The Pennsylvania State University
The Graduate School

RECORD LINKAGE BETWEEN CITESEERX AND SCHOLARLY BIG DATASETS

A Thesis in
Computer Science and Engineering
by
Athar Sefid

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

December 2019

The thesis of Athar Sefid was reviewed and approved* by the following:

Clyde Lee Giles
Professor of Computer Science
Thesis Co-Advisor

Prasenjit Mitra
Professor of Information Science
Thesis Co-Advisor

Jesse Barlow
Professor of Computer Science

Chitaranjan Das
Professor of Computer Science
Department Head

*Signatures are on file in the Graduate School.

Abstract

CiteSeerX is a public search engine and digital library for scientific papers in the field of computer science. This crawl based digital library harvests scientific documents from internet. Indexing and retrieval of this scholarly big data require correct bibliographic metadata to empower the search engine. Unfortunately, automatic metadata extractors are often incomplete or with errors. These noisy metadata could be cleaned using reference data sets with correct manual input metadata. The cleaning process needs a linking module to match entities between noisy dataset and the correct reference dataset. In this work, we develop a system using supervised machine learning and information retrieval models to match entities in a target database to reference databases, which can further be used to clean metadata in the target database. The supervised model employs hand crafted features extracted from both headers and citations of a scientific paper. The header matching module has a high score of $F1 = 0.91$. The integrated model using both header and citation information achieves an $F1$ as high as 0.992 and precision as high as 0.984. We apply this system to match the database of CiteSeerX against Web of Science (WOS), PubMed, and DBLP. The result indicates that the current CiteSeerX dataset includes about 3 million WoS documents, 1.62 million Medline papers, and about 1.35 million DBLP papers.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my thesis co-advisors: Dr. Clyde Lee Giles and Dr.Prasenjit Mitra for their support and guidance throughout my master program.

The National Science Foundation is gratefully acknowledged for partial support.

Chapter 1
Introduction

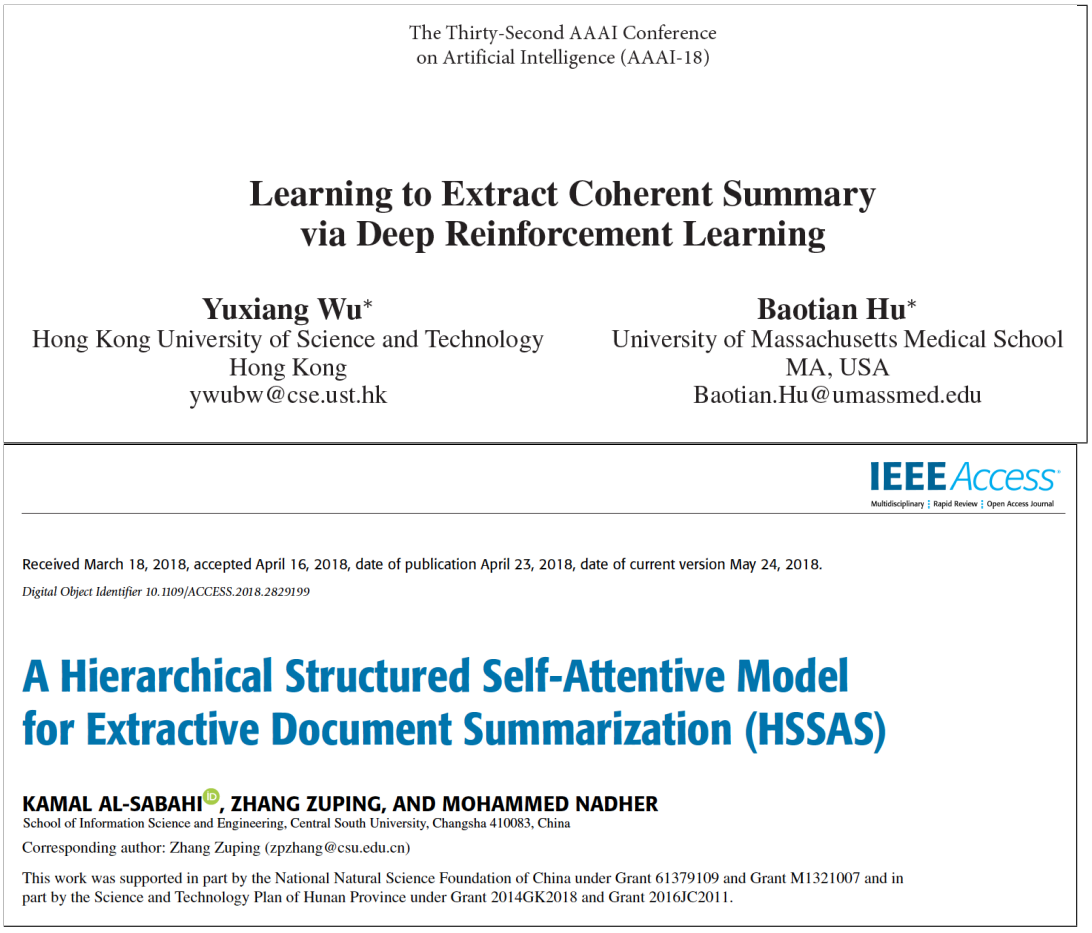
1.1 Motivation

The fast growth of scholarly big data in the form of scientific documents presents enormous challenges for scientists to find relevant literature. While many scientists still rely on subscription to journal publishers and digital libraries pertaining to specific domains behind a paywall, digital library search engines (DLSEs) such as Google Scholar, Microsoft Academic, Semantic Scholar, and CiteSeerX are gaining popularity due to their collections covering multiple subject domains, quick response, and plenty of open access resources. These digital libraries collect data by actively crawling PDF documents from the Web. Then, scientific papers are identified among the crawled PDF documents. Afterward, the PDF files are parsed to extract the content of paper including both textual and non-textual parts. CiteSeerX is a public search engine and digital library for scientific papers in the field of computer science and information science. This search engine indexes the automatically extracted metadata of the papers to enable the search APIs and the user interface of CiteSeerX search engine.

The parsers and the extractors are not perfect which results in erroneous metadata. The source of the errors comes from the heterogeneous structure of the scientific papers especially in the header part (See Figure 1.1) or missing a piece of text from the original paper in the extraction process. The erroneous metadata diminishes the quality of the search engine and reduces the robustness of the experimental results. Therefore, devising automatic tools for cleaning the noisy metadata in digital libraries such as CiteSeerX is essential. One automatic technique leverages reference datasets with reliable data, the source of which usually comes from manual input, or it has been visually inspected and verified. For example, the ACM digital library metadata are manually typed by authors.

The linking procedure is first to link entities in the target dataset to entities in a

reference dataset. The erroneous data are then overwritten by the clean data. The target corpus can also be augmented to include more information from the reference dataset.



The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Learning to Extract Coherent Summary via Deep Reinforcement Learning

Yuxiang Wu*
Hong Kong University of Science and Technology
Hong Kong
ywubw@cse.ust.hk

Baotian Hu*
University of Massachusetts Medical School
MA, USA
Baotian.Hu@umassmed.edu

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 18, 2018, accepted April 16, 2018, date of publication April 23, 2018, date of current version May 24, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2829199

A Hierarchical Structured Self-Attentive Model for Extractive Document Summarization (HSSAS)

KAMAL AL-SABAHI, ZHANG ZUPING, AND MOHAMMED NADHER
School of Information Science and Engineering, Central South University, Changsha 410083, China
Corresponding author: Zhang Zuping (zpzhang@csu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61379109 and Grant M1321007 and in part by the Science and Technology Plan of Hunan Province under Grant 2014GK2018 and Grant 2016JC2011.

Figure 1.1. Different header structures for scientific papers. The heterogeneous structure of the headers result in errors in extractions and parsing of the metadata

1.2 Problem Definition

The paper entity linking problem is defined as matching the records in a noisy scholarly dataset with a manual input high-quality dataset. We denote the target corpus, which contains noisy data, as T, containing n paper entities $t_i, 1 \leq i \leq n$, and a reference corpus R, which contains reference data, containing m paper entities $r_j, 1 \leq j \leq m$. Each entity can be represented by a number of attributes. Our goal is to find a set M,

$$M = \{(t, r); t = r, t \in T, r \in R\}.$$

The paper entity linking can be formalized as a binary classification problem in which the classifier decides whether a candidate pair is a real match or not.

1.3 Related Work

Record linkage is a well-studied research topic [1] [2] which aims to identify the same items in different datasets. The matching process between big datasets requires the application of information retrieval tools in order to improve the efficacy of the models. Blocking functions improve the scalability of the alignment by limiting the search space [3].

Record linkage between digital libraries had been investigated in [4] and [5]. Caragea et al. [4] indexed the DBLP dataset as their reference with Apache Solr. Multiple attributes of a record in the target set are searched against the reference index. The candidate matches are ranked based on their similarity to the query. Their result using 3-grams of titles with the Jaccard similarity threshold of 0.7 achieved the best result with $F1 = 0.77$ on their manually generated ground truth. Since the final goal of linking is to overwrite the metadata of noisy records in CiteSeerX with metadata of reference sets, the precision of the matching module is very important as we don't want to replace metadata of a paper with metadata of another document. Therefore, the model proposed by [4] can not be used in the cleaning of CiteSeerX data due to the low precision. Wang et al. [5] claim the performance of $F1 = 0.96$ for their online matching algorithm. However, we could not replicate their work.

Another similar approach to paper matching is author name disambiguation [6]. The goal is to put the papers of an author in one cluster by pairwise comparison between papers. This results in matching of the same authors in different papers.

There are other applications of record linkage in matching free text such as wiki articles [7]. This task is challenging as different languages can be used to present same topics.

Chapter 2
Data

2.1 Datasets

We investigated the linkage between CiteSeerX and 4 big scholarly data sets: Web Of Science, PubMed, DBLP and IEEE.

2.2 CiteSeerX

CiteSeerX [1] data is a library of over $10M$ scientific articles in the field of computer science and information science. Less than 15% of CiteSeerX papers have unique identifiers such as DOIs as most of the papers are preprints provided by their authors. These articles are in PDF and need to be parsed into textual units. The SVMHeaderParser [8] is used to extract the header metadata of the papers. With a variety of header formats in scientific papers (Figure 1.1), designing a general parser to extract the header information is challenging. Citation information is also available and could be used for the matching process. The advantage of using citation information is that usually, parsers show better performance in the processing of bibliographic data at the end of the papers as they are more structured with limited boundaries and predictable language. CiteSeerX uses the ParsCit [9] package to get the citation metadata out of PDF files. ParsCit is based on conditional random filed and labels the token sequences in a reference string.

[1] https://citeseerx.ist.psu.edu

2.3 Web of Science (WoS)

Web of Science digital library is a comprehensive collection of scientific papers and books. The WoS has 45,261,744 articles and 906,825,446 citations. Only 2% of WoS papers are in the field of computer science with the following subjects starting with "Computer Science", which are "Computer Science", "Computer Science, Artificial Intelligence", "Computer Science, Cybernetics", "Computer Science, Hardware & Architecture", "Computer Science, Information Systems", "Computer Science, Interdisciplinary Applications", "Computer Science, Software Engineering" and "Computer Science, Theory & Methods".

Web of Science is known to have a very high quality. However, there are some errors in some papers especially the old ones.

1. Titles of many papers (0.34%) are 'untitled' or 'corrections'

2. Titles of many citations (17.8%) are Null.

2.4 PubMed

Medline is the premier bibliographic dataset released by NCBI [2] with about 24 million academic papers in the area of biomedicine published since 1966. Citation information does not exist in this dataset. The Medical Subject Headings (MeSH) thesaurus is a hierarchically-organized vocabulary that is controlled and maintained by the National Library of Medicine. It is used for indexing and searching of biomedical and health-related literature. A MeSH term describes the subject of an article (e.g., Alzheimer's disease) with a short description. Currently, there are about 26,000 MeSH terms and are regularly updated to reflect the changes in the biomedicine field. Mesh terms in the higher levels of the hierarchical structure are more general and more specific terms are under the more general terms. All of the papers in PubMed are tagged with related MeSH keywords and these keywords are good clues for the linkage problem. Part of the MeSH terms related to Alzheimer's disease is depicted in Figure 2.1.

2.5 DBLP

DBLP is a bibliographic dataset covering more than $5,000$ conferences and $1,500$ journals in computer science. We use the version published in March, 2017 with about 4 million

[2] https://www.ncbi.nlm.nih.gov/

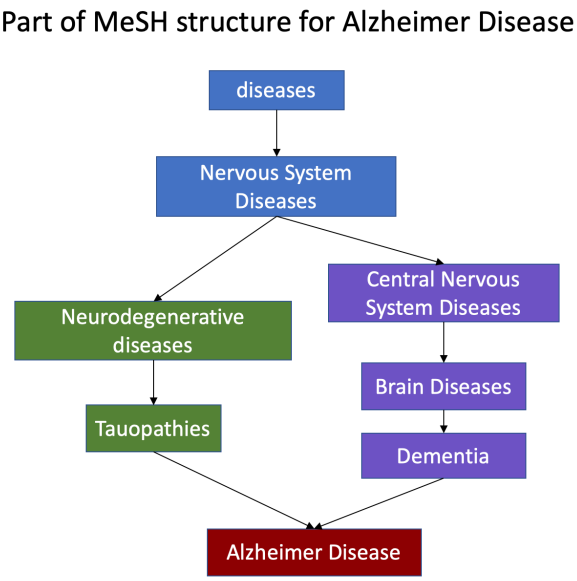


Figure 2.1. Part of MeSH structure for Alzheimer disease

documents. This dataset does not contain citation information. The original data is in XML format with document type definition file dblp.dtd to validate the XML file.

2.6 IEEE

The IEEE corpus is a subset of the IEEE Xplore database, containing about 2 million bibliographic records downloaded from IEEE FTP sites. It does not contain citations.

Chapter 3
Method

3.1 Paper Entity Matching Problem

The problem is defined as matching the records in a noisy scholarly dataset with a manual input high-quality dataset. We denote the target corpus, which contains noisy data, as T, containing n paper entities $t_i, 1 \leq i \leq n$, and a reference corpus R, which contains reference data, containing m paper entities $r_j, 1 \leq j \leq m$. Each entity can be represented by a number of attributes. Our goal is to find a set M,

$$M = \{(t, r); t = r, t \in T, r \in R\}.$$

Paper entity linking can be formalized as a binary classification problem in which the classifier decides whether a candidate pair is a real match or not.

An officially published paper usually is assigned a unique identifier, i.e., DOI. A journal article can also be identified by the journal name, the volume, the issue number, and the starting page number. However, for a large fraction of open access scholarly papers crawled from the Web, such information is not fully available. Empirically, a paper entity can be uniquely identified by four header fields,

(title, authors, year, venue),

in which the venue is a conference or a journal. These fields are very heterogeneous in many aspects. For example, titles may contain non-ASCII characters, and the encoding may vary when they are extracted from PDF files. Author names may contain diacritics, e.g., "Ádám" and should be converted to "Ádám" and then normalized to "Adam". A manuscript may contain a timestamp before the official publication date. So, an offset should be allowed for the year attribute. The venue name may be expressed

in different ways. Ideally, the venue names should be normalized and resolved into a canonical form. Due to a lack of general venue disambiguator for domains beyond computer science venue information is not incorporated in the header model in this work. A citation record parsed from a citation string usually contains the above four fields. So, matching a single citation record can be done in the same manner as matching a paper entity. The abstract is usually a paragraph of text that may contain non-ASCII characters with different encodings, but normalizing abstracts and calculating SimHash values takes heavy overhead, so in this work, we use abstracts without normalization.

Because many digital library datasets do not have citation information, we consider two separate models, one with only header information called Header Matching Model (HMM), and the other with only citation information called Citation Matching Model (CMM). The combination of them is called the Integrated Matching Model (IMM). The Title Evaluation Model (TEM) is designed to evaluate the quality of the title. If title quality is acceptable the matching is done using the header information. However, if the title quality is low, we use the integrated model to find the matches.

3.1.1 Header Matching Model (HMM)

HMM is a supervised machine learning model to classify candidate pairs using information from the paper header.

One prerequisite of HMM is indexing all document metadata in the reference corpus using an open-source search platform. We use ElasticSearch (ES) because of its relatively low setup overhead and scalability. The default settings are applied for our experiments. The indexed metadata contains header fields including titles, authors, abstracts, and years. For each paper in the target corpus, we query the title against the index, if it contains a minimum of 20 characters. Otherwise, the first author's last name and publication year are used in queries. If the year is not available, only the first author's last name is used. For each field, the query string is segmented into unigrams connected by "OR". The ranking scores are calculated using the Okapi BM25 algorithm [10]. The query algorithm is shown in Algorithm 1.

For each target paper, the top 10 papers from the reference set are retrieved and 10 candidates are formed as matching pairs. The features of each pair include a list of similarities calculated using the header metadata.

1. Title similarity represented by Levenshtein distance of SimHashes of normalized

Algorithm 1 Query Builder

```
1: function QUERY(title, lastName, year)
2:     if title ≠ Null and title.length > 20 then
3:         query ← title
4:     else if lastName ≠ Null and year ≠ Null then
5:         query ← lastName and year
6:     else if lastName ≠ Null then
7:         query ← lastName
8:     end if
9:     return query
10: end function
```

titles [11]:

$$\mathrm{Sim}_L(\mathrm{title}_t, \mathrm{title}_r) = \mathrm{lev}_{a,b}\left(|a|, |b|\right) \tag{3.1}$$

$$a = \mathrm{SimHash}_{16}(\mathrm{Norm}(\mathrm{title}_t)) \tag{3.2}$$

$$b = \mathrm{SimHash}_{16}(\mathrm{Norm}(\mathrm{title}_r)) \tag{3.3}$$

in which a SimHash string contains 16 alphanumeric characters. The normalization process of titles is as follows:

(a) All letters are lowercased;

(b) Diacritics are removed, e.g., "á" is converted to "a";

(c) Consecutive spaces are collapsed;

(d) Punctuation marks are trimmed;

(e) Single characters "s" and "t" are removed because they are mostly resulted from removing the apostrophe from possessives or abbreviations such as "can't".

2. Abstract similarity $\mathrm{Sim}_L(\mathrm{abstract}_t, \mathrm{abstract}_r)$ represented by Levenshtein distance of SimHashes of abstracts, calculated in a similar way as Equations (1)–(3) without normalization.

3. Jaccard similarities between normalized titles and original abstracts. For example,

$$\mathrm{Sim}_J(\mathrm{title}_t, \mathrm{title}_r) = \frac{|W_t \cap W_r|}{|W_t \cup W_r|} \tag{3.4}$$

in which W_t and W_r represent the token set of the title of the target and the reference paper, respectively.

4. The absolute difference of the years.

5. The **first and the last author's full name similarity**. Author similarities are measured in multiple metrics. An author's full name similarity is represented by a three-digit binary lmf, representing whether the last name, the middle initial, and the first initial matches, respectively. If a certain name component is missing or it does not match, the binary is set to 0. The decimal value of a binary is used as the full name similarity index. Author names are also normalized before comparison. Diacritics are removed and letters are lowercased. Prefixes, e.g., Prof., and suffixes, e.g., "II", and their variants are removed. For example, if first authors are "Jane C. Huck" and "J. Huck", the binary is 101, and the name similarity is 5.

6. The **last name similarities of the first and last author.** The last name similarity is computed in this way

$$\text{Sim}(\text{N}_t, \text{N}_r) = \begin{cases} \text{0: } \text{N}_t \neq \text{NULL} \wedge \text{N}_r \neq \text{NULL} \wedge \text{N}_t \neq \text{N}_r \\ \text{1: } \text{N}_t = \text{NULL} \vee \text{N}_r = \text{NULL} \\ \text{2: } \text{N}_t \neq \text{NULL} \wedge \text{N}_r \neq \text{NULL} \wedge \text{N}_t = \text{N}_r \end{cases} \tag{3.5}$$

   in which N is the last name and NULL means the value is not available.

7. **All authors' last name Jaccard similarity**

$$\text{Sim}_J\left(L_t, L_r\right) = \frac{|L_t \cap L_r|}{|L_t \cup L_r|} \tag{3.6}$$

   in which L_t and L_r stand for the set of last names in the target and the reference corpora, respectively.

The pseudocodes of the HMM is in Algorithm 2.

3.1.2 Citation Matching Model (CMM)

CMM is an unsupervised model that matches paper entities by citations. The paper entity matching problem can benefit from this model when HMM cannot find a matching entity because the header metadata is noisy but the references are available. Similar to HMM, a prerequisite is to index all citations in the reference corpus. On average, one paper contains about 20 citations [12], so the citation index is usually much larger than the document index, which slows down the searching. Besides the header information of

Algorithm 2 Header Matching Model

```
function HMM()
    T ← target corpus
    R ← reference corpus
    index_R ← index of reference corpus
    matchList ←Ø
    for t ∈ T do
        Q ← Query(t.title, t.firstLastName, t.year)
        Candidates ← query Q to index_R
        for c ∈ Candidates do
            prediction←Model.predict(t,c)
            if prediction=1 then
                matchList.add (t, c)
                break
            end if
        end for
    end for
end function
```

each citation record, it is important to store the paper ID by which these references are cited [1]

In CMM, for a target paper t, its citation records tc_i are retrieved from the database. We attempt to find the matching record rc_j in the reference corpus using the query builder in Algorithm 1. Retrieved citations and rc_i are matched by the HMM model in Algorithm 2. Citations do not contain abstracts so relevant features are not used. Assuming such $j = 1$ exists (if not, then no matching entity is found) and rc_1 is cited by r_1, the next step is to compare r_1 with t. The CMM uses both the *paper title* and the *citation titles* (Algorithm 3). First, the title similarities are calculated using Equations (1)–(3). If this distance is less than a threshold of θ_{title}, r_1 is believed to be the matching entity of t. Otherwise, CMM extracts the tokens from all the reference titles of t, denoted by BoW_t, and tokens from all the reference titles of r_1, denoted by BoW_r. The judgment is made by comparing the Jaccard similarity between BoW_t and BoW_r. If the similarity is greater than a threshold θ_{ref}, then (t, r_1) is determined as a matching pair. Otherwise, the algorithm continues to examine the next paper r_3 (because r_3 also shares rc_1 with t). If no papers citing rc_1 is found to be a match for t, CMM continues and attempts to find the matching record of tc_2. It should be noted that when title quality is low, the similarity between matching pairs is captured by the BoW representation of citations.

[1] There is a difference between *indexed* and *stored*. Only searchable fields are *indexed*. The paper IDs are *stored* and retrieved along with the associated citation records, but it is not indexed.

Algorithm 3 Citation Matching Model

```
1: function CMM()
2:     T ← target corpus
3:     R ← reference corpus
4:     matchList ←Ø
5:     CitationIndex_R ← citations index of reference corpus
6:     for t ∈ T do
7:         t_citations ← citations of t.
8:         for tc_i ∈ t_citations do
9:             Q ← Query (tc_i.title,tc_i.firstLastName, tc_i.year)
10:            results ← query Q to CitationIndex_R
11:            for rc_j ∈ results do
12:                prediction←Model.predict (tc_i, rc_j)
13:                if prediction=1 then                          ▷ citations match
14:                    r_k ← paper that cites rc_j
15:                    r.title ← SimHash (r_k.title)
16:                    t.title ← SimHash (t.title)
17:                    title_dist = lev (t.title, r.title)
18:                    if title_dist < θ_title then
19:                        matchList.add (t, r_k)
20:                        break
21:                    end if
22:                    BoW_r ← BoW (r_k.referenceTitles)
23:                    BoW_t ← BoW (t.referenceTitles)
24:                    sim ← Jaccard(BoW_t_i, BoW_r_i)
25:                    if sim > θ_ref then
26:                        matchList.add (t, r_k)
27:                        break
28:                    end if
29:                end if
30:            end for
31:        end for
32:    end for
33: end function
```

3.1.3 Title Evaluation Model (TEM)

TEM is a light-weight supervised learning model designed to provide a quantitative evaluation of the title quality. The input is a title string, and the output is a probability θ of how likely the input string looks like a paper title. The title quality is determined to be high if θ is greater than a threshold θ_{tq}. The TEM exploits the features in Table 3.1.3 extracted from the original title string. The TEM is trained on a sample of 8200 title strings containing 6270 high-quality titles and 1930 titles with low quality. Titles are

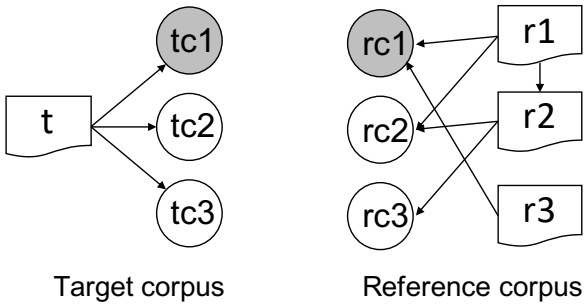


Figure 3.1. An illustration of CMM. t is a document in the target corpus; r_i is a document in the reference corpus. rc_j is a citation record in the reference corpus. $t \rightarrow tc_1$ means t cites tc_1. tc_1 and rc_1 are highlighted because they are identified to be a matching pair.

labeled to be of low quality if:

1. They are NULL;
2. They have many non-ASCII characters;
3. They include evidently irrelevant information such as authors;
4. They are not in English.

A few examples of low quality titles are listed in Figure 3.1.

Table 3.1. Examples of wrongly parsed titles. The gibberish characters are intentionally quoted from extraction results.

Bad Titles
" A New Metric for Banking ..." by Reint Gropp and Anil K. Kashyap
A D-A 213 994
! l)JL fY15/e-K ♦ rfl Vi!J ♦ 'rÃĆÂůy
!DOCUMENT TRANSMITTAL AND RECEIPT ACKNOWLEDGEMENT FORM "VOLUM?

Four supervised models, including Logistic Regression (LR), Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forests (RF), are trained, 10-fold cross-validated and compared. The LR model achieves an F1-measure greater than 0.999, which we adopted.

3.1.4 Integrated Matching Model

IMM integrates HMM, CMM, and TEM (Figure 3.2). If HMM is able to find the match of a paper entity, the process continues to the next paper. Otherwise, the paper title

Table 3.2. Features used to train TEM.

Character-level features
#ASCII characters #non-ASCII characters #white spaces
#punctuation marks #consecutive punctuation marks #digits
Type of the first/last character (punctuation, digit, or letter)
Word-level features
$\#\max(\mathrm{DF}(w)), w \notin \mathbb{S}$[1] $\#\min(\mathrm{DF}(w)), w \notin \mathbb{S}$
$\#\mathrm{media}(\mathrm{DF}(w)), w \notin \mathbb{S}$ #words
#Appearance in a controlled list[2] {Abstract, LIST,Acknowledgments, NOTICES, CONTENTS, Accepted, CONTENT, Authors, References, ACKNOWLEDGMENTS, NULL, Chapter, Discussions, Summary}

[1] DF: document frequency, calculated on all DBLP titles. $\mathbb{S}$ is a set of stopwords adopted from Apache Solr.

[2] The value is set to 1 only if the string contains at least one exact match to the controlled list.

quality is evaluated by TEM. If the title quality is high ($\theta \geq \theta_{tq}$), it is likely that there is not a matching entity existing in the reference corpus, otherwise ($\theta < \theta_{tq}$), the matching entity is not found due to the poor title quality. In this situation, there is a chance that citation information can be used if available.

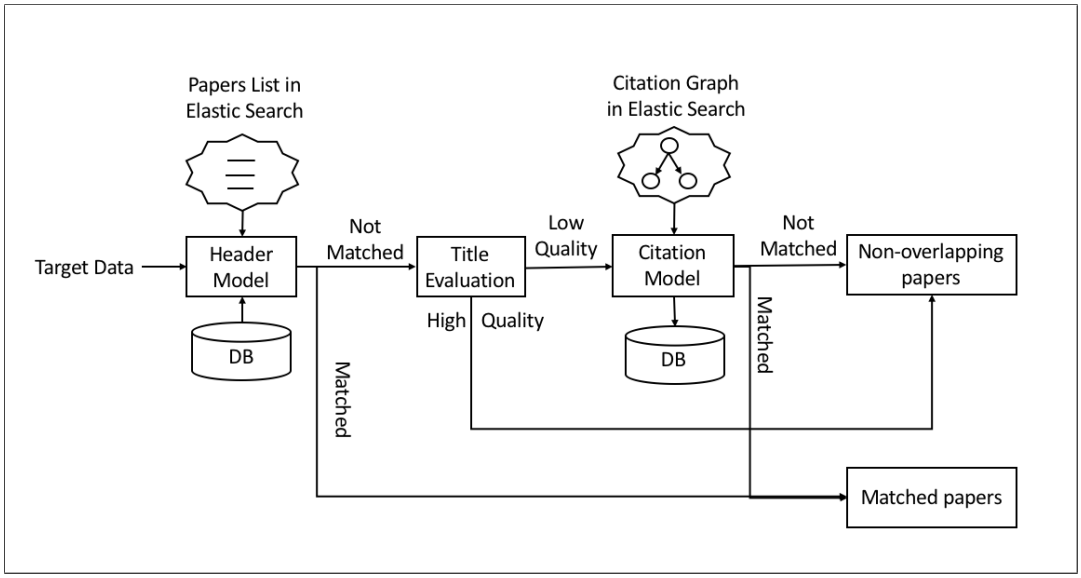


Figure 3.2. Pipeline of IMM.

Chapter 4
Experiments and Results

We evaluated the proposed models by a manually generated ground truth. Procedure of ground truth labeling is described in section 4.0.1. Final results and the related discussions are in section 4.0.4.1.

4.0.1 Ground Truth Labeling

The procedure for labeling process comprises three steps

- First, for each paper in the CiteSeerX sample set, a candidate set of 10 papers is retrieved from the reference index using the same manner described in Algorithm 1.
- To determine the true match out of candidate papers, other metadata of the papers including authors, abstract, year, venue, keywords, and the number of pages were visually inspected independently by two graduate students.
- Finally, if it was not possible to decide based on papers profiles, actual PDF files were used side by side to make a final decision.

We generated the following ground truth datasets:

- **CiteSeerX-IEEE** This dataset, adopted from Wu et al. [13], is built based on 1000 CiteSeerX papers with 51 true matching pairs found in the IEEE corpus.
- **CiteSeerX-DBLP** This dataset, revised based on Caragea et al. [14], contains 292 matching pairs identified between 1000 CiteSeerX papers and the DBLP dataset.
- **CiteSeerX-WoS** This dataset contains 345 matching papers found in WoS out of 533 CiteSeerX papers.

- **Combined Sample** This dataset contains 688 matching pairs, which forms the positive samples. The negative samples are selected using 1845 candidate matching pairs, containing the most similar but unmatched papers.

4.0.2 Evaluation Metrics

The performance of binary classification problems are evaluated by precision, recall, and F1-measure defined below:

Precision, Recall, and F-Measure are defined as:

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

$$F1 = \frac{2Precision \times Recall}{Precision + Recall} \tag{4.3}$$

Where TP, FP, FN represent the True Positives, False Positives, and False Negatives, respectively. Symbols TP, FP, and FN are defined as follows in entity linking problem: TP are pairs identified as a match, which are indeed matches; FP are pairs identified as matches, which in fact are not matched against reference dataset; FN are pairs that they match but are not identified by the algorithm.

Precision gives the fraction of pairs correctly identified as a match among all pairs identified as matches, and Recall gives the fraction of matched pair correctly identified by the algorithm among all positive matching pairs. F-Measure gives the harmonic mean of Precision and Recall.

4.0.3 Experiment Setups

We trained binary classifiers that decide whether a pair of candidate matching pair from target and reference corpora is a true matching pair. Four machine learning models, Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), and XGBoost are trained on the Combined Sample (Section 4.0.1). The grid search is applied to tune and find the hyperparameters yielding the best results. Due to the relatively small size of ground truth data, we use 10-fold CV to evaluate our models. Precision, Recall, and F1-Measure values are reported in Table 4.0.4.1.

All experiments were run on a machine with the following hardware and software

specifications: CPU: ex32 x Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GH; RAM: 330 GB; OS: Red Hat Enterprise Linux (RHEL).

The training and testing are implemented using Scikit-learn v0.19.1 under Python v3.6.

4.0.4 Results and Discussion

4.0.4.1 HMM Results

The 10-fold cross validated results of the HMM are tabulated in Table 4.0.4.1. The precision-recall curves are demonstrated in Figure 4.0.4.1.

Table 4.1. HMM model 10-fold CV results.

Model	Precision	Recall	F1-measure
SVM	**0.926**	**0.937**	**0.931**
LR	0.794	0.968	0.872
RF	0.912	0.931	0.921
XGBoost	0.925	0.899	0.912

In four models, SVM achieves the highest F1-measure. RF has a comparable F1-measure but requires a significantly reduced training period ($< 30\%$), so we employ RF for HMM. The information gain (IG) is calculated for each feature in Table 4.0.4.1 indicating that the most informative features are related to titles and the first authors.

To make an even comparison with the method proposed by Caragea et al. [14], we rerun their experiments on the CiteSeerX-DBLP ground truth with the best parameter settings in which $n = 3$ and the Jaccard similarity threshold $\theta_J = 0.7$. We then compare the results with our RF model trained on the combination of CiteSeerX-WoS

Table 4.2. The top 7 features of HMM ranked by IG.

Rank	IG score	Feature Names
1	0.629	$\text{Sim}_L(\text{title}_t, \text{title}_r)$
2	0.139	$\text{Sim}_J(\text{title}_t, \text{title}_r)$
3	0.065	First author's last name similarity
4	0.064	First author's full name similarity
5	0.056	Last author's full name similarity
6	0.016	$\text{Sim}_J(L_t, L_r)$
7	0.015	$\text{Sim}_L(\text{abstract}_t, \text{abstract}_r)$
8	0.006	Jaccard similarity of the BoW of abstracts
9	0.006	Absolute year difference
10	0.003	Last author last name similarity

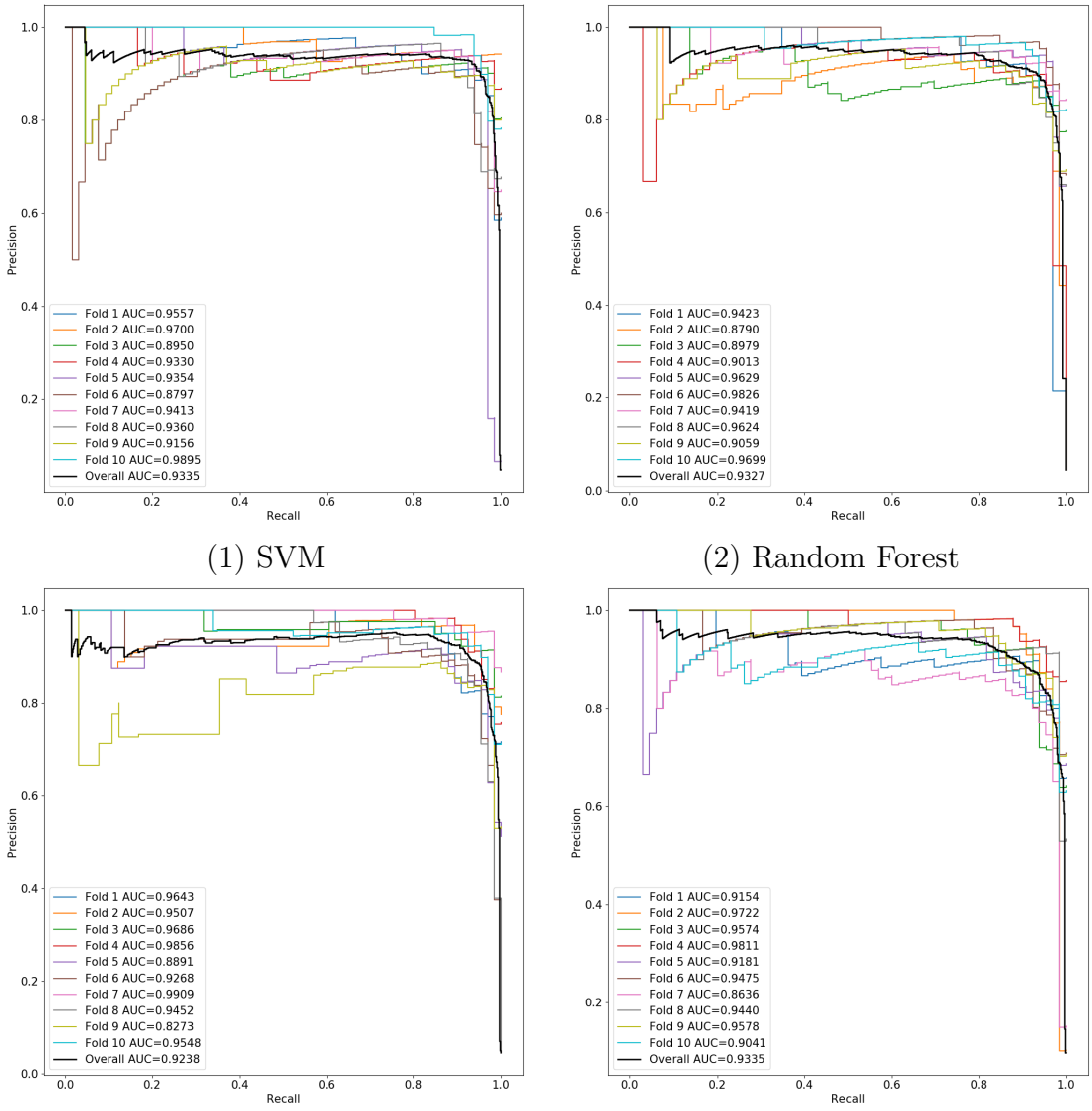


Figure 4.1. Precision-recall curve for HMM model using Random Forest, XGBoost, SVM, and Logistic Regression. Black curves is overall performance for all 10-folds.

and CiteSeerX-IEEE datasets. The HMM outperforms the IR-based model with 14% improvement in precision (100% vs. 86%) and a 3% improvement in the F1 score (91% vs. 88%). One advantage of our model is to achieve higher precision while sacrificing the recall within the tolerance, because as we stated before, we should minimize the number of papers whose metadata are mistakenly "corrected".

Table 4.3. IR vs. ML model

model	Precision	Recall	F1
Random Forest	1.00	0.830	0.91
IR-based	0.86	0.91	0.88

4.0.4.2 TEM Results

The purpose of the title evaluation model (TEM) is to decide when it is necessary to use CMM for matching. The binary classifier made to recognize bad titles is evaluated again by Precision, Recall, and F-Measure for the positive class (bad titles) since we are mainly interested in the identification of low quality titles. Precision gives the fraction of titles correctly identified as a bad title by our classifier among all bad titles identified by algorithm, whereas Recall gives the fraction of bad titles correctly identified by the algorithm among all bad titles.

Table 4.0.4.2 presents 10 fold cross validation results of Title Evaluation model for four machine learning algorithms namely, Logistic Regression (LR), Support Vector Machine (SVM), Naïve Bayes (NB), and Random Forest (RF). Number of words, number of consecutive punctuation marks and number of ASCII letters play the most important roles in the Title classification task.

Table 4.4. TEM 10-fold CV results.

Model	Precision	Recall	F1
LR	**0.9995**	**0.9999**	**0.9997**
SVM	0.9982	0.9958	0.9970
NB	0.9362	0.9855	0.9602
RF	0.9989	0.9992	0.9990

4.0.4.3 CMM Results

Table 4.0.4.3 demonstrates the results of the CMM classifier on the CiteSeerX-WoS dataset. We investigate how the reference Jaccard similarity threshold θ_{ref} and title Levenshtein distance θ_{title} affects the performance of CMM in (Algorithm 3). A higher value of θ_{ref} indicates that two papers need more common citations to be considered as a matching pair. The best F1 score is obtained at $\theta_{ref} = 0.5$ and $\theta_{title} = 0.35$.

Table 4.5. The CMM performance with different θ_{ref} and θ_{title}.

θ_{ref}	θ_{title}	Precision	Recall	F1
0.40	0.15	0.876	0.719	0.790
	0.25	0.877	0.725	0.794
	0.35	0.878	**0.730**	0.797
	0.45	0.850	0.725	0.782
0.50	0.15	0.968	0.690	0.797
	0.25	0.969	0.714	0.822
	0.35	0.965	0.728	**0.830**
	0.45	0.927	0.725	0.814
0.60	0.15	0.982	0.651	0.783
	0.25	**0.983**	0.662	0.791
	0.35	0.979	0.691	0.810
	0.45	0.938	0.691	0.796
0.70	0.15	0.955	0.609	0.743
	0.25	0.955	0.620	0.752
	0.35	0.953	0.652	0.775
	0.45	0. 912	0.658	0.764

4.0.4.4 IMM Results

Table 4.0.4.4 compares HMM, CMM, and IMM based on the CiteSeerX-WoS dataset (because only this dataset contains citations). In the first column, as the threshold θ_{tq} increases from 0.01 to 0.2, the testing corpus encloses more papers with higher quality titles, which results in a better performance of HMM. The CMM alone is getting better with constantly remarkably high precision but poor recalls. The integrated model achieves both high recall and precision. This indicates that (1) CMM tend to be more useful when the title quality is low; (2) The integrated model significantly increases the overall performance, especially for papers with low quality titles.

One result in Table 4.0.4.4 that is counter-intuitive is the HMM consistently achieves high performance when the title quality is low. To answer this question, we trained a RF classifier on papers with low-quality tiles only. Information gained from features of the new model reveals that the most important features in the absence of good titles are First author features, Jaccard similarity of all authors' last names, and Abstract features, implying that when title quality is low, accurate author information can also provide accurate matches.

Table 4.6. Comparisons of HMM, CMM, and IMM performances using the CiteSeerX-WoS dataset with different title quality thresholds. T/s stands for testing time in seconds.

$\theta < \theta_{tq}$	**Data Portion**	**HMM**			
		P	R	F1	T/s
$\theta < 0.01$	16.1 %	0.971	0.872	0.919	10
$\theta < 0.02$	17.8 %	0.973	0.857	0.911	12
$\theta < 0.10$	21.20%	0.978	0.833	0.900	14
$\theta < 0.20$	24.39%	0.981	0.869	**0.922**	14.5
$\theta < \theta_{tq}$	**Data Portion**	**CMM**			
		P	R	F1	T/s
$\theta < 0.01$	16.1 %	1.0	0.513	0.678	
$\theta < 0.02$	17.8 %	1.0	0.524	0.688	12761
$\theta < 0.10$	21.20%	1.0	0.593	0.745	13732
$\theta < 0.20$	24.39%	1.0	0.59	0.742	14823
$\theta < \theta_{tq}$	**Data Portion**	**IMM**			
		P	R	F1	T/s
$\theta < 0.01$	16.1 %	0.975	1.0	0.987	9082
$\theta < 0.02$	17.8 %	0.977	1.0	0.988	9791
$\theta < 0.10$	21.20%	0.982	1.0	0.991	10206
$\theta < 0.20$	24.39%	0.984	1.0	**0.992**	11490

4.0.5 Error Analysis

Although the combination of HMM and CMM achieve superior performance, the recall of CMM alone is poor (Table 4.0.4.4). This could be due to two reasons: (1) Citation parsing errors. For example, more than 1 million papers in CiteSeerX contain less than 5 citations; (2) Null title citations. In Wos, there are 17% citation records and 8.4% citations in CiteSeerX have null titles.

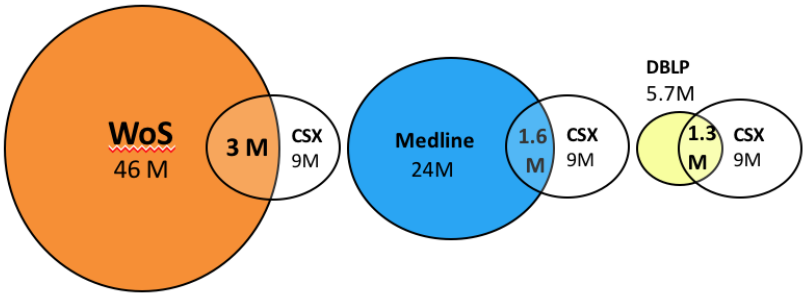
The citation-based model is slow because (1) searching through 906 million WoS citations is time-consuming and (2) the candidate set for each CiteSeerX citation could be huge for highly-cited papers. The integrated model only applies citation model to papers with low-quality titles to improve recall.

4.0.6 Application and Conclusion

We applied HMM on CiteSeerX documents against DBLP, WoS, and Medline. Figure 4.0.6 reveals overlaps between CiteSeerX and these datasets. The result indicates that **the current CiteSeerX dataset includes about 3 million WoS documents, 1.62 million Medline papers, and about 1.35 million DBLP papers**. The result

reveals that there is still a large number of papers that CiteSeerX can index. The unmatched document metadata can aid CiteSeerX find relevant resources in its focused web crawling [15].

Figure 4.2. Overlap between CiteSeerX and other repositories.



Bibliography

[1] Herzog, T. N., F. J. Scheuren, and W. E. Winkler (2007) *Data quality and record linkage techniques*, Springer Science & Business Media.

[2] Christen, P. (2012) *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*, Springer Science & Business Media.

[3] Papadakis, G., J. Svirsky, A. Gal, and T. Palpanas (2016) "Comparative analysis of approximate blocking techniques for entity resolution," *Proceedings of the VLDB Endowment*, **9**(9), pp. 684–695.

[4] Caragea, C., J. Wu, A. Ciobanu, K. Williams, J. Fernández-Ramírez, H.-H. Chen, Z. Wu, and L. Giles (2014) "Citeseer x: A scholarly big dataset," in *European Conference on Information Retrieval*, Springer, pp. 311–322.

[5] Wang, Y., H. Zhang, Y. Li, D. Wang, Y. Ma, T. Zhou, and J. Lu (2016) "A data cleaning method for citeseer dataset," in *International conference on web information systems engineering*, Springer, pp. 35–49.

[6] Smalheiser, N. R. and V. I. Torvik (2009) "Author name disambiguation," *Annual review of information science and technology*, **43**(1), pp. 1–43.

[7] Yang, Y., Y. Sun, J. Tang, B. Ma, and J. Li (2015) "Entity matching across heterogeneous sources," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, pp. 1395–1404.

[8] Han, H., C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox (2003) "Automatic document metadata extraction using support vector machines," in *2003 Joint Conference on Digital Libraries, 2003. Proceedings.*, IEEE, pp. 37–48.

[9] Councill, I. G., C. L. Giles, and M.-Y. Kan (2008) "ParsCit: an Open-source CRF Reference String Parsing Package." in *LREC*, vol. 8, pp. 661–667.

[10] Robertson, S., H. Zaragoza, and M. Taylor (2004) "Simple BM25 Extension to Multiple Weighted Fields," in *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, ACM, New York, NY, USA, pp. 42–49.
URL `http://doi.acm.org/10.1145/1031171.1031181`

[11] Charikar, M. (2002) "Similarity estimation techniques from rounding algorithms," in *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pp. 380–388.
URL http://doi.acm.org/10.1145/509907.509965

[12] Wu, J., C. Liang, H. Yang, and C. L. Giles (2016) "CiteSeerX Data: Semanticizing Scholarly Papers," in *Proceedings of the International Workshop on Semantic Big Data*, SBD '16, ACM, New York, NY, USA, pp. 2:1–2:6.
URL http://doi.acm.org/10.1145/2928294.2928306

[13] Wu, J., A. Sefid, A. C. Ge, and C. L. Giles (2017) "A Supervised Learning Approach To Entity Matching Between Scholarly Big Datasets," in *Proceedings of the Knowledge Capture Conference*, K-CAP 2017, ACM, New York, NY, USA, pp. 42:1–42:4.
URL http://doi.acm.org/10.1145/3148011.3154470

[14] Caragea, C., J. Wu, A. Ciobanu, K. Williams, J. Fernández-Ramírez, H.-H. Chen, Z. Wu, and L. Giles (2014) *CiteSeerX: A Scholarly Big Dataset*, Springer International Publishing, Cham, pp. 311–322.
URL http://dx.doi.org/10.1007/978-3-319-06028-6_26

[15] (2019) *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, AAAI Press.
URL https://www.aaai.org/Library/AAAI/aaai19contents.php