Automatic Extraction Method for Talent Knowledge Structure in Network Environment (Postprint)

Authors: Liu Qingxiang, Zhang Pengzhu, Zhang Xiaoyan, Liu Jingfang

Date: 2017-10-11T00:00:00+00:00

Abstract

[Objective] To construct an automatic extraction method for talent knowledge structure. [Method] Based on web information collection technology, web page analysis, text segmentation, and semantic web related technologies, an automatic extraction system for talent knowledge structure in network environments was constructed. [Result] Experiments verified the usefulness of the system: the overall accuracy of course recognition exceeded 95%; for semi-structured files, the recall rate was above 95%; for unstructured files, the recall rate of some files was below 90%. [Limitation] The recall rate of course recognition is constrained by the content of the lexical database. [Conclusion] This method can provide a useful tool for talent knowledge structure research and meets the basic requirements for constructing talent knowledge structure.

Full Text

Preamble

Automatically Extracting Talents' Knowledge Structure Online

Liu Qingxiang[1], Zhang Pengzhu[1], Zhang Xiaoyan[2], Liu Jingfang[3]
[1](Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai 200030, China)
[2](School of Management, Shanghai University of Engineering Science, Shanghai 201620, China)
[3](School of Management, Shanghai University, Shanghai 200444, China)

Abstract

[Objective] This study aims to construct an automatic method for extracting talents' knowledge structures. **[Methods]** Based on web information collection

technologies, webpage analysis, text segmentation, and semantic web technologies, we developed an automated system for extracting talent knowledge structures from online environments. [**Results**] Experiments validated the system' s utility, achieving over 95% overall accuracy in course recognition. For semi-structured documents, the recall rate exceeded 95%, while for unstructured documents, some files showed recall rates below 90%. [**Limitations**] The recall rate of course recognition is constrained by the content of the dictionary database. [**Conclusions**] This method provides a useful tool for talent knowledge structure research and meets the fundamental requirements for constructing talent knowledge structures.

Keywords: Web-based outsourcing; Competency; Knowledge structure; Automatic extraction
Classification Codes: C931; G35

Web-based innovation service outsourcing refers to the practice where enterprises leverage external human resources via the internet to complete innovation tasks [1]. To achieve rapid and precise matching between innovation service supply and demand—effectively aligning talent capabilities with enterprise tasks—accurate description of talent competency is crucial. In 1973, the renowned American psychologist McClelland first proposed the concept of talent competency [2], upon which Mirable further developed the KSAO model [3]. The "K" represents Knowledge, referring to job-specific and domain-specific expertise required for particular positions, such as professional knowledge. This research focuses primarily on the automatic extraction of talents' professional knowledge structures.

Current methods for obtaining talent knowledge structure information remain limited to manual entry, which cannot meet the demand for acquiring large-scale data in short timeframes and incurs relatively high costs. Developing an automated approach for information collection, analysis, and extraction is therefore essential.

The objective of this research is to construct a method that automatically extracts talent knowledge structure information from web resources. By leveraging abundant online resources, the method collects and analyzes talent knowledge structure information and automatically extracts usable data for talent knowledge base construction. The output format adopts semantic web and domain ontology technologies and standards to facilitate flexible utilization of talent knowledge structure information.

Building a comprehensive talent knowledge base is fundamental to matching talent capabilities with outsourcing tasks. Suppliers need only provide basic information such as a talent' s university, major, and enrollment year, and the system can rapidly retrieve structured knowledge structure information from the backend, including courses taken and course content, enabling comparative analysis across different talent resources. The functionality of the talent knowledge structure information automatic extraction system is illustrated in [Figure

1: see original paper].

2. Background Analysis of Automatic Talent Knowledge Structure Extraction

(5) High Content Variability: Due to different educational histories and institutional characteristics, most Chinese universities currently publish information about their faculties, majors, and courses through official websites and academic affairs portals, providing rich information sources that serve as the primary objects of analysis in this study.

2.1 Typical Sample Analysis

After analyzing typical samples from internet sources, we identified several key characteristics: **(1) Massive Information Volume:** The total number of courses offered is estimated to be on the order of 10^7 to 10^9. Considering additional factors such as enrollment years, manual compilation becomes extremely difficult. **(2) Frequent Changes:** In educational practice, universities adjust their faculties, majors, and training programs annually, creating further challenges for information compilation. **(3) Diverse File Types:** Universities publish training program information in various formats, including HTML, PDF, Word, Excel, and others, requiring system design to accommodate multiple file formats. **(4) Significant Semi-structured/Unstructured Features:** The internal organizational structures for faculty, major, and course information vary widely across documents, appearing in both tabular formats (see [Figure 2: see original paper]) and free-text descriptions (see [Figure 3: see original paper]).

Overall, the information organization exhibits semi-structured or unstructured characteristics. System design should differentiate processing approaches based on file structure to improve information processing accuracy. Course offerings vary dramatically across institutions. Taking the Management Information Systems major as an example, curriculum design differs substantially across universities [4], with some programs belonging to computer science faculties and others to management faculties. System design must account for such variability to avoid format compatibility issues.

2.2 System Design Requirements and Technical Challenges

(1) Minimize Manual Intervention: The massive information volume and frequent changes make manual entry and compilation extremely costly. Therefore, information acquisition and knowledge structure construction must be automated to reduce manual intervention as much as possible. While web information collection technologies can meet these needs, building a web crawler system from scratch involves enormous workload and complexity, and existing frameworks often fail to meet personalized requirements. Leveraging mature

open-source frameworks as a development foundation should be fully considered.

(2) Compatibility Requirements: The diversity of file types and semi-structured/unstructured characteristics demand system compatibility with different files and structural types. This imposes high requirements for modularity and reusability in system design and implementation, necessitating forward-looking planning and well-extensible design.

(3) Data Format Extensibility Requirements: Content variations in professional curricula across different universities impose requirements on system data format design, demanding flexibility in data storage and transformation during knowledge structure construction. Traditional relational databases offer relatively single storage methods with limited extensibility; data formats such as XML, widely used in modern web applications, should be considered.

(4) Data Accuracy Requirements: Information accuracy, including precision in acquisition and extraction, is key to ensuring the utility of talent knowledge bases. While automated web information collection programs can rapidly and batch-obtain large volumes of web resources, this approach inevitably introduces issues of low information purity. System analysis and design should consider measures such as centralizing data sources, constructing domain dictionaries to exclude irrelevant information, and intermediate data screening to mitigate these adverse effects.

2.3 Output Result Requirements

Given the research background, to enable intelligent and flexible subsequent utilization of resources, output results must be easily readable and understandable by computers, possessing certain semantic features and ontology implementation. The ontology construction process involves analyzing entities and object-properties, with preliminary analysis of relevant entity relationships as follows:

(1) Primary Entity Analysis The study involves four entity types: university, faculty, major, and course: **University Entity:** A university comprises multiple faculties and has attributes such as name, code, and introduction. **Faculty Entity:** A faculty uniquely belongs to one university and offers multiple majors, with its own profile attribute. **Major Entity:** A major is offered under a faculty (not necessarily exclusively) and includes many courses, both required and elective, with attributes such as overview and description. **Course Entity:** A course is offered in a major' s training program and may also be offered in other majors, with a content description attribute.

(2) Primary Relationship Analysis Potential relationships between entities include offering, being offered, affiliation, and possession: **opens Relationship:** An offering relationship applicable from university to faculty, faculty to major, and major to course, indicating an offering or possession relationship but

not exclusive ownership. **is_{{{opened}}_{{by}}} Relationship:** A being-offered relationship representing subordination (e.g., a course may belong to multiple majors) but not exclusive subordination. This is the inverse of opens, applicable from major to faculty and course to major. **associates_{to} Relationship:** An exclusive subordination relationship applicable from faculty to university (faculties with identical names in different universities are treated as distinct entities). **opens_{{{as}}*{{required}}} and opens*{{{as}}_{{optional}}} Relationships:** Required and elective offering relationships, inheriting from opens and applicable from major to course to distinguish whether a course is required or elective within a major.

(3) Conceptual Entity and Attribute Relationship Diagram The conceptual entity and attribute relationships are shown in [Figure 4: see original paper].

3. System Architecture

3.1 Overall Architecture

(1) Information Acquisition: A web crawler program automatically collects large volumes of talent knowledge structure information from the internet. **(2) Preliminary Conversion:** Text parsing programs perform initial conversion of heterogeneous raw data into corresponding text data. **(3) Knowledge Structure Extraction:** Faculty, major, and course entities are identified from text data of different structural types to construct corresponding in-memory data. **(4) Data Dumping:** To facilitate subsequent data utilization, in-memory data is persisted as intermediate data with strong extensibility. **(5) Semantic Data Construction:** Based on intermediate data, an ontology construction program generates semantic data that can supplement dictionary content and improve system information utilization.

Thus, data undergoes transformation from raw data to text data, in-memory data, intermediate data, and finally semantic data, as shown in [Figure 5: see original paper]. The overall system architecture is illustrated in [Figure 6: see original paper].

3.2 Web Crawler Program

Web resource traversal typically employs breadth-first or depth-first algorithms. This study adopts breadth-first search, as valuable professional curriculum information generally resides closer to data sources. In addition to selecting appropriate search algorithms, proper URL and file filtering rules should be followed to improve crawling efficiency: **(1) URL Filtering:** Since this study sources information from Chinese university official websites, webpages without "edu" in their URLs are filtered out. **(2) File Filtering:** For HTML pages, the program sets a keyword set and matches it against text content. HTML pages containing none of the keywords are not crawled; the keyword set includes terms such as "major," "faculty," and "course."

3.3 File Parsing Program

Given the diversity of raw data formats (HTML, PDF, Office types like Word and Excel), direct utilization is difficult. Therefore, technical measures are necessary to parse various files into uniformly usable text data. The file parser program calls different parsing interfaces based on file type, with the process shown in [Figure 7: see original paper].

3.4 Text Analysis and Dumping Program

The text analysis program reads text data returned by the file parser, analyzes content, and generates in-memory objects. The text analysis process is shown in [Figure 8: see original paper]. For semi-structured text content, hierarchical major and course information can be obtained without reading the domain dictionary. The program reads table data from text, maps major and course information to in-memory objects based on header content, and—for example—maps course name, credits, and hours from training program tables to a Course object with corresponding name, credit, and period attributes (which may be null).

For unstructured text, content is relatively scattered and disordered, requiring more complex analysis based on domain dictionary matching. The program reads all text content, performs word segmentation using major and course vocabulary from the system' s domain dictionary, searches for course attribute keywords (credits, hours, description, etc.) within sentences containing course entities, locates attribute content, and maps it to Course objects. The in-memory object data for majors and courses is then dumped as intermediate data.

3.5 Ontology Building Program

Using the returned intermediate data, the program constructs semantic ontology data according to final information structure requirements, enhancing computer readability and understanding and adapting to potential flexible utilization needs. The ontology building process is shown in [Figure 9: see original paper].

4. Data Format Design

4.1 Crawler Database Design

Candidate solutions for the crawler database include relational databases such as Oracle and SQL Server, file-based structures like XML and RDF, and embedded databases such as Berkeley DB. Considering access efficiency, security, and massive data load requirements, Berkeley DB was selected. Berkeley DB manages databases in key/value format, providing high access efficiency through APIs that retrieve corresponding data via keywords. Its underlying structure can be understood as a HashMap storing large volumes of data with O(1) access complexity, significantly outperforming relational and file-based databases.

Berkeley DB structures include: **(1) BdbFrontier:** A link factory using Berkeley DB structure in Heritrix to verify whether an object awaiting queue entry has been crawled. **(2) BdbMultipleWorkQueues:** A set of link object queues with different Key values, forming "Key/Value" pairs as records in Berkeley DB. **(3) BdbWorkQueue:** A link queue based on Berkeley DB, with each BdbWorkQueue assigned a key value. **(4) BdbUriUniqFilter:** A filter invoked by BdbFrontier containing a Berkeley DB database of crawled links [5].

4.2 Intermediate Data Format Design

Candidate formats include XML, SQL Server, and MySQL. Considering difficulty in transforming to semantic data and data migration convenience, XML was selected. XML files are widely considered the foundational layer for semantic web implementation, with unified syntax standards and significantly better extensibility than relational databases. The XML file specification format.xsd structure is shown in [Figure 10: see original paper].

4.3 Semantic Data Format Design

Candidate semantic data formats include OWL, XML, and RDF. Considering expressiveness requirements, OWL format is prioritized. OWL (Web Ontology Language) is part of the W3C recommendation standard stack related to the semantic web, using XML-based RDF syntax [6], with far stronger expressiveness than XML and RDF. The semantic data format design follows [Figure 4: see original paper].

5. Implementation and Operation

5.1 Programming Language and Development Environment

The system uses Java as the programming language. Compared with C and C++, Java is thoroughly object-oriented, facilitates design pattern application, and offers many mature open-source frameworks due to its cross-platform nature. Eclipse was selected as the development environment. Compared with other tools like JBuilder and IDEA, Eclipse offers openness, freedom, and numerous extensible plugins, meeting rapid development needs.

5.2 Web Crawler Program and Operation Effect

Considering the system' s high complexity and numerous personalized requirements, an open-source crawler framework was selected as the development foundation. Options included Scrapy, Cola, Heritrix, and Beautiful Soup. After considering programming language, interface friendliness, and extensibility, Heritrix was ultimately chosen.

Heritrix is an open-source, extensible web crawler project started in 2003, developed on the Java platform [7]. Compared with Scrapy and Cola, it offers more

powerful configuration functionality, better extensibility, and web-based operation with greater user-friendliness. Its basic framework is shown in [Figure 11: see original paper]. The CrawlController class coordinates module operations as the framework' s core, acting as the central nervous system that determines crawler process start and end. It comprises three major components: scope, frontier, and processor chain. The scope component determines the next URL for queue entry and allows custom intervention. The frontier component validates boundary conditions for URLs in the unvisited queue, where URL rules are set. The processor chain contains the queue of URLs being processed simultaneously, with results passed to boundary conditions.

The crawler task creation interface is shown in [Figure 12: see original paper]. This example shows a task originating from Shanghai Jiao Tong University' s undergraduate teaching information service website, with the data source specified in Seeds. After configuring parameters in the Modules and Settings sections, the task is created.

5.4 Text Analysis and Dumping Program Implementation and Operation Effect

The file parser maps files to memory by calling different interfaces and extracts usable strings, outputting text files. Using PDF parsing as an example: the PDFBox API was adopted for development. It obtains PDF documents in an object-oriented manner, treating a PDF file as a combination of basic objects including arrays, numbers, strings, and dictionaries—unlike text file streams—making it highly suitable for this research framework.

Key methods for converting text information to XML files include: `txtToXml(String in, String out)` as the main method receiving text file path and output XML file path to complete conversion; `coursesToXml(List<Course> courses, String out)` called by txtToXml to persist course object lists to XML files; and `parseOneCourse(String line)` receiving a line of text content and returning a Course object or null. The intermediate data generation program uses the Dom4j interface for XML file construction.

The test PDF file before conversion is shown in [Figure 13: see original paper], representing an architecture major' s training program document from a certain university. The converted text document is shown in [Figure 14: see original paper].

5.5 Semantic Data Construction and Test Results

Numerous ontology construction tools exist, including commercial products and research outcomes from universities and institutions [8], such as OntoEdit, WebOnto, Protégé, and WebODE. This study selected Protégé as the construction tool. Protégé is an ontology editing and knowledge acquisition software developed by Stanford University based on Java [9], with many excellently designed

plugins offering superior extensibility and user-friendliness, making it the most widely used ontology editor.

After ontology construction tool setup, test data was selected for similarity calculation. The test data description: University S has a Computer Science Department and Software Engineering School. The Computer Science Department offers a Computer Science major with required courses including C++ Programming, Data Structures, Software Engineering Introduction, and Algorithms & Complexity, plus electives such as Massive Data Processing. The Software Engineering School offers a Software Engineering major with C++ Programming, Data Structures, Software Engineering Introduction, IT Service Management, plus electives such as Middleware Technology. Test results are shown in .

6. Experimental Results and Analysis

An experiment was organized to verify the utility of the automatic extraction method.

6.1 Experimental Content

Raw data comprised training program documents from over 2,000 majors across 50 universities, totaling 834MB. Based on content structure, files were divided into two groups: semi-structured and unstructured, referring to tabular and loose-text formats respectively. This yielded 37 semi-structured file groups and 13 unstructured file groups. From each group, 10 universities were randomly selected, with 10 majors chosen per university, resulting in 200 major training programs manually segmented into 200 files. Due to varying formats, individual file sizes ranged from 10-200KB, totaling 23.1MB. These files were processed to obtain course recognition results, which were manually verified by two experimenters to calculate precision and recall rates.

For experimenter convenience, results were converted to Excel format, as shown in [Figure 15: see original paper].

6.2 Evaluation Metrics

Evaluation adopted commonly used natural language processing metrics: Precision and Recall. Precision measures information retrieval result quality (accuracy), while Recall measures comprehensiveness. Evaluation indicators are defined in .

6.3 Results and Analysis

Experimental results show the system achieved high average precision for input files, exceeding 95%. For semi-structured training program files, recall rates were high, above 99%. However, for unstructured files, average recall rates were lower, below 90%, with some files achieving only around 80% course recognition.

This primarily stems from the system' s dependence on dictionary completeness when processing unstructured files. Course vocabulary absent from the dictionary is often unrecognizable. For example, "Programming Design Methods and Thinking" is named "Computational Thinking" at some institutions; if the dictionary lacks this term, identification fails. However, as the system processes more structured/semi-structured files, dictionary content will expand, improving recall rates for unstructured files.

This study comprehensively analyzed the background of talent knowledge structure extraction in web-based innovation outsourcing and designed and implemented an automatic talent knowledge structure information extraction system. The output results support talent knowledge base construction. Relying on this research, only basic talent information is needed to rapidly obtain their university courses and descriptions, with similarity analysis provided based on final semantic data. Future data updates will no longer require extensive manual effort.

This research will contribute to future talent knowledge base and innovation task matching studies. The vast professional curriculum data will provide strong support for talent knowledge base construction, and the final semantic data will gradually improve system recall rates. In practice, knowledge structure encompasses not only knowledge acquired through university education but also skills and professional expertise gained through later training and work. The construction and storage formats for such knowledge can follow the approach demonstrated herein, with subsequent research focusing on these domains and specific professional fields.

References

[1] Li Xiaomao, Zhang Jianjun. Internet-based Outsourcing and Enterprise Innovation [J]. China Soft Science, 2003(1): 93-99.
[2] McClelland D C. Testing for Competence Rather than for Intelligence [J]. American Psychologist, 1973, 28(1): 1-14.
[3] Mirable R J. Everything You Wanted to Know About Competency Modeling [J]. Training and Development, 1997, 51(8): 73-77.
[4] He Yonggang, Huang Lihua. Research Reviews on the Curriculums for the Information Systems [J]. Journal of Information, 2007, 26(8): 128-131.
[5] Qiu Zhe, Fu Taotao. Develop Your Own Search Engine: Lucene 2.0+Heritrix[M]. Beijing: Posts & Telecom Press, 2007.
[6] Gao Zhiqiang. The Principle and Application of Semantic Web [M]. Beijing: China Machine Press, 2009.
[7] Luo Gang, Wang Zhendong. Build a Web Crawler by Yourself [M]. Beijing: Tsinghua University Press, 2010.
[8] Xu Guohu, Xu Fang. A Comparative Study of Ontology-building Tool [J]. Library and Information Service, 2006, 50(1): 44-48.
[9] Hong Na, Zhang Zhixiong. Practice of Creating and Reasoning Science Ontology by Protégé [J]. New Technology of Library and Information Service,

2009(7-8): 1-5.

Author Contributions: Liu Qingxiang: Conceptualization, system development, drafting and final revision; Zhang Pengzhu: Research conceptualization, methodology design; Zhang Xiaoyan: Experimental design, data collection, cleaning and analysis; Liu Jingfang: Final manuscript revision.

Conflict of Interest Statement: All authors declare no conflict of interest.

Supporting Data:
[1] Liu Qingxiang. knlgbuild.xls. Talent Knowledge Structure Automatic Extraction Experiment.
[2] Liu Qingxiang. crowdsourcing.zip. Talent Knowledge Structure Automatic Extraction System.
Contact: kentliu1990@163.com

Received: 2015-12-09
Revised: 2016-01-08

Automatically Extracting Talents' Knowledge Structure Online

Liu Qingxiang[1], Zhang Pengzhu[1], Zhang Xiaoyan[2], Liu Jingfang[3]
[1](Antai College of Economics and Management, Shanghai Jiao Tong University, Shanghai 200030, China)
[2](School of Management, Shanghai University of Engineering Science, Shanghai 201620, China)
[3](School of Management, Shanghai University, Shanghai 200444, China)

Abstract: [Objective] To extract talents' knowledge structure automatically. [Methods] We built an online knowledge structure extraction system based on Web information retrieval, webpage analysis, word segmentation and semantic Web technologies. [Results] We examined the usability of the new system. For course recognition, the overall precision rate was more than 95%. For semi-structured files, the recall rate was above 95%. For some non-structured files, the recall rate was below 90%. [Limitations] The recall rate of course recognition was restricted by the content of the dictionary. [Conclusions] The proposed method meets the requirements of constructing talents' knowledge structure and is a useful tool for related research.

Keywords: Web-based outsourcing; Competency; Knowledge structure; Automatic extraction

Note: Figure translations are in progress. See original paper for figures.