Smarter Wikis through Integrated Natural Language Processing Assistants

Bahar Sateli  René Witte

Semantic Software Lab
Concordia University, Montréal, QC, Canada

SMWCon Spring 2013  March 22\textsuperscript{nd}, New York City, USA
Outline

1. Introduction
2. Wiki-NLP Integration
3. Applications
4. Conclusion
I. Wikis as Collaborative Authoring Environments

- Globalization of Software Development
  - Teams are spatially and temporally apart
  - Stakeholders with various backgrounds are involved
  - Requirements Specifications are written in natural language

- Often, the generated specifications are...
  - **Ambiguous** → inherited from using natural languages
  - **Inconsistent** → difficult to maintain manually
  - **Poor quality** → accounts for 50% of project failures
II. Wikis as Knowledge Management Platforms

- Biomedical Literature Curation
  - Finding and extracting relevant knowledge from the domain literature
  - Manually refining and updating bioinformatics databases

- Manual literature curation is...
  - Expensive → requires domain experts
  - Labour-intensive → ever growing amount of scientific publications
  - Error-prone → critical knowledge can be easily missed
Desiderata

In such contexts, we need a wiki that can:

- Detect various defects in its content
  e.g., spelling mistakes, ambiguities, readability issues

- Extract entities that are relevant to a user’s interest or context
  e.g., extract all person names mentioned in the wiki

- Formally model the knowledge contained inside the wiki
  e.g., generate Semantic MediaWiki markup from unstructured wiki text

- Offer searching for content beyond keyword-based approaches
  e.g., find all articles containing an enzyme name

- Generate its own content
  e.g., create summaries from long articles
Natural Language Processing (NLP)

- A branch of Artificial Intelligence
  - uses various techniques to process content written in natural language

- Multitude of NLP techniques
  - Named Entity Recognition
  - Quality Assessment
  - Summarization

- Various NLP APIs (e.g. OpenCalais, GATE, ...)

**Natural Language Processing (NLP)**

- A branch of Artificial Intelligence
  - uses various techniques to process content written in natural language

- Multitude of NLP techniques
  - Named Entity Recognition
  - Quality Assessment
  - Summarization

- Various NLP APIs (e.g. OpenCalais, GATE, ...)
Semantic Assistants

- Service-oriented Architecture (SOA) [4]
- Publishes various NLP pipelines as W3C Standard Web services
- Open source framework (http://www.semanticassistants.com)
**Problem Statement**

Wiki System

![Wiki Engine](image)

NLP Techniques

- Information Extraction
- Automatic Summarization
- Question Answering

---

**Solve common problems**

- Wikis' Loose Structure
- Information Overload

**Introduce new features**

- Enable human-AI interaction
- Bring semantics to wiki content

---

Smarter Wikis through Integrated NLP Assistants (Bahar Sateli, René Witte)
Wiki-NLP Integration Architecture

- Seamless integration of NLP capabilities within wikis [3]
- Open-source Software (AGPLv3)
- General, semantics-based architecture
The NLP Interface

- The NLP user interface offers various text mining services
- Dynamically-generated interface
- Customizing services at runtime
- Collection-based Analysis

Text Mining Assistants inside the wiki

<table>
<thead>
<tr>
<th>Available Assistants</th>
<th>Results Target</th>
<th>Global Settings</th>
<th>Console</th>
</tr>
</thead>
<tbody>
<tr>
<td>mycolMNTE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IR Information Extractor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information Extractor</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OrganismTagger</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Step 1. Select the service you wish to execute on your collection.
Once you add this page to your collection, you can continue browsing as your collection is saved.
Solution: Wiki-NLP System Architecture

Transformation of Results

- From the Semantic Assistants server response to wiki markup

Sample Service Invocation Response

```xml
<saResponse>
  <annotation type="Location" annotationSet="Annotation" isBoundless="false">
    <document url="http://localhost/wiki/sample_page">
      <annotationInstance content="Canada" start="16" end="22">
        <feature name="locType" value="country"/>
      </annotationInstance>
    </document>
  </annotation>
</saResponse>
```
Presentation of Results

- Templating Mechanism, i.e., separating data model from its presentation

```html
1 { class="wikitable" style="height:50px"
2   width="200" | Content
3   width="80" | Type
4   width="50" style="text-align: center;" | Start
5   width="50" style="text-align: center;" | End
6   ! Features
7   |-
8     |-
9     |
10 }
```

```
1 |{|SemAssist-TableRow| content= Elizabeth Middleton | type=Person | start = 236 | end = 255 | features =
2     gender:female}
```
Introduction

2 Wiki-NLP Integration

3 Applications
  - Software Requirements Engineering
  - Biomedical Literature Curation

4 Conclusion
ReqWiki

- Semantic MediaWiki customized for collaborative Requirements Engineering (RE)
- Follows Unified Process (UP) Methodology
- Structures SRS artifacts using forms and templates
ReqWiki

► Semantic Forms as data entry point

► Embedded traceability links with {{#ask}} inline queries
ReqWiki

► Various NLP services

Available Assistants

Select a service

Information Extractor
Writing Quality
English Durm Indexer
Requirements QA Defects
Requirements QA Stats
Readability Metric Stats
ReadabilityMetrics
Person and Location Extractor

► Detect common defects and suggest solutions, where possible

► Automatically index the SRS documents

UC/Manage Tasks

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>The manager receives a customer service request. The manager directs the operation for creating, updating, deleting and querying a task. Some operations use either the automatic or manual task assignment functionality that were defined in the Supplementary Specification document.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Level</th>
<th>user-goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary Actor</td>
<td>A / Manager</td>
</tr>
<tr>
<td>Stakeholders</td>
<td>Manager, Senior technician, Junior technician</td>
</tr>
<tr>
<td>Pre-Conditions</td>
<td>The manager must be identified and authenticated in the application</td>
</tr>
<tr>
<td>Success end condition</td>
<td>The task is created and assigned to the technicians with status Assigned. The tasks is updated and assigned to the technicians with status Assigned. The task is queried. The task is deleted.</td>
</tr>
<tr>
<td>Failure end condition</td>
<td>The task is created with status Submitted.</td>
</tr>
<tr>
<td>Features</td>
<td>Manage Task</td>
</tr>
</tbody>
</table>

Writing Quality on UC/Manage_Tasks (View)

<table>
<thead>
<tr>
<th>Content</th>
<th>Type</th>
<th>Start</th>
<th>End</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>were defined</td>
<td>AtD</td>
<td>236</td>
<td>248</td>
<td>- problem: Passive voice</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- suggestion: -</td>
</tr>
<tr>
<td>must be</td>
<td>AtD</td>
<td>434</td>
<td>441</td>
<td>- problem: Passive voice</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- suggestion: -</td>
</tr>
<tr>
<td>is created</td>
<td>AtD</td>
<td>521</td>
<td>531</td>
<td>- problem: Passive voice</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- suggestion: -</td>
</tr>
<tr>
<td>The tasks is</td>
<td>AtD</td>
<td>567</td>
<td>599</td>
<td>- problem: Subject Verb Agreement</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- suggestion: The tasks are, The task is</td>
</tr>
</tbody>
</table>
User Study I

- Can text mining assistants help to improve requirements specifications? [1]
- 2 Software Engineering classes at Concordia University, Montréal
  - **Goal:** Identifying defects in manual vs. NLP-assisted requirements engineering
  - **NLP Services:** Spell checking, Readability Analysis, Passive Voice Detection, . . .
  - **Measure:** Number of defects found in assignments
  - **Method:** Comparison against NLP-assisted quality assurance

**Results:**

![缺陷分布图](image)

**Conclusion:** ReqWiki NLP capabilities were indeed effective to significantly reduce SRS defects.
User Study II

- How much NLP background do users need in order to use semantic capabilities?
- Same scenario as User Study I
- Anonymized questionnaire asking participants:
  - Their proficiency level in NLP
  - ReqWiki ease-of-use

Results:

- **Conclusion**: Concrete NLP background is not required to make use of sophisticated semantic support provided in ReqWiki.
IntelliGenWiki

- An intelligent semantic wiki for life sciences [2]
- Integrated bio-related NLP services for literature curation
- Offers basic semantic entity retrieval
Information Extraction (IE)

- Automatically extracting knowledge from text
- Various IE services
  - mycoMINE
  - OrganismTagger
  - Open Mutation Miner
  - ...
- Enrichment of literature content with semantic markup

Example:
[[hasType::Enzyme|cellulbiohydrolase]]
Semantic Entity Retrieval

- Unadorned wikis offer only keyword-based search
- What if we want to *discover* what’s contained in the wiki?
  - e.g., *“Which papers in this wiki mention an enzyme entity in their text?”*
- Exploit the semantic metadata generated by NLP services, e.g., *type* properties
  - Using Semantic MediaWiki inline queries

```wiki
{{#ask: [[hasType::Enzyme]]
| ?Enzyme = Enzyme Entities Found
| format = table
| headers = plain
| default = No pages found!
| mainlabel = Page Name
}}
```

<table>
<thead>
<tr>
<th>Property: Enzyme</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Page Name</strong></td>
</tr>
<tr>
<td>PMID: 20709862</td>
</tr>
<tr>
<td><strong>Enzyme Entities Found</strong></td>
</tr>
<tr>
<td>Collobiohydrolase</td>
</tr>
<tr>
<td>Cellulases</td>
</tr>
<tr>
<td>endoglucoamylases</td>
</tr>
<tr>
<td>β-glucosidases</td>
</tr>
<tr>
<td>InvTrogen</td>
</tr>
<tr>
<td>DNA polymerase</td>
</tr>
</tbody>
</table>
User Study

- Is the integration of text mining assistants in a wiki environment actually effective?

- User study within the Genozymes project context (www.fungalgenomics.ca)
  - **Goal:** Identifying and characterizing fungal enzymes
  - **Dataset:** 30 documents
  - **Users:** 2 expert biocurators
  - **NLP Service:** mycoMINE
  - **Measure:** Time spent on curation
  - **Method:** Comparison against time spent on manual curation

- Results:

<table>
<thead>
<tr>
<th></th>
<th>Abstract Selection</th>
<th>Full Paper Curation</th>
</tr>
</thead>
<tbody>
<tr>
<td>no support</td>
<td>IntelliGenWiki</td>
<td>no support</td>
</tr>
<tr>
<td>1 min.</td>
<td>0.3 min.</td>
<td>37.5 min.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>30.6 min.</td>
</tr>
</tbody>
</table>

- **Conclusion:** IntelliGenWiki was indeed efficient and reduced the paper selection and curation time by almost 70% and 20%, respectively.
Our Contributions

- Design of a cohesive Wiki-NLP integration architecture
- Extensible for other wiki engines
- Allows use of existing text mining techniques in your wiki
- Create machine-accessible information
- Performed the first extrinsic evaluation of an NLP integration within wikis
- Add another party to the wiki community: AI
- The groundwork for a multitude of new projects
What you can do now

- Add NLP capabilities to your wiki for a variety of use cases
  - Find scenarios in which NLP assistance can be useful, e.g., Summarization
  - Develop the actual NLP pipelines, e.g., based on GATE\(^1\)
  - Deploy the pipelines on a Semantic Assistants server
  - Alternatively, use the existing text mining services in our public server

- Download and deploy the Wiki-NLP integration
  - Deploy the Wiki-NLP servlet on a container, e.g., Tomcat or Jetty
  - Install the Semantic Assistants MediaWiki extension on your wiki
  - Configure the extension to point to the servlet endpoint

---

\(^1\) General Architecture for Text Engineering, [http://www.gate.ac.uk](http://www.gate.ac.uk)
Related Publications

B. Sateli, E. Angius, S. S. Rajivelu, and R. Witte.
Can Text Mining Assistants Help to Improve Requirements Specifications?
In Mining Unstructured Data (MUD 2012), Kingston, Ontario, Canada, October 17 2012.

B. Sateli, M.-J. Meurs, G. Butler, J. Powlowski, A. Tsang, and R. Witte.
IntelliGenWiki: An Intelligent Semantic Wiki for Life Sciences.

B. Sateli and R. Witte.
Natural Language Processing for MediaWiki: The Semantic Assistants Approach.
In The 8th International Symposium on Wikis and Open Collaboration (WikiSym 2012), Linz, Austria, 08/2012 2012. ACM.

R. Witte and T. Gitzinger.
Semantic Assistants – User-Centric Natural Language Processing Services for Desktop Clients.

Software Download and More Information

http://www.semanticsoftware.info