From Unstructured Syllabi to Structured: Towards an Intelligent Educational System

Xiaoyan Yu, Manas Tungare, Weiguo Fan, Manuel Pérez-Quiñones, Edward A. Fox
Virginia Tech
{xiaoyany, manas, wfan, perez, fox}@vt.edu

William Cameron, GuoFang Teng, Lillian Cassel
Villanova University
{william.cameron, guofang.teng, lillian.cassel}@villanova.edu

Abstract

Syllabi are important documents created by instructors for students. Students use syllabi to find information and prepare for class. Instructors often need to find similar syllabi from other instructors to prepare new courses or to improve their old courses. Thus, gathering the syllabi that are freely available, and creating useful services on top of the collection, will provide added value for the educational community. However, gathering and building a collection of syllabi is not a trivial task as it is complicated by the unstructured nature of syllabus representation and lack of a unified vocabulary in syllabus construction by different instructors. In this paper, we propose an intelligent system to collect and annotate automatically freely-available syllabi from the Internet to benefit our educational community by supporting services such as semantic search. We discuss our detailed process for automatically converting unstructured syllabi to structured ones, through effective information extraction, segmentation, and classification. Our evaluation results proved the effectiveness of our extractor.

1 Introduction

A course syllabus is the skeleton of a course. One of the first steps taken by an educator in planning a course is to construct a syllabus. Later, a syllabus can be improved by borrowing information from other relevant syllabi. Students prepare for a course by reading a course syllabus to identify textbooks. Students may use the syllabus to identify course policies, assignment deadlines, etc., during a school semester. A life-long self-learner identifies the basic topics of a course and the popular textbooks by comparing syllabi from different universities. A syllabus is thus an essential component of the educational system.

Free and fast access to a collection of syllabi would have a significant impact on education. Furthermore, structured search and retrieval of information from a syllabus is a desired goal if we are to support all the activities mentioned above. One positive example is the MIT OpenCourseWare[1] project which provides free access to MIT course materials to educators, students, and self-learners. It collects 1,400 MIT course syllabi

[1] http://ocw.mit.edu/

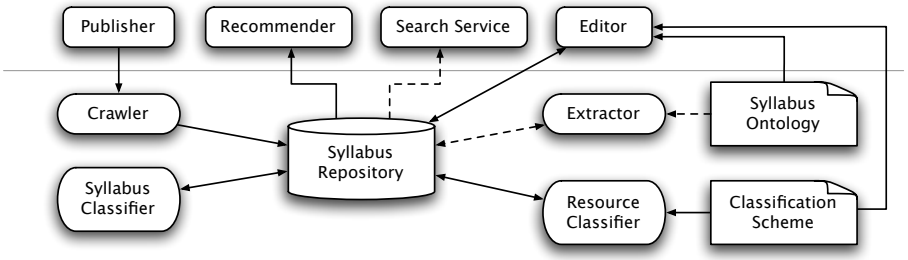


Figure 1: System architecture.

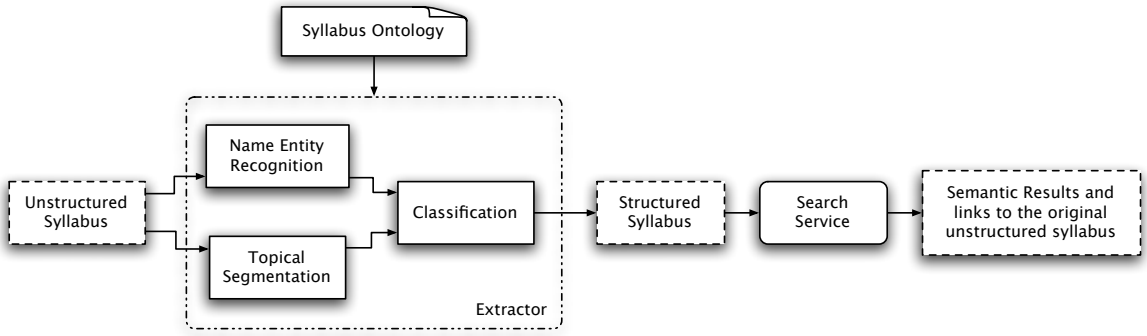


Figure 2: Workflow from an unstructured syllabus to a structured syllabus.

and publishes them in a uniform format. The project is a good start towards making a wide and open collection of syllabi.

Unfortunately, most syllabi are published on the Web in an unstructured format (e.g., HTML or PDF) without the use of a standard metadata representation. In some cases, syllabi are hidden behind password-protected course management systems, and are available only to local users. Searching for a syllabus on the Internet is also an error-prone process and often yields too many irrelevant links. Motivated by the above observations, we believe that designing a system that automates, or partially automates, collecting and tagging syllabi would be beneficial to the educational community.

Thus, in this work, we seek to create a digital library collection of Computer Science syllabi. This paper presents our system design and approaches towards making this collection available to the educational community. We describe a system that collects, annotates, and semantically indexes syllabi, and offers several services to the educational community. We restrict our focus to computer science syllabi offered by universities in the USA as a starting point of our proof-of-concept project. The methodology and the system could be extended easily to other disciplines.

Our overall system, shown in Figure 1, has several major components. A crawler [1] creates a potential syllabus repository as a result of searching the World Wide Web. It also allows individuals to submit URLs to be included in the next crawl. A syllabus classifier [1] filters noise out from the potential syllabus repository. With the guide of a syllabus ontology, an extractor generates structured information (e.g., course names, textbook, course description, etc.) from the unstructured syllabi. People can input their syllabi through our editor service. The system provides a search interface for users to search for syllabi, as well as several information syndication options.

The focus of this paper is to describe the flow marked in dashed arrows in Figure 1, which is the process from an unstructured syllabus to a structured syllabus. Figure 2 shows the major modules involved in the flow. Following the syllabus ontology, and considering the vocabulary of a syllabus, semantic information can be

Syllabus	
Data Type	Object Type
policy	assignment
affiliation	resource
∗category	courseCode
title	grading
objective	teachingStaff
description	specificSchedule
courseWebsite	prerequisite
	textbook
	exam
	generalSchedule

Table 1: The first level of the syllabus ontology, also the list of properties to be extracted except category marked by ∗.

extracted from a syllabus and become part of the Semantic Web[2]. The name entity recognition module identifies name entities such as people and dates. The topical segmentation module identifies the boundary of a syllabus property such as a course description and a grading policy. The classification module finally associates a list of syllabus properties with the segmented values and stores them in the structured syllabus repository. These three modules work together as the extractor of our system. The search service indexes structured syllabi and provides semantic search results in RDF [3] and links to the raw syllabi.

The rest of the paper proceeds by introducing the syllabus ontology in Section 2 and discussing the different extractor modules in Section 3. Section 3.4 presents our initial system evaluation. We also present some of the user services built upon our system in Section 4.

2 Syllabus Ontology

Our syllabus ontology is designed to help reconcile the different vocabularies used by different instructors, who often use their own styles and vocabularies to describe the different parts of a syllabus. For example, instructors often start a course description with headings such as 'Description', 'Overview', or 'About the Course'. Such variations make it difficult to reuse these syllabi. It is also very hard to search for a particular syllabus because the section headings are not uniquely named. In order to facilitate processing of syllabi by different applications, we propose a syllabus ontology[4] developed with the aid of the ontology editor Protégé[5]. The first level of the ontology is shown in Table 1. Among these 17 properties, some of them are data types of a syllabus such as *title* (a course title) and *description* (a course description) while others are object types such as *teachingStaff*, and *specificSchedule* that utilize other vocabularies at a deeper level. It is worth noting that we define a specific schedule as topics and specific dates to cover them, and a general schedule as semester, year, class time, and class location. The defined ontology will help our extraction task (see Section 3) to extract the list of property values (excluding the category property) from each syllabus and to make the collection of structured syllabi available in RDF.

[2]http://www.w3.org/2001/sw/

[3]http://www.w3.org/RDF/

[4]http://doc.cs.vt.edu/ontology

[5]http://protege.stanford.edu/

Entity	Interested Property	Example
People	name of teaching staff	Dr. Mary Peters
Date	semester class time, office hour schedule date	Fall 2006 Tue/Thurs, 1:00pm -2:00pm Nov 01
Organization	university department	University of A Department of Computer Science
Location	mailing list, contact email home page, course web site	mary@a.edu www.a.edu/ mary/cs2604

Table 2: Name entities extracted from a syllabus.

3 Information Extraction

Information extraction aims to extract structured knowledge, especially entity relationships, from unstructured data. In our case, we extract relations on a course such as an instance of TEACH relation "(Mary, Data Structure, Fall 2006)" from its syllabus, "(Mary teaches the Data Structure course in Fall 2006)". There are plenty of research studies, reviewed in [2], that applied machine learning technology to the information extraction task. These approaches can be broadly divided into ruled-based approaches such as Decision Tree, and statistics-based approaches such as Hidden Markov Model (HMM). The extraction task usually involves five major subtasks: segmentation, classification, association, normalization, and deduplication [2]. For our extractor, the segmentation task include mainly two steps - name entity recognition and topical segmentation - while the deduplication task is integrated into the classification task.

Thompson *et al.* [3] have tried fulfilling these tasks with an HMM approach on course syllabi for five properties: course code, title, instructor, date, and readings. They manually identified the five properties on 219 syllabi to train the HMM. However, it would take us much more effort to label 16 properties for a collection of unstructured syllabi. Therefore, we needed a method that is unsupervised, i.e., not requiring training data. In the next subsections, we explain our choice in detail.

In addition, the association task in [2], which determines which extracted information belongs to the same record given a web page with a list of courses, is not required in our extractor, since we aim to extract information given a syllabus. The normalization task, which forms extracted information in a standard format such as formatting "3:00pm-4:00pm" and "15:00-16:00" uniformly as "15:00-16:00" for the class time, will be performed in the future since it does not affect extraction accuracy.

3.1 Name Entity Recognition

Name Entity Recognition (NER), a subtask of information extraction, can recognize entities such as persons, dates, locations, and organizations. An NER F-measure of around 90%, (a combination of the precision and the recall of recognition), has been achieved since the 7th Message Understanding Conference, MUC [6], in 1998. We therefore chose to make our name entity recognizer based on ANNIE[7] of the GATE natural language processing tool [4], which has been successfully applied to many information extraction tasks such as in [5] and is easily embedded in other applications. Table 2 shows the name entities extracted through the recognizer.

[6] http://www-nlpir.nist.gov/related_projects/muc/

[7] http://www.aktors.org/technologies/annie/

3.2 Topical Segmentation

A course syllabus might talk about many different aspects of the course such as topics to be covered, grading policy, and readings. Because such information is usually expressed in sentences, NER is not applicable for such an extraction task. In order to extract such information, it is basically to find the boundary of a topic change and then to classify the content between identified boundaries into one of the syllabus data/object types. The first half falls in the topical segmentation task and the other half will be described in the next section. Much research work has already been done for the topical segmentation. We chose C99 [6] because it does not require training data and has comparable performance with the supervised learning approach which requires training data [7]. C99 measures lexical cohesion to divide a document into pieces of topics. It requires a pre-defined list of preliminary blocks of a document, and calculates cosine similarity between the blocks by stemming and removing stop words of each block. After the contrast enhancement of the similarity matrix, it partitions the matrix successively into segments.

C99 is not good, however, at identifying a short topic, which will be put into its neighboring segment. Therefore, we do not expect the segmenter to locate a segment with only a single syllabus property but expect it not to spread a syllabus property value into different segments. It is also critical to define a correct preliminary block which is the building block of a topical segment of C99. We defined a preliminary block at the sentence or the heading level. A heading is a sequence of words ahead of a syllabus property. We first located possible headings and sentences. If two headings are next to each other, the first one will be a preliminary block; otherwise a heading and the following sentence form a preliminary block in case they are partitioned into different segments.

3.3 Classification

Given topical segments and name entities of a syllabus, the final step is to associate them with the list of interesting syllabus properties in Figure 1 through classification. We defined the classification heuristic rules as follows.

1. A course code, a semester, and a course affiliation (university and department) appear at the top of a syllabus, i.e., in the first segment. A course title is a heading and follows a course code. A course code identified in the prerequisite section will be eliminated.

2. Information about teaching staff might include their names, emails, home pages URLs, phone numbers, and office hours. They should fall in the same segment.

3. For other syllabus properties, each starts with the heading of the property and falls into a single topical segment. A heading is identified based on a list of keywords. For example, a course description heading might contain ‘description’, ‘overview’, ‘abstract’, ‘summary’, ‘catalog’, and ‘about the course’.

4. For a segment without a heading, it is marked as UNKNOWN.

3.4 Evaluation

To evaluate the accuracy of the information extraction and conversion process, we randomly selected 60 out of 1000+ syllabi from our syllabus repository and removed all HTML tags to get the free text of each syllabus document. The free text was then fed into our extractor and its syllabus properties based on the list shown in Table 1 were recognized and automatically extracted. By comparison of extracted information and the original syllabus, we manually judged the correctness of extraction. Figure 3 summarizes our evaluation results. Our judgment criterion is that a piece of information is extracted correctly for a syllabus property if it is identified at the right starting position of the property in the syllabus and might miss or include information that does not affect our understanding of this piece of information. For example, we judged a course title with semester information as

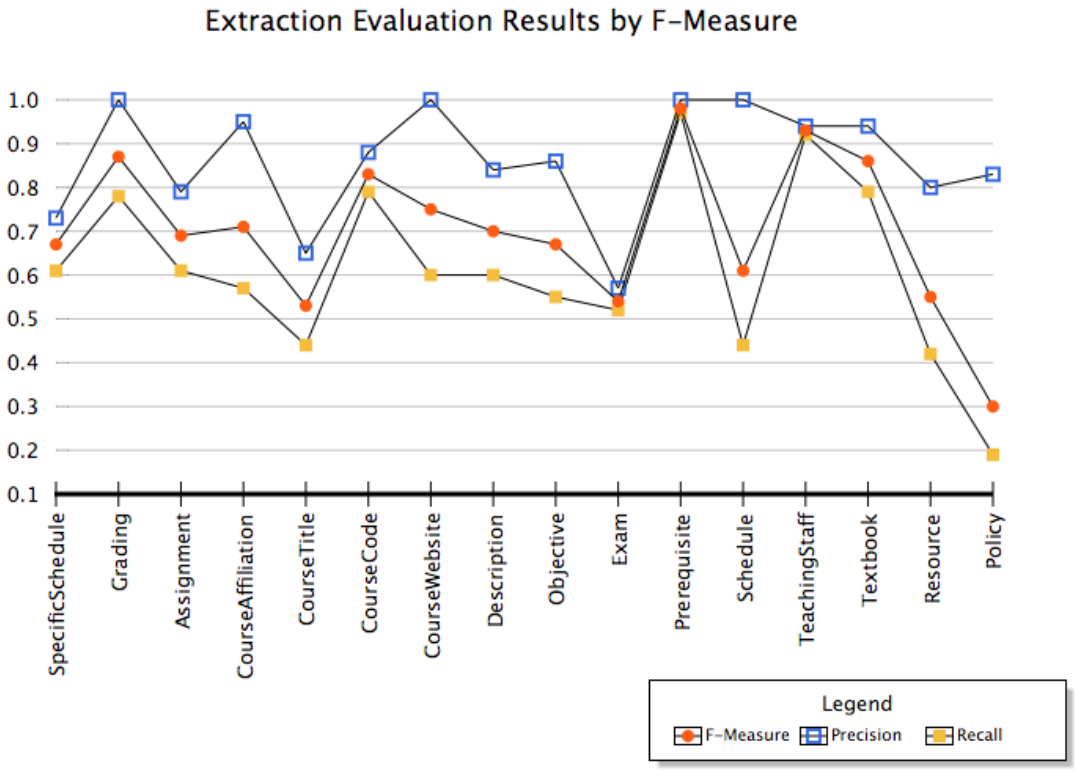
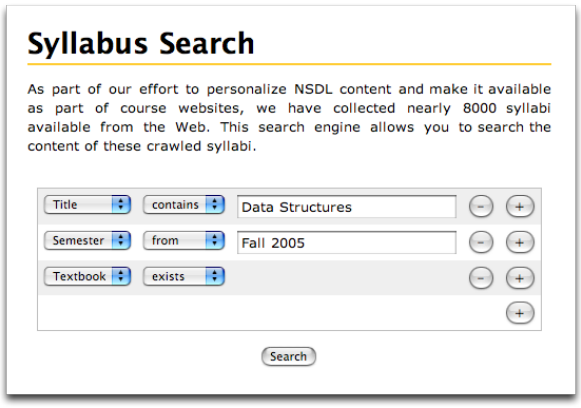


Figure 3: Extraction evaluation results by F-Measure.

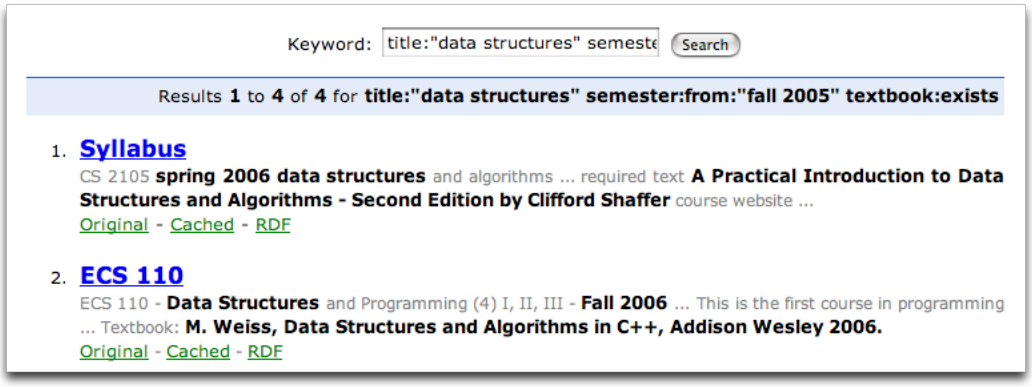
a positive extraction. F-measure was employed on each interested property by calculating precision and recall over the syllabi with this property, which is a widely accepted evaluation metric on information extraction tasks [3]. The higher the F value, the better is the extraction performance.

The results are generally promising. Our extractor is more effective on some properties than others. We achieved high accuracy on the 'prerequisite' property since this field is usually started from a heading with the keyword 'prerequisite' and contains a course code. The heuristic rule to identify a course title, the heading next to a course code, is too specific to have high accuracy. From the 60 syllabi, the extraction accuracy on the course title property can be improved by looking into the words next to a course code. If they are semester information, then a sentence next to them might be a course title. Policy and resource properties are identified with high precision but with low recall due to the same reason, that they are mis-classified to other properties such as grading and textbook. For example, many readings are given under the textbook section without an additional heading. In addition, some resources such as course-required software are hard to identify just from the heading. The accuracy on the exam property is low in terms of both recall and precision. It is also mis-classified into gradings sometimes, which leads to low recall. On the other hand, the low precision is because the exam time which belongs to a specific schedule property is mis-classified into an exam property.

The evaluation results discussed above indicate that our extractor still needs further refinement. For example, it can be improved by adjusting rules on specific properties such as course title. We can also employ more advanced machine learning approaches in this module to classify segments.



(a)



(b)

Figure 4: Syllabus search engine interface: (a) advanced search dialog; (b) search results.

4 Services made possible by the Syllabus Repository

The availability of syllabi in a standard format with the appropriate metadata extracted from them makes several new applications and services possible. We next describe several such services, as examples, in more detail.

4.1 Searching Syllabi

We provide a semantic search service over our structured syllabus repository (Figure 4). This is different from other general-purpose keyword search engines in that our search engine indexes a set of documents known with confidence to be syllabi, and provides extracted metadata to assist the user in various tasks.

For example, as shown in Figure 4, an instructor may query, from our advanced search dialog box, popular textbooks used in Data Structures courses, as recently as since Fall 2005. The search results will highlight these keywords and also textbooks, plus a link to the original unstructured syllabus, and a link to the parsed syllabus in RDF format.

4.2 Comparison Tools

Other applications that may benefit from the search service include comparison services. For example, a textbook comparison service would send a query for two textbooks and request the names of universities where these textbooks are used. The results would provide data about relative popularity, the specific courses which use these textbooks, and the salient features of those courses.

Customers considering buying the book, or booksellers wishing to stock books, or publishers trying to understand their audience better, would all be users of such a service.

4.3 Unified View of Multiple Syllabi

Metadata extracted from a syllabus is made available in a standard format, regardless of the layout and presentation of the original source document. This allows users to leverage their familiarity with one particular presentation of a syllabus (e.g., the one adopted at their own school) to understand syllabi from other schools better. Informal comparisons can be made more easily between syllabi that "look" similar, as compared to those that look totally different.

4.4 Browsing by Course Names

Similar courses at different universities are often named similarly. Being able to look at a list of all courses, organized by course names, opens up the doors to compare them in a way that has not yet been possible. Utilizing the Unified View outlined above, syllabi from multiple universities can be placed side-by-side for a subjective comparison of their content and resources.

It is also possible to visualize courses such as via histograms of most popular topics, or most popular knowledge units used in the syllabus corpus.

4.5 Syndication in Standard Formats

Many applications today can read syndicated data from disparate data sources or other applications. RSS, Atom, and others have emerged as popular choices for syndicating web content. Similarly, we see value in syndicating syllabus content in formats that may be easily consumed by other services or applications. Open syndication of this harvested data will likely lead to greater adoption of authoring tools [1] for syllabus schema.

4.6 Personalization

A structured course syllabus can be used as a profile for the personalization service [8] for the course, such as for recommending papers about topics to be covered in the course. Our personal NSDL[8] project[9] aims to bring NSDL resources closer to teachers and students through course web sites. Given a structured syllabus, our service infers what a course is about and delivers relevant resources in context.

4.7 Assisting Instructors when Creating New Syllabi

When an instructor creates a new syllabus, there is a high likelihood of the end-product being similar to other syllabi (or perhaps a combination of two or more syllabi). Although the similarity between two syllabi at the same institution is likely to be minimal, the availability of syllabi from other institutions increases the chances of finding a match. The instructor can be assisted in one or more of the following tasks:

- Recommending: The relative number of hours to be assigned to each knowledge unit, as per standardized classification schemes for syllabi, or simply based on what other courses do with the particular knowledge unit;
- Locating books, reading material, and other resources that similar syllabi have included;
- Enabling the instructor to import material from other syllabi (licensed appropriately) instead of having to recreate the entire syllabus involving extra effort (e.g., by copying the full text reference from another syllabus);
- Identifying possible contact persons to discuss and share experiences in a particular course.

5 Conclusions

In this paper, we proposed an intelligent system to automatically collect and annotate freely-available syllabi from the Internet to benefit the educational community by supporting services such as semantic search. We discussed our detailed process on how we can automatically convert unstructured syllabi to structured through effective information extraction, segmentation, and classification. Our evaluation results proved the effectiveness of our extractor and indicated the future work to improve the extraction task.

6 Acknowledgments

This work was funded by the National Science Foundation under DUE grant #0532825.

References

[1] Tungare, M., Yu, X., Cameron, W., Teng, G., Pérez-Quiñones, M., Fox E., Fan W., Cassel L. 2007. Towards a Syllabus Repository for Computer Science Courses. (To appear in) Proc. Special Interest Group on Computer Science Education (SIGCSE) 2007.

[2] McCallum, A. 2005. Information Extraction: Distilling Structured Data from Unstructured Text. ACM Queue, Volume 3, Number 9, November 2005.

[8] National Science Digital Library, http://nsdl.org/

[9] Project website http://doc.cs.vt.edu

[3] Thompson, C. A., Smarr, J., Nguyen, H., and Manning, C. 2003. Finding educational resources on the web: Exploiting automatic extraction of metadata. In Proc. ECML Workshop on Adaptive Text Extraction and Mining, 2003.

[4] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July 2002.

[5] Dowman, M., Tablan, V., Cunningham, H., and Popov, B. 2005. Web-assisted annotation, semantic indexing and search of television and radio news. In Proceedings of the 14th International Conference on World Wide Web (Chiba, Japan, May 10 - 14, 2005). WWW '05. ACM Press, New York, NY, 225-234. DOI= http://doi.acm.org/10.1145/1060745.1060781.

[6] Choi, F. Y. 2000. Advances in domain independent linear text segmentation. In Proceedings of the First Conference of North American Chapter of the Association for Computational Linguistics (Seattle, Washington, April 29 - May 04, 2000). ACM International Conference Proceeding Series, vol. 4. Morgan Kaufmann Publishers, San Francisco, CA, 26-33.

[7] Kehagias, A., Nicolaou, A., Petridis, V. and Fragkou, P. Text Segmentation by Product Partition Models and Dynamic Programming. Mathematical and Computer Modelling, 39, Issues 2-3, (January 2004), 209-217.

[8] Gordon, M., Fan, and W., Pathak, P. 2006 Adaptive Web Search: Evolving a Program That Finds Information. IEEE Intelligent Systems 21, 5 (Sep. 2006), 72-77. DOI= http://dx.doi.org/10.1109/MIS.2006.86.