form of Wikipedia hyperlinks to construct a corpus in which some of the words are (fully or partially) segmented. For our task, the tags in the corpus will denote the existence or absence of proclitics.[3] We identified five types of internal hyperlinks and text combinations relevant to word segmentation: 1. [[A]], 2. p[[A]], 3. [[A|pA]], 4. p[[B|A]], 5. [[A|B]]; where p is a sequence of one or more proclitics, and A and B are different text segments, consisting of one or more words (note that types 2 and 3 are equivalent in terms of Wiki syntax). Hyperlinks of the first three forms provide reliable data on the correct segmentation of the first word in the text A (provided that A, which is an article name, does not begin with a proclitic, an issue discussed in the next subsection). For example, the hyperlink *[[lwndwn]] (London)* of type 1 indicates that the token *lwndwn* does not contain proclitics, and the hyperlinks *l[[lwndwn]]* of type 2 and *[[lwndwn|llwndwn]]* of type 3 both indicate that the word *llwndwn* should be segmented into *l+lwndwn (to-London)*. Hyperlinks of types 4 and 5 may also contain information on word segmentation, but this information is not consistent, since prepositional letters may appear both in and out of the hyperlink. In the first step of the construction of our word segmentation corpus, we removed all Wikitext syntax from the article code, except for internal hyperlinks of types 2 and 3, and hyperlinks of the first type in which the first word in the brackets is not ambiguous in terms of word segmentation. In the second step, the double brackets format was replaced by a unified XML setting. The result was a corpus of Hebrew text in which some of the tokens are tagged for word segmentation. Since automatic taggers for Hebrew work at the sentence level and do not make use of higher context, we also constructed a more succinct corpus containing only complete sentences (and not list items, or other isolated text units) in which one or more tokens contain segmentation data. The resulting corpus includes 363,090 sentences with 523,599 tagged words (out of 8.6 million tokens altogether), taken from 54,539 Wikipedia articles. Each tagged word in the resulting corpus is potentially ambiguous, meaning that its beginning matches some sequence of proclitics. (other than *h*, for reasons specified in the next subsection).

Accuracy

The method described in this section relies on two assumptions: that the vast majority of article names in Wikipedia do not start with a proclitic and that almost all hyperlinks in Wikipedia are written correctly, according to Wikitext syntax. The first assumption does not hold for one proclitic - *h*, the definite article. A non-negligible amount of articles do start with this proclitic (*e.g., hmhpkh hcrptit, The-revolution the-French, the French revolution*). Due to this fact, and to the fact that *h*, unlike any other proclitic, may be covert, we decided not to include any data regarding the proclitic *h* in the corpus. With the *h* excluded, we manually checked a random sample of 600 tags generated by our corpus and found all of them to be correct. Our assumption on article names seems to hold in this case since article names are given according to a naming policy, and are usually names of people, places, organizations, objects or concepts, that do not start with a proclitic, except for the definitive article case. However, a small fraction of article names do start with other proclitics. This group includes titles of works of art (such as movies and books) and expressions. There are several ways to address this issue. The first is to ignore it, as it corresponds to a very small number of tags: a randomly picked sample of 600 article names with a possible segmentation ambiguity (note that this check is not the same as picking hyperlinks at random) contained just one title that begins with a proclitic - *lgnawlwgih $l hmwsr (to-Genealogy of the-Morality, On the Genealogy of Morality)*, the name of a book. This low amount of noise can be considered bearable. A second option of dealing with the article names problem is to use the fact that Wikipdeia articles are categorized, and that almost all 'bad' article names are concentrated in few categories. We could omit the tags in the case of possibly ambiguous article names from categories such as movies, books or expressions. A third option is to review the list of possibly ambiguous article names, and remove from the corpus all tags that are based on hyperlinks to articles whose names start with a prepositional letter. The first option will result in some noise in the corpus, the second will not allow full usage of the data, and the third requires some manual labor, although considerably less than the effort needed to tag the corpus manually. (Note that most articles have several

[3] We did not deal with enclitics, as they are quite rare in Modern Hebrew.

hyperlinks leading to them).

Our second assumption - that hyperlinks are almost always correct - seems to hold since great effort is made by Wikipedia editors to remove syntax errors. Still, it may be wise to use the articles' versions history to filter out new articles, or articles that were only edited a few times, or by too few people, before generating a tagged corpus out of a snapshot of Wikipedia.

Using the Word Segmentation Corpus

An immediate usage of the "Wikipedia segmentation corpus" generated by our method is to evaluate Hebrew parts-of-speech taggers. Large-scale tagged corpora are needed to properly evaluate the quality of any NLP tool, and are of particular importance at the current stage of research in Hebrew POS-tagging. In the last decade, significant progress has been made in this field. Further improvements in existing systems require fine tuning, for example, redefining the tagset (Netzer et al. 2007) in a way that affects a small percentage of the words. The impact of such changes is difficult to measure using currently available annotated corpora. Since Hebrew is highly inflectional, the number of POS categories and word-forms is high, and a large corpus is needed so that a sufficient number of combinations would be encountered. Constructing large manually tagged corpora is an expensive and laborious task, and lack of tagged corpora remains a bottleneck. We first used the corpus obtained by our method to evaluate the performance of Adler's HMM POS tagger (Adler and Elhadad 2006). The tagger gave the correct segmentation on 74.8 percent of the tagged words. This result suggests that word segmentation in Hebrew is not yet fully resolved, at least in the case of unsupervised taggers. The accuracy rate is significantly lower than that reported in previous results, as can be explained by the high rate of out-of-vocabulary words and by the fact that every token in the Wikipedia segmentation corpus is ambiguous in terms of segmentation, while previous success rates refer to unambiguous words as well. Any POS-tagger or morphological disambiguation tool must also provide word segmentation: it is reasonable to assume that the results on segmentation are a good indicator of overall performance, as failure to tag a word correctly is likely to lead to segmentation errors further on in the sentence. In all reported results, heuristics that improve segmentation also improve tagging accuracy. Thus, a very large segmentation corpus may be used, together with a small fully-annotated corpus, to evaluate overall performance of tools that do more than segmentation. The Wikipedia corpus cannot be used to effectively train supervised segmentors or taggers on its own, since it only contains partial, non-representative data on the segmentation of words in the corpus. It can be used to improve the training done by a manually tagged corpus. It may also be used to tune the initial conditions in unsupervised methods (see (Goldberg, Adler, and Elhadad 2008) for details on the importance of initial conditions). Experiments are currently being performed to evaluate the effectiveness of this approach.

Future work

The Hebrew Wikipedia Word Segmentation corpus can be extended to include enclitics. With some manual labor, it can also be extended to deal with the proclitic *h*. The general approach described in this paper may be applied to any language in which word segmentation is a problem and where a large enough Wikipedia (or other sources written in Wikitext) exists, while large-scale manually annotated corpora do not. Thai, with more than 34,000 articles in its version of Wikipedia (as of March 2008), is an example of such a language.

References

[Adler and Elhadad 2006] Adler, M., and Elhadad, M. 2006. An unsupervised morpheme-based hmm for hebrew morphological disambiguation. In *ACL06*, 665–672. Morristown, NJ: Association for Computational Linguistics.

[Adler 2007] Adler, M. 2007. *Hebrew Morphological Disambiguation: An Unsupervised Stochastic Word-based Approach*. Ph.D. Dissertation, Ben-Gurion University of the Negev, Beer-Sheva, Israel.

[Bar-Haim, Sima'an, and Winter 2005] Bar-Haim, R.; Sima'an, K.; and Winter, Y. 2005. Choosing an optimal architecture for segmentation and pos-tagging of modern Hebrew. In *Proceedings of ACL-05 Workshop on Computational Approaches to Semitic Languages*.

[Bar-Haim, Sima'an, and Winter 2008] Bar-Haim, R.; Sima'an, K.; and Winter, Y. 2008. Part-of-speech tagging of modern hebrew text. *Journal of Natural Language Engineering* 14:223–251.

[Cohen and Smith 2007] Cohen, S. B., and Smith, N. A. 2007. Joint morphological and syntactic disambiguation. In *EMNLP07*, 208–217. Prague, Czech: Association for Computational Linguistics.

[Goldberg, Adler, and Elhadad 2008] Goldberg, Y.; Adler, M.; and Elhadad, M. 2008. Em can find pretty good hmm pos-taggers (when given a good start). In *Proceedings of ACL 2008*.

[Mihalcea 2007] Mihalcea, R. 2007. Using wikipedia for automatic word sense disambiguation. In *Proceedings of NAACL HLT 2007*, 196–203. Rochester, NY: Association for Computational Linguistics.

[Netzer et al. 2007] Netzer, Y.; Adler, M.; Gabay, D.; and Elhadad, M. 2007. Can you tag the modal? you should! In *ACL07 Workshop on Computational Approaches to Semitic Languages*, 57–65. Prague, Czech: Association for Computational Linguistics.

[Sciullo and Williams 1987] Sciullo, A. M. D., and Williams, E. 1987. *On the Definition of Word*. Cambridge, MA: MIT Press.

[Tsarfati 2006] Tsarfati, R. 2006. Integrated morphological and syntactic disambiguation for modern hebrew. In *Proceedings of CoLing/ACL-06 Student Research Workshop*, 49–54. Association for Computational Linguistics.

Method for Building Sentence-Aligned Corpus from Wikipedia

Keiji Yasuda[†, ‡] **and Eiichiro Sumita**[†, ‡]

[†] ATR Spoken Language Translation Research Laboratories [‡] National Institution of Information and Communications Technology

Abstract

We propose the framework of a Machine Translation (MT) bootstrapping method by using multilingual Wikipedia articles. This novel method can simultaneously generate a statistical machine translation (SMT) and a sentence-aligned corpus. In this study, we perform two types of experiments. The aim of the first type of experiments is to verify the sentence alignment performance by comparing the proposed method with a conventional sentence alignment approach. For the first type of experiments, we use JENAAD, which is a sentence-aligned corpus built by the conventional sentence alignment method. The second type of experiments uses actual English and Japanese Wikipedia articles for sentence alignment. The result of the first type of experiments shows that the performance of the proposed method is comparable to that of the conventional sentence alignment method. Additionally, the second type of experiments shows that we can obtain the English translation of 10% of Japanese sentences while maintaining high alignment quality (rank-A ratio of over 0.8).

Introduction

Wikipedia has articles in more than 200 languages, and it is one of the most varied language resources in the world. The current version of Wikipedia expresses multilingual relationships by only interlanguage links. Although Wikipedia has been used as a bilingual language resource in some natural language processing (NLP) researches (Erdmann et al. 2008), an interlanguage link is insufficient information for conducting some NLP researches such as corpus-based machine translation. Therefore, sentence-aligned parallel corpuses are required for these researches.

In this study, we propose a novel MT bootstrapping framework that can simultaneously generate a statistical machine translation (SMT) and a sentence-aligned corpus. SMT assist general users of Wikipedia by translating Wikipedia articles automatically. The sentence-aligned corpus assists NLP researchers by expanding the application of Wikipedia as a multilingual language resource.

Proposed Method

Figure 1 illustrates the framework of MT bootstrapping in the case of using English and Japanese Wikipedia. As shown in this figure, the MT bootstrapping method involves the following steps:

Step 1: Translate Wikipedia articles using an MT system[1].

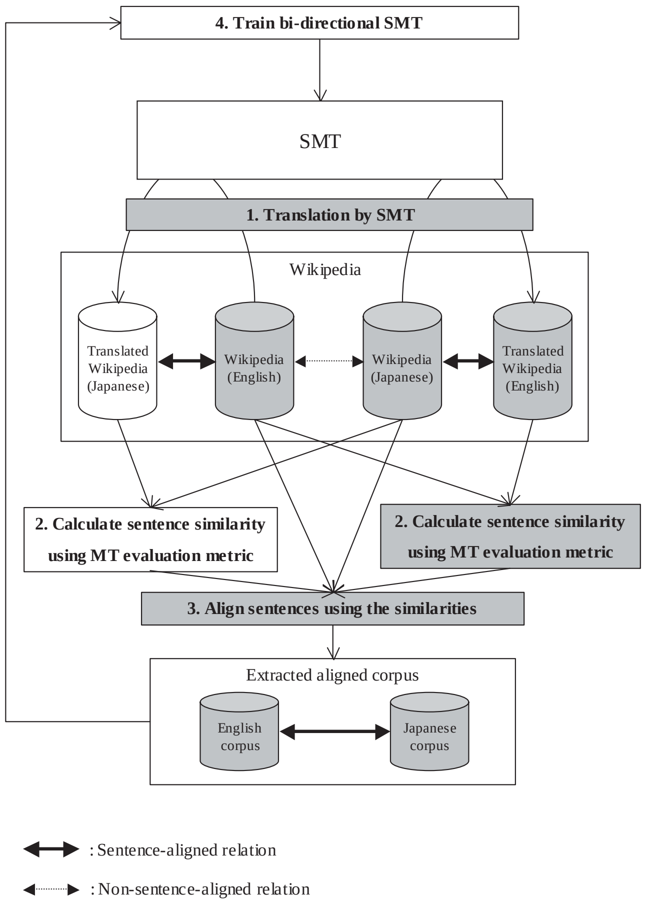

4. Train bi-directional SMT
SMT
1. Translation by SMT
Wikipedia
Translated Wikipedia (Japanese)
Wikipedia (English)
Wikipedia (Japanese)
Translated Wikipedia (English)
2. Calculate sentence similarity using MT evaluation metric
2. Calculate sentence similarity using MT evaluation metric
3. Align sentences using the similarities
Extracted aligned corpus
English corpus
Japanese corpus
: Sentence-aligned relation
: Non-sentence-aligned relation

Fig. 1 MT bootstrapping framework

Step 2: Calculate the sentence-level MT evaluation score between Japanese sentences in the original Japanese Wikipedia and the Japanese sentences obtained by translating English Wikipedia. Similarly, calculate the sentence-level MT evaluation score between the target English sentences in the original English Wikipedia and the English sentences obtained by translating Japanese Wikipedia.

Step 3: Align sentence pairs from the original Japanese Wikipedia and English Wikipedia using either or both of the scores calculated in step 2.

Step 4: Train the SMT using the sentence-aligned corpus.

There are two main problems with this method. One is

[1] For the first loop, com mercial MT systems can be used instead of SMT.

the computational cost of implementing the above steps. Step 3 in particular requires excessive calculations because we have to calculate a sentence-level score for $2 \times n_{source} \times n_{target}$ pairs, where n_{source} is the number of sentences in the source language comparable corpus and n_{target} is the number of sentences in the target language comparable corpus.

One method to reduce the calculation cost is to use an interlanguage link to prune candidate articles. Then, a sentence-level bilingual evaluation understudy (BLEU) score can be calculated only for the candidate articles. This method effectively reduces the computational cost of step 3. However, the iteration of steps 1 to 4 still induces a large computational cost. To deal with these problems, we have raised funds and we use a supercomputer (HPC2500, 2002).

The other problem concerns the alignment confidence measure that is based on the MT evaluation measure. Most proposed MT evaluation measures are used to evaluate translation quality, and no research uses these measures for evaluating sentence alignment confidence. If our proposed framework functions effectively, it will be a promising application because most MT evaluation techniques do not require any language resources such as a bilingual thesaurus or lexicon, which are used in most conventional sentence alignment methods. In the experiments described in the next section, we test the effectiveness of the MT evaluation technique.

Experiments

We perform two types of experiments. The aim of the first type of experiments is to verify the sentence alignment performance by comparing the proposed method with a conventional sentence alignment approach. A JENAAD corpus, which is a sentence-aligned corpus built by the conventional sentence alignment method, is used for the experiments. The second type of experiments uses actual English Wikipedia and Japanese Wikipedia articles for sentence alignment.

Experiments using JENAAD corpus

Experimental settings. To verify the effectiveness of sentence alignment based on the MT evaluation measure, we perform the experiments depicted in Fig. 2 by using a preliminary sentence-aligned corpus. As shown in this figure, the experiments involve the following steps.

1. Train the SMT using the preliminary sentence-aligned corpus.

2. Translate the corpus using the trained translation system.

3. Calculate the sentence-level BLEU score (Papineni et al. 2002) between the preliminarily aligned sentence pairs.

4. Filter out unreliable aligned pairs using the BLEU score.

5. Retrain the STM using the filtered sentence pairs.

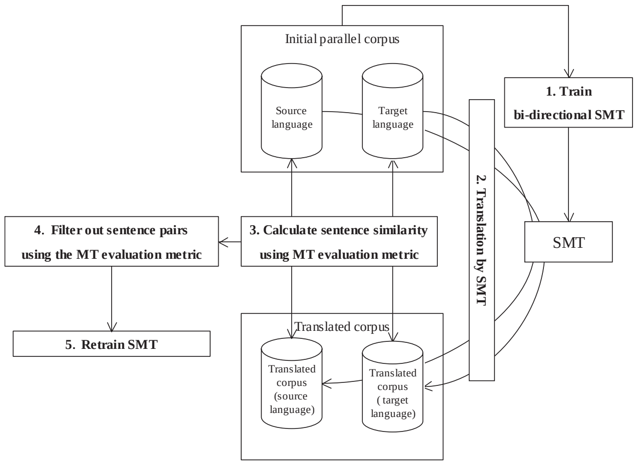

Initial parallel corpus
Source language
Target language
1. Train bi-directional SMT
2. Translation by SMT
SMT
4. Filter out sentence pairs using the MT evaluation metric
3. Calculate sentence similarity using MT evaluation metric
5. Retrain SMT
Translated corpus
Translated corpus (source language)
Translated corpus (target language)

Fig. 2 Flow of JENAAD corpus experiments

To evaluate the effectiveness of the MT evaluation measure in sentence alignment, we compare the system performance before and after filtering. If the measure is useful for sentence alignment, we can reduce the size of the training set for the SMT without degrading the performance, by filtering out unreliable sentence pairs.

We used the JENAAD corpus, which is a Japanese-English newspaper corpus aligned by a conventional sentence alignment method (Utiyama & Isahara 2003). We use 150,000 sentence pairs from the corpus. For the SMT training, we use a Pharaoh training toolkit and an SRI language model toolkit.

Experimental results. Table 1 lists the results of the JENAAD corpus experiments. The BLEU score of the 500-sentence-pair test set is calculated in a manner similar to that in previous experiments. Here, the baseline system is trained on the sentence pairs that are filtered using the measure proposed by Utiyama et al. (Utiyama & Isahara 2003). As indicated in the table, the performance of the proposed method is comparable to that of the conventional sentence alignment measure.

Experiments using Wikipedia

Experimental settings. In the experiments using Wikipedia, we execute the first loop of the components shaded in gray[2], shown in Fig. 1, in order to verify the relationship between the yield ratio and the alignment quality. The other experimental conditions are as follows:

-The Wikipedia version is Sept. 2007.

-The number of sentences from Japanese Wikipedia is 1,500.

-The sentences of Japanese Wikipedia are preprocessed by a Chasen morphological analyzer.

-The number of sentences from English Wikipedia is 50,000,000.

[2] In these experiments, we used a commercial MT system instead of SMT.

Table 1 Results of JENAAD corpus

Method	# of the training sentence pairs	BLEU score
Proposed method	50000	0.1019
Proposed method	100000	0.109
Baseline	50000	0.1069
Baseline	100000	0.1057
Baseline	150000	0.1092

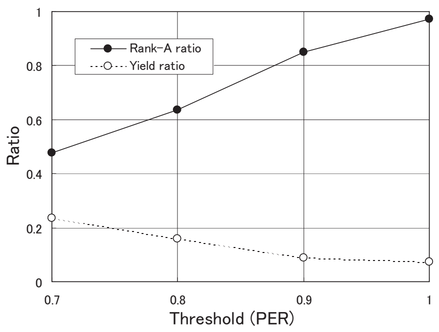


Fig. 3 Results of Wikipedia experiments

To align the Japanese and English sentences (step 3 in Fig. 1), first, we calculate the sentence similarity between Japanese sentences and all 50,000,000 Japanese translated sentences. We consider that original English sentence of a Japanese translated sentence, which gives the highest similarity value. For English sentences extracted from English Wikipedia articles as the translations of 1,500 Japanese sentences, we carry out a two-grade subjective evaluation using the following definition.

Rank A: more than 60% overlap.

Rank B: less than 60% overlap.

Experimental results. Figure 3 plots the results obtained by the Wikipedia experiments. In this figure, the vertical axis denotes the ratio and the horizontal axis shows the cutoff threshold for a translation pair. The blank circles in the figure indicate the yield ratio, which is the ratio of the obtained parallel sentences to all Japanese sentences (1,500 sentences). The filled circle indicates the rank-A ratio, which is the ratio of rank-A sentence pairs to all obtained sentence pairs. The results shown in the figure reveal that we can obtain the English translation of 10% of Japanese sentences while maintaining high alignment quality (rank-A ratio of over 0.8).

Related Works

Some previous researches have attempted to extract sentence pairs from comparable corpuses. Some of these researches require manually built language resources such as a bilingual thesaurus or lexicon to enhance the alignment performance (Utiyama & Isahara 2003; Ma 2006). However, our method requires only an initial sentence-aligned corpus.

There are two researches that propose concepts similar to our proposed concept (Fung & Cheung 2004; Munteanu & Marcu 2006). A common feature of these researches (Fung & Cheung 2004) is that they apply the bootstrapping framework for sentence alignment. Our proposed method uses a bilingual lexicon for sentence alignment and automatically updates the lexicon using the aligned sentences by the method by Fung and Cheung and that by Munteanu and Marcu These methods (Fung & Cheung 2004; Munteanu & Marcu 2006) use some elemental technology of STM to build a bilingual lexicon automatically; however, the entire STM technology has not been used. This is one of the differences between the proposed method and the conventional method.

Conclusions

We have proposed an MT bootstrapping method that simultaneously generates an STM and a sentence-aligned parallel corpus. This method iterates the following steps: (1) translation of a comparable corpus using the SMT, (2) sentence alignment of the comparable corpus using the MT evaluation measure, and (3) SMT training.

To test the effectiveness of the proposed method, first, we conducted preliminary experiments using a newspaper corpus. According to the experimental results, we thought that the sentence alignment based on the MT evaluation measure was effective and performed comparably to the conventional sentence alignment method.

Second, we performed sentence alignment experiments using English and Japanese Wikipedia articles.
The results of these experiments show that we can obtain the English translation of 10% of Japanese sentences while maintaining high alignment quality (rank-A ratio of over 0.8).

Future Works

Currently, we are performing actual MT bootstrapping experiments shown in Fig. 1 by using an HPC2500 supercomputer (HPC2500, 2002), which has 11 nodes with 128 CPUs in each node.

References

Erdmann, M., Nakayama, K., Hara, T., and Nishio, S. 2008. An Approach for Extracting Bilingual Terminology from Wikipedia, *Proc. of DASFAA-2008*.

HPC2500. 2002. http://pr.fujitsu.com/en/news/2002/08/22.html

Papineni, K., Roukos, S., Ward, T., and Zhu, W. 2000. Bleu: A Method for Automatic Evaluation of Machine Translation, *Proc. of ACL-2002*, pp. 311–318.

Utiyama, M., and Isahara, H. 2003. Reliable Measures for Aligning Japanese-English News Articles and Sentences. *Proc. of ACL-2003*, pp. 72–79.

Ma, X. 2006. Champollion: A Robust Parallel Text Sentence Aligner. *Proc. of LREC-2006*. pp.489-492.

Fung, P., and Cheung, C. 2004. Mining Very-Non-Parallel Corpora: Parallel Sentence and Lexicon Extraction via Bootstrapping and EM. *Proc. of EMNLP-2004*. pp.57-63.

Munteanu, D., and Marcu, D. 2006. Extracting Parallel Sub-Sentential Fragments from Non-Parallel Corpora. *Proc. of ACL-2006*, pp. 81–88.

Powerset's Natural Language Wikipedia Search Engine

Tim Converse, Ronald M. Kaplan, Barney Pell, Scott Prevost, Lorenzo Thione, Chad Walters

Powerset, Inc.
475 Brannan Street
San Francisco, California 94107
{converse, kaplan, barney, prevost, thione, chad}@powerset.com

Abstract

This demonstration shows the capabilities and features of Powerset's natural language search engine as applied to the English Wikipedia.

Powerset has assembled scalable document retrieval technology to construct a semantic index of the World Wide Web. In order to develop and test our technology, we have released a search product (at http://www.powerset.com) that incorporates all the information from the English Wikipedia. The product also integrates community-edited content from Metaweb's Freebase database of structured information. Users may query the index using keywords, natural language questions or phrases. Retrieval latency is comparable to standard key-word based consumer search engines.

Powerset semantic indexing is based on the XLE, Natural Language Processing technology licensed from the Palo Alto Research Center (PARC). During both indexing and querying, we apply our deep natural language analysis methods to extract semantic "facts" -- relations and semantic connections between words and concepts -- from all the sentences in Wikipedia. At query time, advanced search-engineering technology makes these facts available for retrieval by matching them against facts or partial facts extracted from the query.

In this demonstration, we show how retrieved information is presented as conventional search results with links to relevant Wikipedia pages. We also demonstrate how the distilled semantic relations are organized in a browsing format that shows relevant subject/relation/object triples related to the user's query. This makes it easy both to find other relevant pages and to use our Search-Within-The-Page feature to localize additional semantic searches to the text of the selected target page. Together these features summarize the facts on a page and allow navigation directly to information of interest to individual users.

Looking ahead beyond continuous improvements to core search and scaling to much larger collections of content, Powerset's automatic extraction of semantic facts can be used to create and extend knowledge resources including lexicons, ontologies, and entity profiles. Our system is already deployed as a consumer-search web service, but we also plan to develop an API that will enable programmatic access to our structured representation of text.