ICIC Express Letters
Volume **10**, Number **7**, July **2016**

ICIC International ©2016 ISSN 1881-803X
pp. 1787–1794

RESEARCH AND DEVELOPMENT OF SEMANTIC ANNOTATION PLATFORM FOR SCIENTIFIC LITERATURE

Yao Liu[1,2], Ziyuan Zhang[3] and Yi Huang[1]

[1]Institute of Scientific and Technical Information of China
No. 15, Fuxing Road, Beijing 100038, P. R. China
liuy@istic.ac.cn

[2]Beijing Key Laboratory of Internet Culture and Digital Dissemination Research
No. 35, North Fourth Ring Road, Beijing 100101, P. R. China

[3]Department of Language Information Engineering
Peking University
No. 5, Yiheyuan Road, Beijing 100081, P. R. China

Received December 2015; accepted March 2016

Abstract. *A semantic annotation platform can effectively organize digital resources and extract knowledge fragments from structured and instructed text. The existing semantic annotation platforms are facing with a great challenge, namely, the difficulty to adapt and reuse. At present, some of the existing semantic annotation platforms in the general domain demonstrate that they have high performance and high analytical accuracy. However, a domain oriented semantic annotation platform is still in the initial stage, especially in scientific literature. Additionally, scientific literature processing tasks are becoming increasingly diverse, so how to quickly solve these issues becomes a serious problem. In view of the cases above, this paper develops a semantic annotation platform for scientific literature. Using this platform to test and compare the processing components for scientific literature can effectively complete the comparative testing work. For researchers focusing on scientific literature, it is a testing platform to compare the performance of components. All of these will help to promote the popularization and application of semantic annotation platform for scientific literature.*

Keywords: Scientific literature, Semantic annotation platform, Customization

1. Introduction.

With the rapid development of computer science and Internet technology, we are confronted with the information overload problem rather than the problem of lack of information. In particular, the number of scientific literature increases every year, which makes it difficult for researchers to keep up with the growth rate of information. The advent of digital library is in response to such an overload, to make it easier to retrieve the scientific literature for us. However, the researchers still need to spend a lot of time and efforts finding documents and information they really need.

Semantic annotation platform can effectively organize digital resources and extract knowledge from domain structured text and Web unstructured text. Currently, some of the existing semantic annotation platforms show high performance and high analytical accuracy in general fields. A big challenge the existing semantic annotation platforms face is that they are often difficult to adapt and reuse. In addition, scientific literature processing task is becoming increasingly diverse and pluralistic. In order to effectively solve these problems, this paper conducts an in-depth study, and develops a semantic annotation platform for scientific literature.

In recent years, various methods on semantic annotation have been introduced. Carr et al. [1] described the attempts of the COHSE project to define and deploy a Conceptual Open Hypermedia Service to conduct semantic annotation. Kahan et al. [2] described a

Web-based shared annotation system based on a general-purpose open resource description framework (RDF) infrastructure to annotate webpages. Handschuh et al. [3] provided an S-CREAM framework that allows for creation of metadata for Web documents through semantic annotation. Vargas-Vera et al. [4] presented MnM, an annotation tool which provides both automated and semi-automated support for annotating Web pages with semantic contents. Liu et al. [5] described a method to use domain ontology to annotate professional literature. There are two main types according to the tagging methods used.

(1) Pattern-based semantic annotation platform: capable of performing pattern discovery, and a user can define pattern or model manually. Most pattern discovery methods followed the basic approach Brin [6] outlined. Defining an initial set of entities, scanning the corpus to find models covering the entities, with the discovery of new models, and then new entities are founded. This process is continuous recursively until no new entity is found or the user terminates.

(2) Machine learning-based semantic annotation platform: to use statistical probability model or induction model to predict the position of entities in documents. DATAMOLD algorithm used hidden Markov model to find the entity data in Web documents. Armadillo and Ont-O-Mat used Amilcare toolkit to conduct wrapper induction-based machine learning [7-9].

However, those semantic annotation platforms did not perform well in scientific literature, and most of them did not have the ability to process Chinese documents. Thus, a semantic annotation platform for scientific literature (SAPSL), with flexible architecture to customize processing tasks towards a user's different needs, is urgently needed.

2. **Framework.**

Based on the analysis above, open source software and natural language processing technology are used to form the framework of the platform. The basic process is shown as Figure 1.

The platform consists of three modules:

(1) File manager: a database that stores the text information and a database schema based on an object-oriented model;

(2) A graphic user interface: to start data processing, view and evaluate results;

(3) CREOLE: a collection of reusable objects for language engineering.

3. **Key Technologies.**

Here are the key technologies we use to construct SAPSL.

3.1. **Interface customization.**

Based on the GUI of Gate Developer, we draw the GUI of SAPSL. The interface provides a convenient graphical environment to develop required functions accordingly. The main tasks of SAPSL are to annotate scientific literature, and its core modules are as the following:

(1) Language components: composed by the documents and corpus;

(2) Processing components: common operations to processing documents;

(3) Applications: a sequence processing resources can be applied to a document or corpus.

Therefore, an in-depth analysis on Gate Developer source code is conducted to prepare for GUI customization. The source file directory structure is shown as Figure 2.

In addition to the directory structure, the structure of src/main, the core part of the Gate Developer, is shown as Figure 3.

By analyzing the source code, we found that MainFrame.java in the GUI package provides the interface elements of the platform: language and icons. There are several methods to achieve localization and icon customization. With *static public Icon getIcon (String baseName)*, we can easily customize icon for SAPSL. However, localization is relatively complicated. Localization for menus, buttons, and logs call for different methods, for instance, *resourcesTreeRoot = new DefaultMutableTreeNode ("首页", true)*, *button.setText*

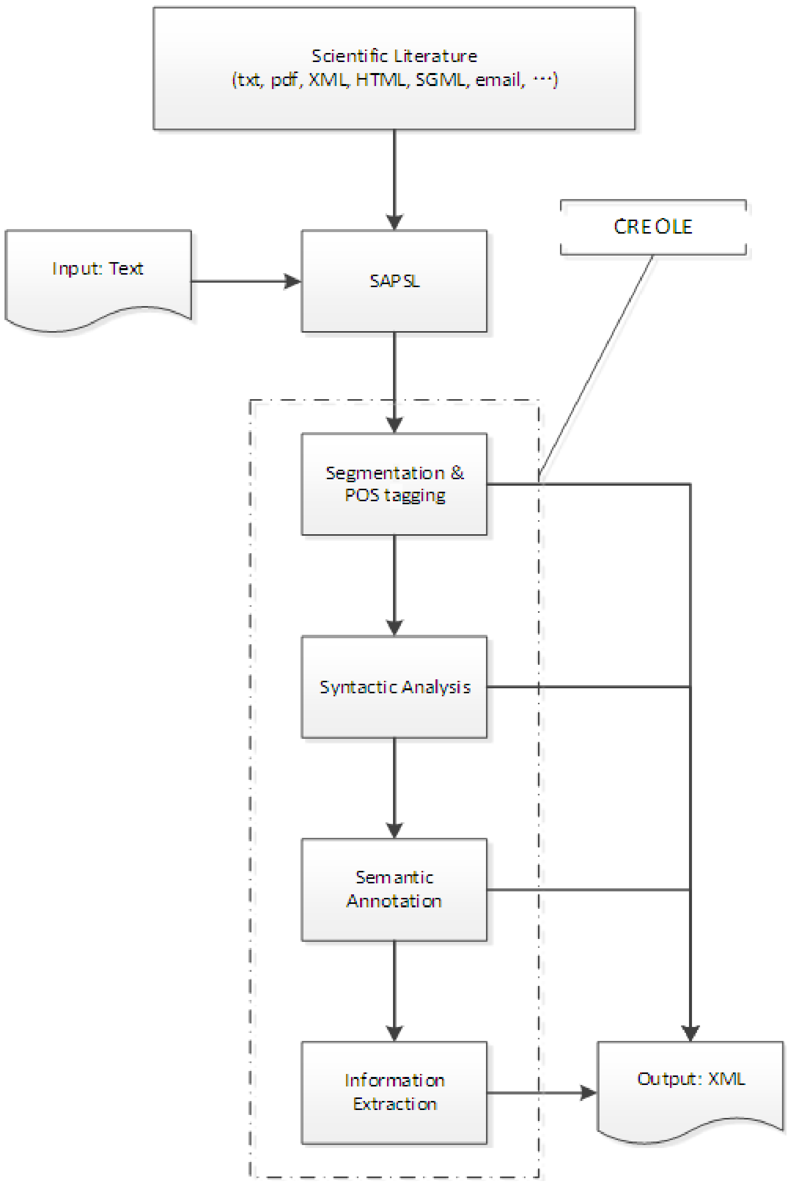


Figure 1. The basic process of SAPSL

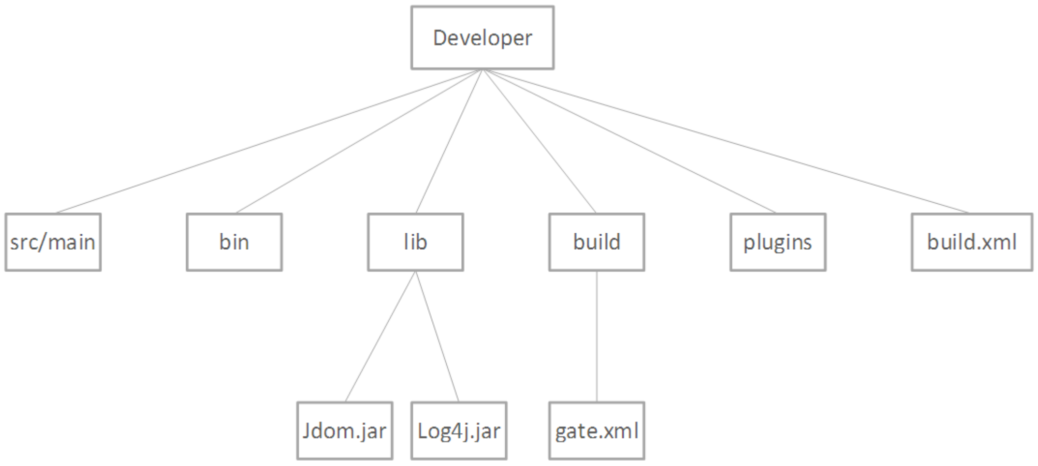


Figure 2. The directory structure of gate developer

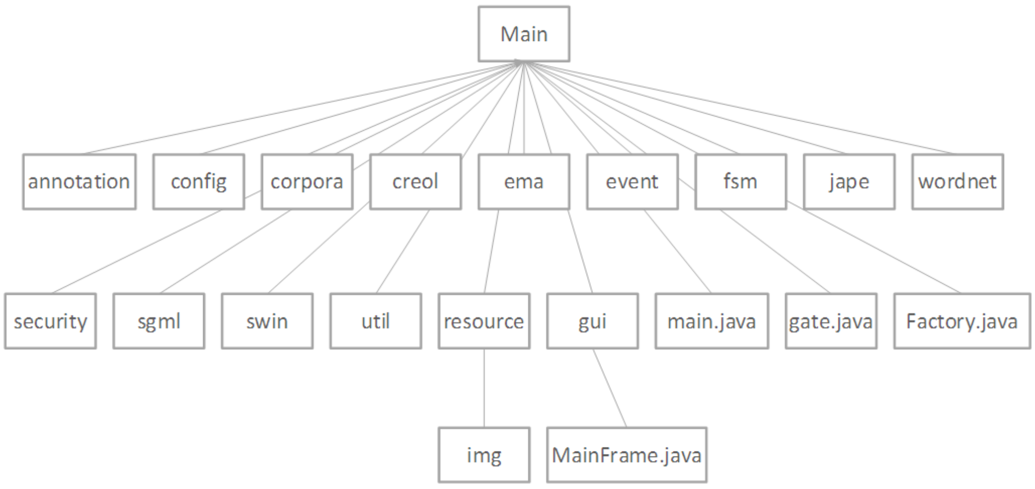


FIGURE 3. The structure of src/main

(), Out.prln (String x). Through these methods, a customized Chinese GUI for SAPSL can be generated.

3.2. **Plug-in development.** In most cases, to use a processing component and language component, a user must first load the plug-in that contains those resources. Definition of these resources are stored in the CREOLE directory, i.e., an XML file describing all the plug-ins and resources.

Plug-ins can be loaded from the following sources:

(1) Core plug-in: plug-in installed in the default directory;

(2) User plug-in: plug-in in the personal plugin directory;

(3) Native plug-in: plug-in stored in the disk, but not in the core plug-in or user plugin directory;

(4) Remote plugin: plug-in that can be loaded from a remote machine via http protocol.

To create a user's resource components or modify existing components JAVA code, we have to go through the work in a life cycle, which includes:

(1) Instantiate a resource component: use *Factory* class, which includes the parameter of resource component, recovery data, etc., to create a resource component;

(2) Import the resource component into development environment: use "New Resource component" under the "File" menu;

(3) Resource component configuration and execution: to edit creole.xml or modify part of the JAVA code that executes the resource component to build a jar file.

3.3. **Embedding.** JNI is a programming framework that enables Java code running in a JVM to call and be called by native applications and libraries written in other languages such as C, C++ and assembly. When an application is not entirely written by JAVA, JNI is often introduced to handle this situation. For example, the standard JAVA library does not support some specific features or libraries needed by SAPSL. To be more specific, we need to embed a segmentation application written in C++ into SAPSL to get a much better performance and result. The process that SAPSL calls ICTCLAS to execute POS tagging function through JNI is shown as follows:

(1) Create a class (MyICTCLAS.java) to declare the native method;

(2) Use javac to compile the source code to generate class file;

(3) Use javah -jni to generate C/C++ header file;

(4) Code the local method in C/C++;

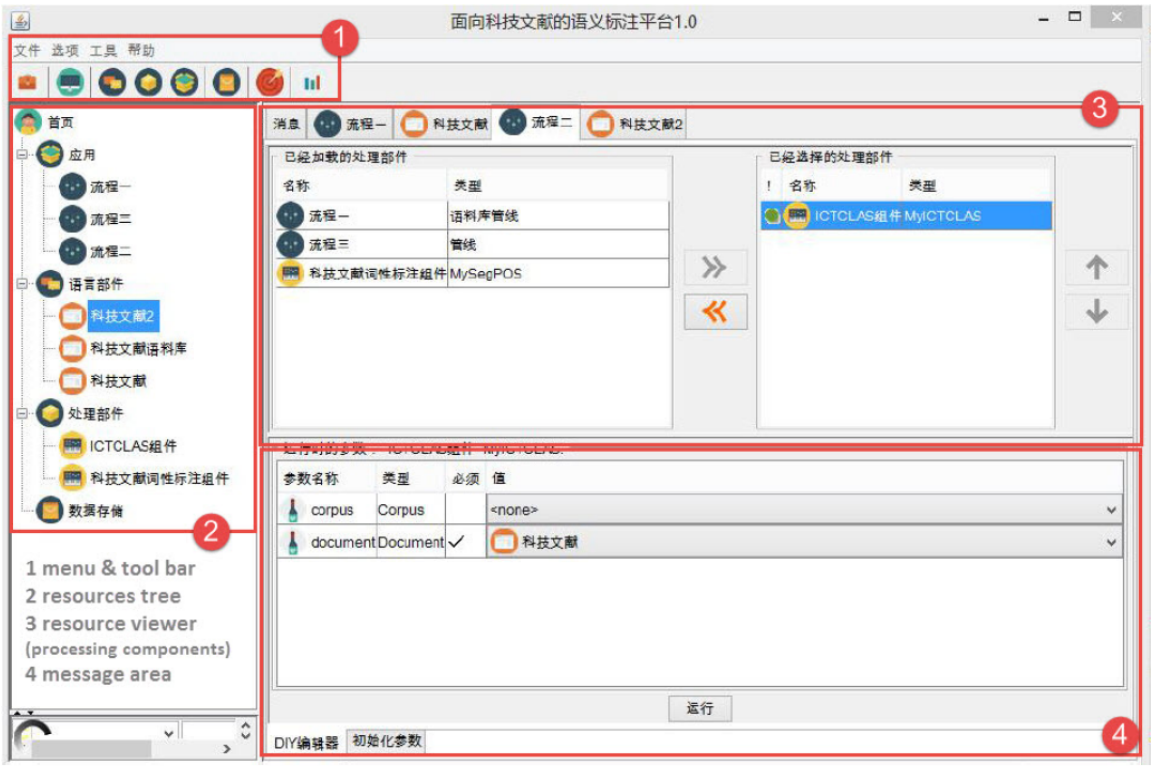
(5) Compile C/C++ to the local library, and create a dynamic link library file;

(6) Run the program.

4. Experiments and Analysis.

4.1. Setup. The experiment environment included an Intel Core (TM) 2 Duo P8600 (2.4 GHz) processor with 4.00 G RAM and Windows 8 operating system.

4.2. Function test. In the process of segmentation and POS tagging in natural language processing, two methods were chosen to complete the task. The first method loaded scientific literature cutting segmentation and POS tagging processing component, with segmentation model "skin_me_pfr6000_seg.model" and POS tagging model "skin_me_pfr6000_tag.model"; the second method loaded ICTCLAS processing component, with PKU level mark as the tag set. All the tasks were processing the same corpus and documents. The results are shown as Figure 4 and Figure 5.



名称	类型
流程一	语料库管线
流程三	管线
科技文献词性标注組件	MySegPOS

!	名称	类型
	ICTCLAS組件	MyICTCLAS

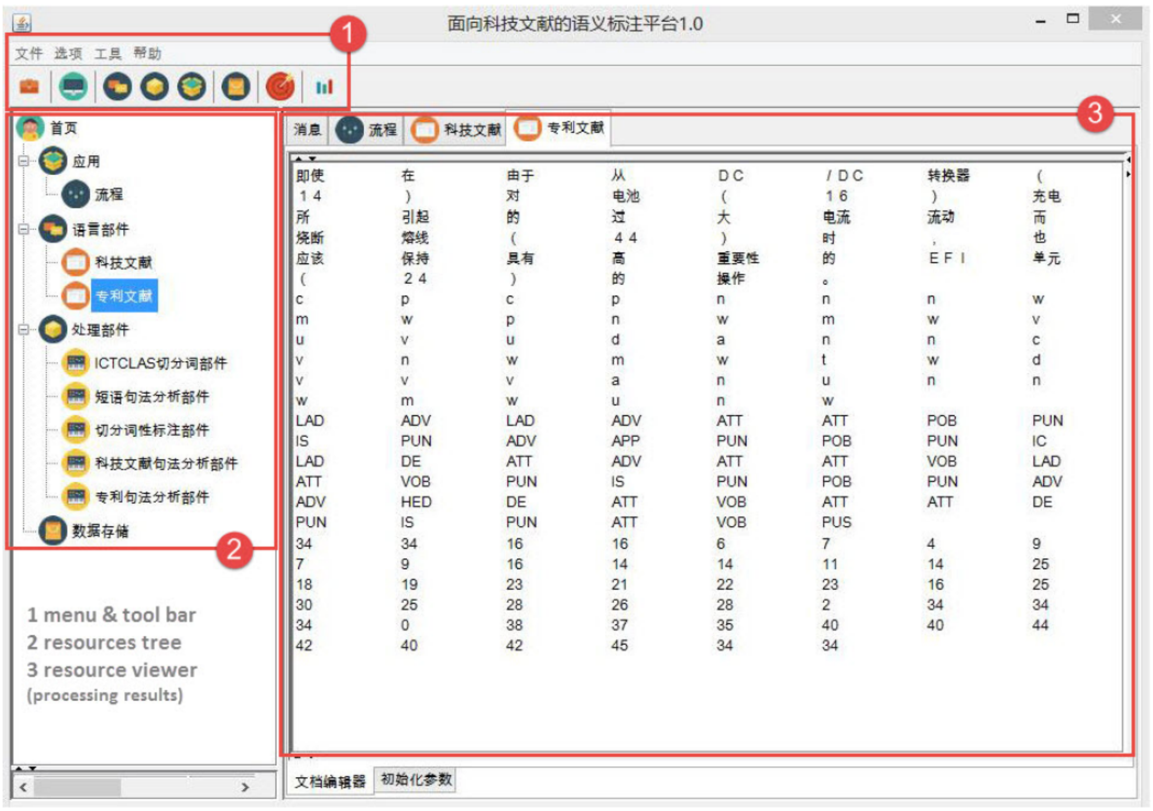
参数名称	类型	必须	值
corpus	Corpus		<none>
document	Document	✓	科技文献

FIGURE 4. The graphical user interface of SAPSL

As can been seen from Figure 4, a user can choose some more reasonable processing components according to his own needs; for example, he can just choose the ICTCLAS component to cut sentence into words, or he can choose those components needed to complete the information extraction process. And Figure 5 shows the results of choosing both word segmentation and POS tagging processing components for a Chinese patent text.

4.3. Performance comparison. When processing scientific literature, a user can choose different processing components to compare the pros and cons in SAPSL. MSTParser and MaltParser are chosen to complete the syntactic analysis task. We selected a patent corpus with 2000 sentences (1500 for training, 500 for testing). ACT, ADV, PUS, VOB, VV, LA, UAS and LAS are the main index to evaluate the results of the experiment. The results are shown in Table 1.

As can been seen from Table 1, ADV demonstrates a very high accuracy rate, recall and F-measure while HED, VOB also make an acceptable performance. Notably, the VV scores in both MSTParser and MaltParser are significantly lower. Overall, MSTParser had a higher performance than MaltParser in most of the cases. However, MaltParser



```
即使    在      由于    从      ＤＣ     ／ＤＣ   转换器   （
１４    ）      对      电池    （       １６     ）       充电
所      引起    的      过      大       电流     流动     而
烧断    熔线    （      ４４    ）       时       ，       也
应该    保持    具有    高      重要性   的       ＥＦＩ   单元
（      ２４    ）      的      操作     。
c       p       c       p       n        n        n        w
m       w       p       n       w        m        w        v
u       v       u       d       a        n        n        c
v       n       w       m       w        t        w        d
v       v       v       a       n        u        n        n
w       m       w       u       n        w
LAD     ADV     LAD     ADV     ATT      ATT      POB      PUN
IS      PUN     ADV     APP     PUN      POB      PUN      IC
LAD     DE      ATT     ADV     ATT      ATT      VOB      LAD
ATT     VOB     PUN     IS      PUN      POB      PUN      ADV
ADV     HED     DE      ATT     VOB      ATT      ATT      DE
PUN     IS      PUN     ATT     VOB      PUS
34      34      16      16      6        7        4        9
7       9       16      14      14       11       14       25
18      19      23      21      22       23       16       25
30      25      28      26      28       2        34       34
34      0       38      37      35       40       40       44
42      40      42      45      34       34
```

FIGURE 5. The results of word segmentation and POS tagging

TABLE 1. The results of syntactic analysis of patent corpus

Index	MSTParser Component			MaltParser Component		
	P	R	F1	P	R	F1
ACT	0.506	0.568	0.535	0.307	**0.622**	0.411
ADV	0.933	0.884	0.908	**0.943**	0.857	0.898
PUS	0.668	0.368	0.475	**0.805**	**0.496**	**0.614**
HED	0.708	0.705	0.707	0.571	0.419	0.483
VOB	0.746	0.758	0.752	0.671	0.721	0.695
VV	0.312	0.447	0.368	**0.373**	0.408	**0.390**
LAS	0.752			0.729		
LA	0.857			0.826		
UAS	0.800			0.778		

had a high rate in PUS. Thus, a researcher with little technical background can easily use SAPSL to make a performance comparison between different components with the same function.

4.4. **Annotation and extraction.** This paper also took patent as a general scientific literature to go through a full information extraction process, i.e., word segmentation, POS tagging, syntactic analysis, semantic annotation and extraction. According to the characteristic of patent text, we classified verb into four categories as shown in Table 2, to extract type, feature, component and function information respectively of a patent.

In addition to verb, we looked into other clues that connect with the core verb, for instance, preposition sometimes being a key factor to improve extraction accuracy. When analyzing the dependency tree, we can also extract knowledge based on the syntactic structure, as new extraction rules:

Table 2. Verb categories in patent literature

Verb Category	Verbs
Patent Type	belong to (属于), involve (涉及), provide (提供), suit (适合), be (为)
Patent Feature	have (有), possess (具有)
Patent Component	adopt (采用), include (包括), install (安装), set (设置), connect (连接)
Patent Function	achieve (实现), solve (解决), improve (提高), add (增加), promote (促进), facilitate (方便), ensure (保证), eliminate (消除), control (控制), direct (指导), avoid (防止), apply (应用)

Table 3. The results of annotation and extraction of patent literature

Rules	P	R	F1
Patent Type (QUN)	81.57%	96.88%	88.57%
Patent Function (ACT)	66.67%	92.31%	77.42%
Patent Component (ADV)	77.78%	92.45%	84.48%

(1) QUN in a sentence indicates "patent type" information;
(2) A phrase that acts as the object of ACT indicates "patent function" information;
(3) "Patent component" information often appears in a phrase with a "P + N + V" pattern.

We processed 1000 patent abstract data with SAPSL, the results are shown as Table 3.

As can be seen from Table 3, the F value on average of the experiment is 86.49%, which meets the basic requirement of patent knowledge extraction. It can be concluded that the rules are very effective and SPASL is capable of processing scientific literature according to users' needs.

5. Conclusions.

A semantic annotation platform can effectively organize digital resources and extract knowledge fragments from structured and untrusted text. The existing semantic annotation platforms are facing with a great challenge, namely, the difficulty to adapt and reuse. In view of the issues above, this paper develops a semantic annotation platform for scientific literature. With SAPSL, a user can compare the pros and cons of the components for word segmentation, POS tagging, syntactic analysis, and semantic annotation etc. in scientific literature. This paper aims to promote the popularization and application of semantic annotation system for scientific literature. In the future we intend to proceed along two lines in parallel: on one hand, to develop and adopt more processing components for SAPSL; on the other hand, to optimize the processes of processing large corpus of scientific literature.

Acknowledgment. This work is partially supported by the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research (ICDD201502) National Key Project of Scientific and Technical Supporting Programs No. 2013BAH21-B02. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] L. Carr, S. Bechhofer, C. Goble and W. Hall, Conceptual linking: Ontology-based open hypermedia, *Proc. of the 10th International Conference on World Wide Web*, pp.334-342, 2001.
[2] J. Kahan, M.-R. Koivunen, E. Prud'Hommeaux and R. R. Swick, Annotea: An open RDF infrastructure for shared Web annotations, *Computer Networks*, vol.39, no.5, pp.589-608, 2002.

[3] S. Handschuh, S. Staab and F. Ciravegna, S-CREAM – Semi-automatic creation of metadata, *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pp.358-372, 2002.

[4] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt and F. Ciravegna, MnM: Ontology driven semi-automatic and automatic support for semantic markup, *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pp.379-391, 2002.

[5] Y. Liu, H. Shi, D. Zheng and Y. Huang, Study on semantic annotation for professional literature, *ICIC Express Letters, Part B: Applications*, vol.5, no.5, pp.1383-1389, 2014.

[6] S. Brin, Extracting patterns and relations from the world wide web, *The World Wide Web and Databases*, pp.172-183, 1999.

[7] V. Borkar, K. Deshmukh and S. Sarawagi, Automatic segmentation of text into structured records, *ACM SIGMOD Record*, vol.30, no.2, pp.175-186, 2001.

[8] A. Dingli, F. Ciravegna and Y. Wilks, Automatic semantic annotation using unsupervised information extraction and integration, *Proc. of SemAnnot 2003 Workshop*, 2003.

[9] J. Kietz, R. Volz and A. Maedche, Extracting a domain-specific ontology from a corporate intranet, *Proc. of the 2nd workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning*, vol.7, pp.167-175, 2000.