OCR++: A Robust Framework For Information Extraction from Scholarly Articles

Mayank Singh, Barnopriyo Barua, Priyank Palod, *Manvi Garg,
Sidhartha Satapathy, Samuel Bushi, Kumar Ayush, Krishna Sai Rohith, Tulasi Gamidi,
Pawan Goyal and Animesh Mukherjee
Department of Computer Science and Engineering
*Department of Geology and Geophysics
Indian Institute of Technology, Kharagpur, WB, India
mayank.singh@cse.iitkgp.ernet.in

Abstract

This paper proposes *OCR++*, an open-source framework designed for a variety of information extraction tasks from scholarly articles including metadata (title, author names, affiliation and e-mail), structure (section headings and body text, table and figure headings, URLs and footnotes) and bibliography (citation instances and references). We analyze a diverse set of scientific articles written in English to understand generic writing patterns and formulate rules to develop this hybrid framework. Extensive evaluations show that the proposed framework outperforms the existing state-of-the-art tools by a large margin in structural information extraction along with improved performance in metadata and bibliography extraction tasks, both in terms of accuracy (around 50% improvement) and processing time (around 52% improvement). A user experience study conducted with the help of 30 researchers reveals that the researchers found this system to be very helpful. As an additional objective, we discuss two novel use cases including automatically extracting links to public datasets from the proceedings, which would further accelerate the advancement in digital libraries. The result of the framework can be exported as a whole into structured TEI-encoded documents. Our framework is accessible online at *http://www.cnergres.iitkgp.ac.in/OCR++/home/*.

1 Introduction

Obtaining structured data from documents is necessary to support retrieval tasks (Beel et al., 2011). Various scholarly organizations and companies deploy information extraction tools in their production environments. Google scholar[1], Microsoft academic search[2], Researchgate[3], CiteULike[4] etc. provide academic search engine facilities. European publication server (EPO)[5], ResearchGate and Mendeley[6] use *GROBID* (Lopez, 2009) for header extraction and analysis. A similar utility named *SVMHeaderParse* is deployed by $CiteSeerX$[7] for header extraction.

Through a comprehensive literature survey, we find comparatively less research in document structure analysis than metadata and bibliography extraction from scientific documents. The main challenges lie in the inherent errors in OCR processing and diverse formatting styles adopted by different publishing venues. We believe that a key strategy to tackle this problem is to analyze research articles from different publishers to identify generic patterns and rules, specific to various information extraction tasks. We introduce *OCR++*, a hybrid framework to extract textual information such as (i) metadata – title, author names, affiliation and e-mail, (ii) structure – section headings and body text, table and figure headings, URLs and footnotes, and (iii) bibliography – citation instances and references from scholarly articles.

[1] http://scholar.google.com
[2] http://academic.research.microsoft.com
[3] https://www.researchgate.net
[4] http://www.citeulike.org/
[5] https://data.epo.org/publication-server
[6] https://www.mendeley.com
[7] http://citeseerx.ist.psu.edu/

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 3390–3400, Osaka, Japan, December 11-17 2016.

The framework employs a variety of Conditional Random Field (CRF) models and hand-written rules specially crafted for handling various tasks. Our framework produces comparative results in metadata extraction tasks. However, it significantly outperforms state-of-the-art systems in structural information extraction tasks. On an average, we record an accuracy improvement of 50% and a processing time improvement of 52%. We claim that our hybrid approach leads to higher performance than complex machine learning models based systems. We also present two novel use cases including extraction of public dataset links available in the proceedings of the NLP conferences available from the ACL anthology.

2 Framework overview

OCR++ is an extraction framework for scholarly articles, completely written in Python (Figure 2). The framework takes a PDF article as input, 1) converts the PDF file to an XML format, 2) processes the XML file to extract useful information, and 3) exports output in structured TEI-encoded[8] documents. We use open source tool *pdf2xml*[9] to convert PDF files into rich XML files. Each token in the PDF file is annotated with rich metadata, namely, x and y co-ordinates, font size, font weight, font style etc. (Figure 1).

```
<?xml version="1.0" encoding="UTF-8"?>
<DOCUMENT><METADATA><PDFFILENAME>/var/www/html/OCR++/myproject/media/documents/input.pdf</PDFFILENAME><PROCESS name="pdftoxml" cmd="
-noImage -noImageInline "><VERSION value="1.2"><COMMENT/></VERSION><CREATIONDATE>Fri Oct  7 07:11:18 2016
</CREATIONDATE>
</PROCESS>
    </METADATA>
    <PAGE width="595.276" height="841.89" number="1" id="p1">
        <MEDIABOX x1="0" y1="0" x2="595.276" y2="841.89"/>
        <CROPBOX x1="0" y1="0" x2="595.276" y2="841.89"/>
        <BLEEDBOX x1="0" y1="0" x2="595.276" y2="841.89"/>
        <ARTBOX x1="0" y1="0" x2="595.276" y2="841.89"/>
        <TRIMBOX x1="0" y1="0" x2="595.276" y2="841.89"/>
        <TEXT width="397.117" height="12.8972" id="p1_t1" x="100.215" y="65.4231">
            <TOKEN sid="p1_s3" id="p1_w1" font-name="NimbusRomNo9L" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="100.215" y="65.4231" base="75.322" width="53.0092" height="12.8972">
            OCR++:</TOKEN>
            <TOKEN sid="p1_s4" id="p1_w2" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="157.672" y="65.4231" base="75.322" width="10.358" height="12.8972">A</
            TOKEN>
            <TOKEN sid="p1_s5" id="p1_w3" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="171.616" y="65.4231" base="75.322" width="43.5551" height="12.8972">
            Robust</TOKEN>
            <TOKEN sid="p1_s6" id="p1_w4" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="218.758" y="65.4231" base="75.322" width="72.3622" height="12.8972">
            Framework</TOKEN>
            <TOKEN sid="p1_s7" id="p1_w5" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="294.706" y="65.4231" base="75.322" width="21.9497" height="12.8972">
            For</TOKEN>
            <TOKEN sid="p1_s8" id="p1_w6" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="320.243" y="65.4231" base="75.322" width="74.5572" height="12.8972">
            Information</TOKEN>
            <TOKEN sid="p1_s9" id="p1_w7" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="398.386" y="65.4231" base="75.322" width="65.3469" height="12.8972">
            Extraction</TOKEN>
            <TOKEN sid="p1_s10" id="p1_w8" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="467.32" y="65.4231" base="75.322" width="30.0123" height="12.8972">
            from</TOKEN>
            </TEXT><TEXT width="109.576" height="12.8972" id="p1_t2" x="243.984" y="81.3631"><TOKEN sid="p1_s11" id="p1_w9" font-name="
            nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="14.3462" font-color="#000000" rotation="0"
            angle="0" x="243.984" y="81.3631" base="91.262" width="58.1882" height="12.8972">Scholarly</TOKEN>
            <TOKEN sid="p1_s12" id="p1_w10" font-name="nimbusromno9l" symbolic="yes" fixed-width="yes" bold="no" italic="no" font-size="
            14.3462" font-color="#000000" rotation="0" angle="0" x="305.759" y="81.3631" base="91.262" width="47.8015" height="12.8972">
            Articles</TOKEN>
```


Figure 1: Screenshot of *pdf2xml* tool output for the current paper.

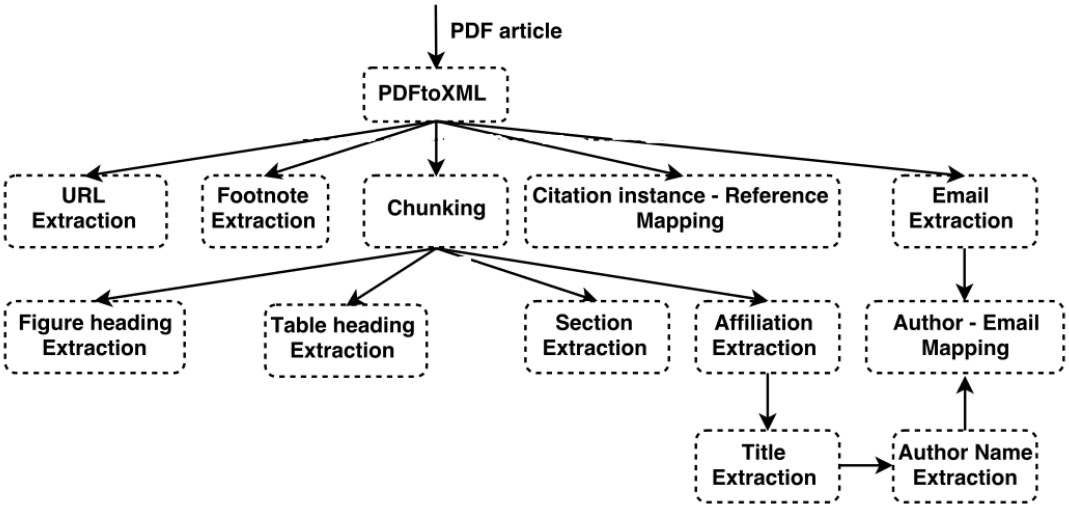
Figure 2(a) describes the sub-task dependencies in *OCR++*. The web interface of the tool is shown in Figure 2(b). We leverage the rich information present in the XML files to perform extraction tasks. Although each extraction task described below is performed using machine learning models as well as hand written rules/heuristics, we only include the better performing scheme in our framework. Next, we describe each extraction task in detail.

2.1 Chunking

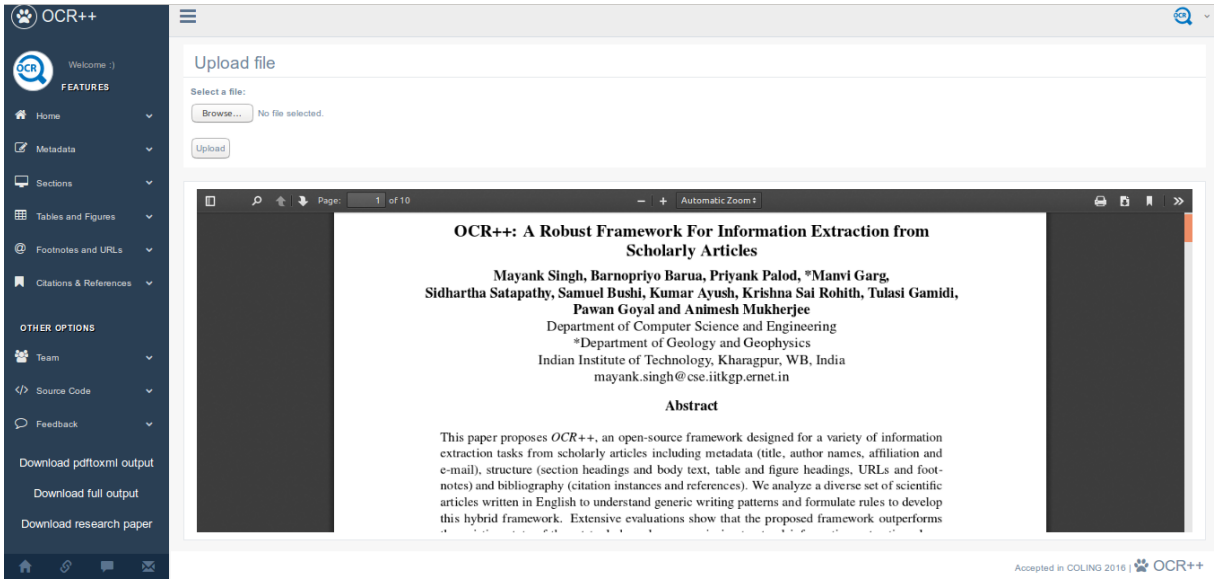
As a first step, we segment XML text into *chunks* by measuring distance from neighboring text and differentiating from the surrounding text properties such as font-size and bold-text.

[8] http://www.tei-c.org/index.xml

[9] URL: http://sourceforge.net/projects/pdf2xml/. We employ version 1.2.7 developed for 64 bit Linux systems.



(a) Sub-tasks dependencies in *OCR++*.



(b) Screenshot of OCR++ web interface.

Figure 2: OCR++ framework overview and user interface

2.2 Title extraction

We train a CRF model to label the token sequences using 6300 training instances. Features are constructed based on generic characteristics of formatting styles. Token level feature set includes boldness, relative position in the paper, relative position in the first chunk, relative size, case of first character, boldness + relative font size overall, case of first character in present and next token and case of first character in present and previous token.

2.3 Author name extraction

In this sub-task, we use the same set of features as described in the title extraction sub-task to train the CRF model along with a heuristic that the tokens eligible for the author names are either present in the first section or within 120 tokens after the title. Different author names are distinguished using heuristics, such as, difference in y-coordinates, tab separation etc. Further, false positives are removed using heuristics such as length of consecutive author name tokens, symbol or digit in token and POS tag. The first word among consecutive tokens is considered as the first name, the last word as the last name, and all the remaining words are treated together as the middle name.

2.4 Author e-mail extraction

An e-mail consists of a user name, a sub-domain name and a domain name. In case of scholarly articles, usually, the user names are written inside brackets separated by commas and the bracket is succeeded by the sub-domain and domain name. On manual analysis of a set of scholarly articles, we find four different writing patterns, author1@cse.domain.com, {author1, author2, author3}@cse.domain.com, [author4, author5, author6]@cse.domain.com and [author7@cse, author7@ee].domain.com. Based on these observations, we construct hand written rules to extract e-mails.

2.5 Author affiliation extraction

We use hand written rules to extract affiliations. We employ heuristics such as presence of country name, tokens like "University", "Research", "Laboratories", "Corporation", "College", "Institute", superscript character etc.

2.6 Headings and section mapping

We employ CRF model to label section headings. Differentiating features (the first token of the chunk, the second token, avg. boldness of the chunk, avg. font-size, Arabic/Roman/alpha-enumeration etc.) are extracted from chunks to train the CRF.

2.7 URL

We extract URLs using a single regular expression described below:

```
http[s]?://(?:[a-zA-Z]|[0-9]|[\$-_@.&+]|[!*\(\),]|(?:\%[0-9a-fA-F][0-9a-fA-F]))
```

2.8 Footnote

Most of the footnotes have numbers or special symbols (like asterisk etc.) at the beginning in the form of a superscript. Footnotes have font-size smaller than the surrounding text and are found at the bottom of a page – average font size of tokens in a chunk and y-coordinate were used as features for training the CRF. Moreover, footnotes are found in the lower half of the page (this heuristic helped in filtering false positives).

2.9 Figure and table headings

Figure and table heading extraction is performed after chunking (described in Section 2.1). If the chunk starts with the word "FIGURE" or "Figure" or "FIG." or "Fig.", then the chunk represents a figure heading. Similarly, if the chunk starts with the word "Table" or "TABLE", then the chunk represents a table heading. However, it has been observed that table contents are also present in the chunk. Therefore, we use a feature "bold font" to extract bold tokens from such chunks.

2.10 Citations and references

The bibliography extraction task includes extraction of citation instances and references. All the tokens succeeding the reference section are considered to be part of references and further each reference is extracted separately. Again, we employ hand written rules to distinguish between two consecutive references. On manual analysis, we found 16 unique citation instance writing styles (Table 1). We code these styles into regular expressions to extract citation instances.

2.11 Mapping tasks

Connecting author name to e-mail: In general, each author name present in the author section associates with some e-mail. *OCR++* tries to recover this association using simple rules, for example, sub-string match between username and author names, abbreviated full name as username, order of occurrence of e-mails etc.

Citation reference mapping: Each extracted citation instance is mapped to its respective reference. Since, there are two different styles of writing citation instances, *Indexed* and *Non-indexed*, we define mapping tasks for each style separately. Indexed citations are mapped directly to references with the

Table 1: Generic set of regular expressions for citation instance identification. Here, AN represents author name, Y represents year and I represent reference index within citation instance.

Citation Format	Regular Expression
<AN> et al. [<I>]	`([A-Z][a-zA-Z]* et al[.][\string\d\{1,3\}])`
<AN> [<I>]	`([A-Z][a-zA-Z]* [\string\d\{2\}])`
<AN> et al.<spaces> [<I>]	`([A-Z][a-zA-Z]* et al[.][ ]*[\string\d\{1\}])`
<AN> et al., <Y><I>	`([A-Z][a-zA-Z]* et al[.],\string\d\{4\}[a-z])`
<AN> et al., <Y>	`([A-Z][a-zA-Z]* et al[.][,] \string\d\{4\})`
<AN> et al., (<Y>)	`([A-Z][a-zA-Z]* et al[.][,] (\string\d\{4\}))`
<AN> et al. <Y>	`([A-Z][a-zA-Z]* et al[.] \string\d\{4\})`
<AN> et al. (<Y>)	`([A-Z][a-zA-Z]* et al[.] (\string\d\{4\}))`
<AN> and <AN> (<Y>)	`([A-Z][a-zA-Z]* and [A-Z][a-zA-Z]*(\string\d\&{4\}))`
<AN> & <AN> (<Y>)	`([A-Z][a-zA-Z]* & [A-Z][a-zA-Z]*(\string\d\&{4\}))`
<AN> and <AN>, <Y>	`([A-Z][a-zA-Z]* and [A-Z][a-zA-Z]*[,] \d\{4\})`
<AN> & <AN>, <Y>	`([A-Z][a-zA-Z]* & [A-Z][a-zA-Z]*[,] \d\{4\})`
<AN>, <Y>	`([A-Z][a-zA-Z]*[,] \string\d\{4\})`
<AN> <Y>	`([A-Z][a-zA-Z]* \string\d\{4\})`
<AN>, (<Y><I>)	`([A-Z][a-zA-Z]*(\string\d\{4\}[a-z]*))`
< multiple indices separated by commas >	`.*?[(.*?)]`

index inside enclosed brackets. The extracted index is mapped with the corresponding reference. Non-indexed citations are represented using the combination of year of publication and author's last name.

3 Results and discussion

Following an evaluation carried out by Lipinski et al. (2013), GROBID provided the best results over seven existing systems, with several metadata recognized with over 90% precision and recall. Therefore, we compare *OCR++* with the state-of-the-art *GROBID*. We compare results for each of the sub-tasks for both the systems against the ground-truth dataset. The ground-truth dataset is prepared by manual annotation of title, author names, affiliations, URLs, sections, subsections, section headings, table headings, figure headings and references for 138 articles from different publishers. The publisher names are present in Table 3. We divide the article set into training and test datasets in the ratio of 20:80. Note that each of the extraction modules described in the previous section also has separate training sample count, for instance, 6300 samples have been used to train the title extraction. Also, we observe that both the systems provide partial results in some cases. For example, in some cases, only half of the title is extracted or the author names are incomplete. In order to accommodate partial results from extraction tasks, we provide evaluation results at token level, i.e, what fraction of the tokens are correctly retrieved.

Table 2: Micro-average F-score for GROBID and OCR++ for different extractive subtasks.

	GROBID			OCR++		
Subtask	Precision	Recall	F-Score	Precision	Recall	F-Score
Title	0.93	**0.94**	**0.93**	**0.96**	0.85	0.90
Author First Name	0.81	**0.81**	**0.81**	**0.91**	0.65	0.76
Author Middle Name	N/A	N/A	N/A	**1.0**	**0.38**	**0.55**
Author Last Name	0.83	**0.82**	**0.83**	**0.91**	0.65	0.76
Email	0.80	0.20	0.33	**0.90**	**0.93**	**0.91**
Affiliation	0.74	0.60	0.66	**0.80**	**0.76**	**0.78**
Section Headings	0.70	**0.87**	**0.78**	**0.80**	0.72	0.76
Figure headings	0.59	0.42	0.49	**0.96**	**0.75**	**0.84**
Table headings	0.77	0.17	0.28	**0.87**	**0.74**	**0.80**
URLs	N/A	N/A	N/A	**1.0**	**0.94**	**0.97**
Footnotes	0.80	0.42	0.55	**0.91**	**0.63**	**0.74**
Author-Email	0.38	0.24	0.29	**0.93**	**0.44**	**0.60**

Table 2 presents comparative results for GROBID and *OCR++*. It shows that in terms of precision,

OCR++ outperforms *GROBID* in all the sub-tasks. Recall is higher for *GROBID* for some of the metadata extraction tasks. In *OCR++*, since title extraction depends on the first extracted chunk from section extraction, the errors in chunk extraction lead to low recall in title extraction. Similar problem results in lower recall in author name extraction. Due to the presence of variety of white space length between author first, middle and last name in various formats, we observe low recall overall in author name extraction subtasks. We also found that in many cases author-emails are quite different from author names resulting in lower recall for author-email extraction subtask. *OCR++* outperforms *GROBID* in majority of the structural information extraction subtasks in terms of both precision and recall. We observe that *GROBID* performs poorly for table heading extraction due to intermingling of table text with heading tokens and unnumbered footnotes. A similar argument holds for the figure heading as well. URL extraction feature is not implemented in *GROBID*, while *OCR++* extracts it very accurately. Similarly, poor extraction of non-indexed footnotes resulted in lower recall for footnote extraction subtask.

Table 3: Micro-average F-score for GROBID and OCR++ for different publishing styles.

		GROBID			OCR++		
Publisher	paper count	Precision	Recall	F-Score	Precision	Recall	F-Score
IEEE	30	0.82	0.61	0.70	**0.9**	**0.69**	**0.78**
ARXIV	25	0.75	0.63	0.68	**0.91**	**0.73**	**0.81**
ACM	35	0.69	0.49	0.58	**0.89**	**0.71**	**0.79**
ACL	16	0.89	0.59	0.71	**0.91**	**0.79**	**0.85**
SPRINGER	17	0.78	0.6	0.68	**0.85**	**0.63**	**0.72**
CHI	3	0.13	0.20	0.16	**0.5**	**0.36**	**0.42**
ELSEVIER	6	0.58	0.6	0.59	**0.82**	**0.74**	**0.78**
NIPS	3	0.82	0.68	0.74	**0.83**	**0.72**	**0.77**
ICML	1	**0.6**	**0.6**	**0.6**	0.59	0.54	0.56
ICLR	1	0.49	**0.55**	0.52	**0.67**	0.52	**0.59**
JMLR	1	0.58	0.55	0.56	**0.86**	**0.83**	**0.85**

Similarly, Table 3 compares *GROBID* and *OCR++* for different publishing formats. Here the results seem to be quite impressive with *OCR++* outperforming *GROBID* in almost all cases. This demonstrates the effectiveness and robustness of using generic patterns and rules used in building *OCR++*. As our system is more biased towards single (ACL, ARXIV, ICLR, etc.) and double column formats (ACM, ELSEVIER etc.), we observe lower performance on three column formats (CHI). Similarly, non-indexed sections format (SPRINGER, ELSEVIER, etc.) show less performance than indexed sections format (IEEE, ACM, etc.).

Since citation instance annotation demands a significant amount of human labour, we randomly select eight PDF articles from eight different publishers from ground-truth dataset PDFs. Manual annotation produces 328 citation instances. We also annotate references to produce 187 references in total. Table 4 shows performance comparison for bibliography related tasks. As depicted from Table 4, *OCR++* performs better for both citation and reference extraction tasks. GROBID does not provide Citation-Reference mapping, which is an additional feature of *OCR++*.

Table 4: Micro-average accuracy for GROBID and OCR++ bibliography extraction tasks.

	GROBID			OCR++		
	Precision	Recall	F-Score	Precision	Recall	F-Score
Citation	0.93	0.81	0.87	**0.94**	**0.97**	**0.95**
Reference	0.94	0.94	0.94	**0.98**	**0.99**	**0.98**
Citation-Reference	N/A	N/A	N/A	**0.94**	**0.97**	**0.95**

Next, we investigate whether better formatting styles over the years lead to higher precision by the proposed tool. Also, we compare *OCR++* with *GROBID* in terms of processing time.

3.1 Effect of formatting style on precision

We select International Conference on Computational Linguistics (COLING) as a representative example to understand the effect of evolution in formatting styles over the years on the accuracy of the extraction task. We select ten random articles each from six different years of publications. *OCR++* is used to extract the title for each year. Figure 3 presents title extraction accuracy for each year, reaffirming the fact that the recent year publications produce higher extraction accuracy due to better formatting styles and advancement in converters from Word/LaTeX to PDF. We also notice that before the year 2000, ACL anthology assumes that PDFs do not have embedded text, resulting into a lower recall before 2000.

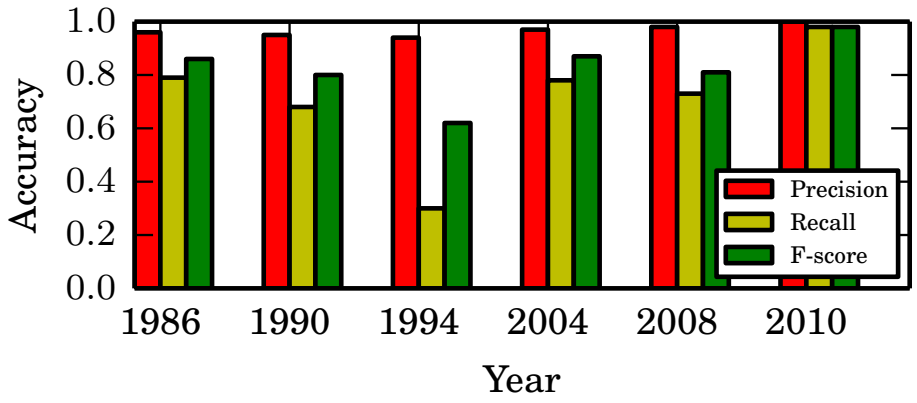


Figure 3: Title extraction accuracy calculated at six different years for COLING.

3.2 Processing time

To compare the processing times, we conducted experiments on a set of 1000 PDFs. The evaluation was performed on a single 64-bit machine, eight core, 2003.0 MHz processor and CentOS 6.5 version. Figure 4 demonstrates comparison between processing time of *GROBID* and *OCR++*, while processing some PDF articles in batch mode. There is significant difference in the execution time of *GROBID* and *OCR++*, with *OCR++* being much faster than *GROBID* for processing a batch of 100 articles.

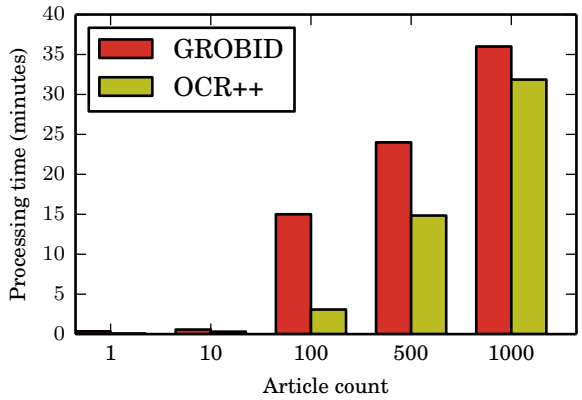


Figure 4: Comparison between batch processing time of *GROBID* and *OCR++*.

3.3 User experience study

To conduct a user experience study, we present *OCR++* to a group of researchers (subjects). Each subject is given two URLs: 1) OCR++ server URL and 2) Google survey form[10]. A subject can upload

[10]http://tinyurl.com/juxq2bt

any research article in PDF format on the server and visualize the output. In the end, the subject has to fill in a response sheet on the Google form. We ask subjects questions related to their experience such as, a) which extraction task did you like the most? b) have you found the system to be really useful? c) have you used similar kind of system before, d) do you find the system slow, fast or moderate, e) comments on the overall system experience and f) drawbacks of the system and suggestions for improvements.

A total of 30 subjects participated in the user experience survey. Among the most liked sub-tasks, title extraction comes first with 50% of votes. Affiliation and author name extraction tasks come second and third respectively. All the subjects found the system to be very useful. Only two of the subjects had used a similar system before. As far as the computational speed is concerned, 50% subjects found the system performance to be fast while 33% felt it to be moderate.

4 Use Cases

4.1 Curation of dataset links

With the community experiencing a push towards reproducibility of results, the links to datasets in the research papers are becoming very informative sources for researchers. Nevertheless, to the best of our knowledge, we do not find any work on automatic curation of dataset links from the conference proceedings. With *OCR++*, we can automatically curate dataset related links present in the articles. In order to investigate this in further detail, we aimed to extract dataset links from the NLP venue proceedings. We ran *OCR++* on four NLP proceedings, ACL 2015, NAACL 2015, ACL 2014, and ECAL 2014, available in PDF format. We extract all the URLs present in the proceedings. We then filter those URLs which are either part of *Datasets* section's body or are present in the footnotes of *Datasets* section, along with the URLs that consist of one of the three tokens: *datasets, data, dumps*. Table 5 presents statistics over these four proceedings for the extraction task. From the dataset links thus obtained, precision was found by human judgement as to whether a retrieved link corresponds to a dataset. One clear trend we saw was the increase in the number of dataset links from year 2014 to 2015. In some cases, the retrieved link corresponds to project pages, tools, researcher's homepage etc. resulting in lowering of precision values.

Table 5: Proceedings dataset extraction statistics: Article count represents total number of articles present in the proceedings. Total links and Dataset links correspond to total number of unique URLs and total number of unique dataset links extracted by OCR++ respectively. Precision measures the fraction of dataset links found to be correct.

Venue	Year	Articles Count	Total links	Dataset links	Precision
ACL	2015	174	345	38	0.74
NAACL	2015	186	186	18	0.50
ACL	2014	139	202	16	0.50
EACL	2014	78	141	12	0.67

4.2 Section-wise citation distribution

Citation instance count plays a very important role in determining future popularity of a research paper. An article's text is distributed among several sections. Some sections have more fraction of citations than the rest. In the second use case, we plan to study the section-wise citation distribution. Section-wise citation distribution refers to how citations are distributed over multiple sections in the article's text. This is an important characteristic of the citations and has recently been used for developing a faceted recommendation system (Chakraborty et al., 2016). We group specific sections to 5 generic sections, Background, Datasets, Method, Result/Evaluation and Discussion/Conclusion. Table 6 shows an example mapping from specific to generic section names. Note that this mapping can be changed as per the requirement. Figure 5 shows citation distribution for article dataset consisting of the 138 articles mentioned earlier. Maximum number of citations are present in the method section, followed by background and discussion and conclusion. Result section comprises the least number of citations.

Table 6: Specific to generic section mapping

Generic section	Specific sections
Background	Introduction, Related Work, Background
Method	Methodology, *Method Specific names*
Result/Evaluation	Results, Evaluation, Metrics
Discussion/Conclusion	Discussion, Conclusion, Acknowledgment

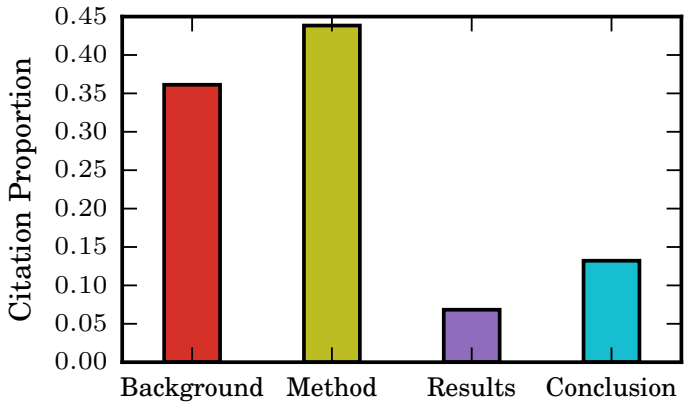


Figure 5: Section-wise citation distribution in article dataset

5 Deployment

The current version of *OCR++* is deployed at our research group server [11]. The present infrastructure consists of single CentOS instance[12]. We make the entire source code publicly available [13]. We also plan to make annotated dataset publicly available in the near future.

6 Related work

Researchers follow diverse approaches for individual extraction tasks. The approaches based on image processing segment document image into several text blocks. Further, each segmented block is classified into a predefined set of logical blocks using machine learning algorithms. Gobbledoc (Nagy et al., 1992) used X-Y tree data structure that converts the two-dimensional page segmentation problem into a series of one-dimensional string-parsing problems. Dengel and Dubiel (1995) employed the concept language of the GTree for logical labeling. Similar work by Esposito et al. (1995) presented a hybrid approach to segment an image by means of a top-down technique and then bottom-up approach to form complex layout component.

Similarly, current state-of-the-art systems use support vector machine (SVM) (Han et al., 2003) and conditional random field (CRF) (Councill et al., 2008; Luong et al., 2012; Lafferty et al., 2001) based machine learning models for information extraction. A study by Granitzer et al. (2012) compares *Parsecit* (a CRF based system) and *Mendeley Desktop*[14] (an SVM based system). They observed that SVMs provide a more reasonable performance in solving the challenge of metadata extraction than CRF based approach. However, Lipinski et al. (2013) observed that *GROBID* (a CRF based system) performed better than *Mendeley Desktop*.

[11] CNeRG. http://www.cnergres.iitkgp.ac.in
[12] OCR++ server. http://www.cnergres.iitkgp.ac.in/OCR++/home/
[13] Source code. http://tinyurl.com/hs9oap2.
[14] https://www.mendeley.com/

7 Conclusions

The goal of this work was to develop an open-source information extraction framework for scientific articles using generic patterns present in various publication formats. In particular, we extract metadata information and section related information. The framework also performs two mapping tasks, author and e-mail mapping and citations to reference mapping. Despite OCR errors and the great difference in the publishing formats, the framework outperforms the state-of-the-art systems by a high margin. We find that the hand-written rules and heuristics produced better results than previously proposed machine learning models.

The current framework has certain limitations. As described in Section 2, we employ *pdf2xml* to convert a PDF article into a rich XML file. Even though the XML file consists of rich metadata, it suffers from common errors generated during PDF conversion. Example of such common errors are the end-of-line hyphenation and character encoding problem. This is a common problem especially in two-column articles. Secondly, the current version of *pdf2xml* lacks character encoding for non English characters. It also suffers from complete omission or in some cases mangling of ligatures, i.e. typical character combinations such as 'fi' and 'fl' that are rendered as a single character in the PDF, which are not properly converted and often lost, for example, resulting in the non-word 'gure' for 'figure'. The three mentioned major problems along with other minor PDF conversion errors are directly reflected in the *OCR++* output.

As discussed in the previous section, a considerable amount of work has already been done to extract reference entities (title, publisher name, date, DOI, etc.) with high accuracy. Therefore, *OCR++* does not aim to extract reference entities. In future, we aim to extend the current framework by extracting information present in figures and tables. Figures and tables present concise statistics about dataset and results. Another possible extension is to add a mapping from unstructured reference to structured reference record. We are also currently in process to extend the functionality for non-English articles.

References

Joeran Beel, Bela Gipp, Stefan Langer, Marcel Genzmehr, Erik Wilde, Andreas Nürnberger, and Jim Pitman. 2011. Introducing Mr. Dlib, a machine-readable digital library. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*, pages 463–464. ACM.

Tanmoy Chakraborty, Amrith Krishna, Mayank Singh, Niloy Ganguly, Pawan Goyal, and Animesh Mukherjee. 2016. FeRoSA: A faceted recommendation system for scientific articles. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 528–541. Springer.

Isaac G. Councill, C. Lee Giles, and Min-Yen Kan. 2008. ParsCit: an open-source CRF reference string parsing package. In *LREC*.

Andreas Dengel and Frank Dubiel. 1995. Clustering and classification of document structure-a machine learning approach. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 2, pages 587–591. IEEE.

Floriana Esposito, Donato Malerba, and Giovanni Semeraro. 1995. A knowledge-based approach to the layout analysis. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 466–471. IEEE.

Michael Granitzer, Maya Hristakeva, Kris Jack, and Robert Knight. 2012. A comparison of metadata extraction techniques for crowdsourced bibliographic metadata management. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 962–964. ACM.

Hui Han, C. Lee Giles, Eren Manavoglu, Hongyuan Zha, Zhenyue Zhang, Edward Fox, et al. 2003. Automatic document metadata extraction using support vector machines. In *Digital Libraries, 2003. Proceedings. 2003 Joint Conference on*, pages 37–48. IEEE.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Mario Lipinski, Kevin Yao, Corinna Breitinger, Joeran Beel, and Bela Gipp. 2013. Evaluation of header metadata extraction approaches and tools for scientific PDF documents. In *Proceedings of the 13th ACM/IEEE-CS joint conference on Digital libraries*, pages 385–386. ACM.

Patrice Lopez. 2009. GROBID: Combining automatic bibliographic data recognition and term extraction for scholarship publications. In *Research and Advanced Technology for Digital Libraries*, pages 473–474. Springer.

Minh-Thang Luong, Thuy Dung Nguyen, and Min-Yen Kan. 2012. Logical structure recovery in scholarly articles with rich document features. *Multimedia Storage and Retrieval Innovations for Digital Library Systems*, page 270.

George Nagy, Sharad Seth, and Mahesh Viswanathan. 1992. A prototype document image analysis system for technical journals. *Computer*, 25(7):10–22, July.