Measuring the Use of ArXiv in Computer Science Research

Linan Gong

Master of Science
Artificial Intelligence
School of Informatics
University of Edinburgh
2016

Abstract

ArXiv is one of the most important preprint repositories that allows researchers submit their papers before peer-reviewed by other researchers or accepted by the conferences. Without the limitation of deadlines and barriers in the academic communication, it becomes more popular in many research areas. This paper aims to find out the features of researchers behaviors in the use of arXiv in the subject area of computer science. Computing similarities through a similarity algorithm between conference papers and arXiv papers, in order to find out the number of conference paper submitted on arXiv. Then using a matching algorithm to predict whether the paper pairs are matched. After getting the results of matched papers of conferences, we analyze the data and gain information and evidence to understand the differences of their behaviors across different computer science research areas or different sub areas within a same computer science field.

Acknowledgements

I would like to express my sincere gratitude to my supervisor Charles Sutton for his continuous support of my master project, for his patience, motivation and immense knowledge. His guidance helped me in all the time and provided me many ideas to do this project and writing this thesis.

Besides, I would like to thank my friends Patty, Anny and Clair for supporting me during my whole academic year and for sharing many sleepless nights with me in Forrest Hill.

I am also deeply grateful to my friend Karl, who always helps me to solve many programming problems and discusses with me about his ideas towards research.

Finally, I want to thank my parents, simply for everything. They always encourage me to believe myself that I can achieve my goals.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Linan Gong*)

Table of Contents

Chapter 1

Introduction

1.1 Introduction of arXiv

Open access (OA) springs up along with the rapidly development of Internet in recent years. It provides the opportunities for researchers to take advantage of the free-global sharing, hence, researchers in many subject areas are more likely to submit their research ideas and results to a preprint repository. While ArXiv is one of this kind of repositories. The arXiv.org is a highly automated electronic archive and distribution server for research areas, which started in August 1991, and hosts over 1 million e-prints of research papers covered in physics, mathematics, computer Science, quantitative biology, quantitative finance and statistics[1]. Moreover, more than 8000 articles are submitted on arXiv per month at present.

For the scholarly communication, it represents one of the greatest technological innovation in the use of web, although it is in the running for 25 years (Tennant, 2016). Majority of the articles submitted to the arXiv are also submitted to the traditional conferences and journals later or before and the content never goes beyond its defines. Also, the open access articles have a higher citation counts without considering the potential confounding factors (Craig et al., 2007).

The reason of the popularity in the use of arXiv is that it is digital, online and the most important one is free. It removes the price barriers such as subscriptions, licensing fees and pay-per-view fees (Suber, 2007). Also this advantage makes it more flexible to submit papers on arXiv, and researchers can cite their own previous work or results of other researchers without the limitation that has to wait for the publication of these papers.

[1] https://arxiv.org

1.2 Object

This popularity influences the use of arXiv in many subject areas and leads to an increasing trend of the submissions on arXiv. This paper aims to measure the use of arXiv in computer science research area. Situations must be quite different in different CS sub fields. The behaviors of scientists vary across research areas or within a specific area. They may have different ideas and attitudes towards using the preprints. Collecting the useful information of arXiv papers and conference papers, finding out the features of the behaviors of researchers, and then analyzing the data as well as gaining information and evidence to understand differences of their behaviors across different sub-field or areas within sub-area in computer science.

Using the matching algorithm to calculate the similarities of a conference paper with a pile of arXiv papers and predicting whether the paper pairs are matched through the model. Then we bring up the questions about what we want to know through data analysis:

- What are the rates of using arXiv in different CS sub-areas?
- What are the trend of the use of arXiv or what are the differences for researchers submitting their papers on arXiv across years?
- Which area use arXiv most pervasively?
- What other features are more predictive on the use of this preprint library?

Therefore, data analyzing is proceeded in the following three aspects based on the questions:

- The number of arXiv papers in different computer science sub fields
- The trends of different conferences papers submitted on arXiv
- The time about when researchers submit their papers on arXiv, before or after the conference deadline.

1.3 Paper structure

In this paper, we introduce the general idea about this project at first, mainly about the purpose of doing this project, the hypothesis we may think of and the results we get from data analysis.

Then we present the relevant background in chapter 2. We explain the advantages of open access and the reasons why some researchers use the preprint as well as why others do not. Also, the situation about the use of preprint in mathematics and physics is mentioned after that. At the same time, reasons are expounded about why we want to measure the use of arXiv in computer science research.

In chapter 3, we introduce the datasets of the project about what tools, software and programming language we used to collect data, what data we collected and how we collected as well as the way we use that. The datasets are mainly in two parts, one part is arXiv paper dataset and conference paper dataset, which are original data prepared for computing the similarities between two papers. We preprocess the data and after calculating the similarities by using the matching algorithm, we randomly choose a small number of data and label the data in person to see whether the two papers are matched or not. Therefore, the labeled data are used as the second part of the datasets. This dataset is prepared for training the model in order to get the weights of the similarity, so that we can apply this model on the first dataset to find out the matched paper pairs.

In chapter 4, a matching algorithm is designed for matching two papers (one from the conference and one from arXiv). The similarities are calculated for the paper pairs and there are two similarities for each pair: title similarity and authors similarity. After evaluating the performance of the matching algorithm by using the labeled data, the title and authors similarities can be treated as the input parameters of a model. The model building and model evaluating is described in chapter 5.

Then the data analysis is reported in chapter 6. Mainly about measuring the features of the arXiv data in computer science subject, and the trend of the use of arXiv for CS conferences. One aspect of the trend is measured from the number of papers that conferences may submit to arXiv. While another aspect is about the time, about when do researchers upload their papers on arXiv (before or after the conference deadline). Finally, conclusions about the results according to the data analysis have been made in the last chapter.

Chapter 2

Background

In early years, researchers can only submit their findings and research results to prestigious journals and highly selective conferences. But nowadays, the rapid development of Internet technology provides opportunities for researchers to present their ideas and research results much easier and quicker through many alternative ways. Open access is one of these ways, which allows scientists to submit their papers before publication and attracts more and more researchers using it.

Due to the low acceptance rate of the conference paper as well as the long period of peer-review and publish process, researchers are more likely to accept and use the open access. Accepting a paper for conference or journals is a binary decision and it is impossible to have any revision once the paper is submitted. Although many conferences have guaranteed to make decision faster, researchers have to wait for another conference in the same year or wait for the same conference in next year if their papers are rejected. In this situation, the open access shows its fascinating advantages. It is online, free of charge and removes the barriers of permission. Also, researchers can submit and revise their papers for multiple times. The researchers need to exchange, evaluate and preserve the new ideas and results. The process to show the idea is expected to be fast and less unnecessary limitation. Therefore, the preprint repository is an alternative way for scholar-communication which can greatly boost the speed of scientific development. For instance, one researcher develops a new method and submit the paper to the preprint repository, then he could apply it to keep on the related work. At the same time, other researchers can use the achievements he did as a reference to help themselves to solve the problems right away (Academia, 2014).

Many researchers complained about the double-blind reviewing system that slows down the progress of innovation and science. In a proposal for a new publishing model

in computer science, Yann LeCun pointed out that "High-selective conferences have biases on the innovation ideas and tend to favor incremental improvements on well-established methods. Hence many ideas suffer from the reviewing purgatory and might be known to others after years when they come to maturity, which makes conferences somewhat boring." (LeCun, 2013). Therefore, some leading researchers advocated using the open access, where they can share their thoughts and ideas in the same field or even cross different disciplines before the papers are peer-reviewed or published. This is a great way for the science communication without the limitations of time, field, price, permissions and the pressures of deadline.

ArXiv is one of the four major open access repositories, which contains over one million papers in lots of areas. It provides researchers opportunities to publish their works without peer-reviewed or accepting by the journals or conferences. Started in 1991, arXiv.org passed the half-million article milestone on October 2008, and hit a million by 2014 (Kling et al., 2004). The open access repositories are mainly dominated by physics, mathematics and computer science (Li et al., 2015). About four fifths of mathematic researchers deposit their achievements in arXiv (Fowler, 2011), as do three fourth of publishing condensed matter physicists (Moed, 2007). It shows that arXiv, as the preprint repository, is significant in natural science including physics and mathematics.

Besides so many advantages of arXiv, the following disadvantages are also emerged. Facing the openness and secrecy, scholars in different fields hold different opinions. Besides of the commercial value of the findings, the researchers compete for the priority of scientific achievements (Gaston, 1971). However, there is a conflict between the secrecy and scientific knowledge creation. In some subjects, sharing the newest information and communicating with others is indispensable. However, without peer reviewed, the quality of the published works on arXiv is spotty, so that researchers have to take time to find the valuable and related paper.

As the major open repository, arXiv provides the access to share the knowledge without technical and social barriers. However, in different disciplines and areas, great and obvious differences exist in the use of open access (Velden, 2013). The preprint servers suffer a disfavor in life sciences and chemistry, by contrast of the popularity in physics and mathematics (Kling et al., 2003, 2004). Velden showed the evidence that social and cultural arrangements of research fields as well as their needs play an important role in sharing new information and using open access. Jenny Fry and Sanna Talja applied Whitleys theory of the intellectual and social organization of academic

fields to explain differences in forms and types of papers across fields (Fry and Talja, 2007).

Not only in different research areas, there are also significant differences within the same area. The situations and motivations vary across different sub areas in publishing on specific journals or conferences and submitting papers on arXiv. Collins found great variety among the scientists that they interviewed. Some methods of finding, sharing and using information were common to researchers in many sub-fields of the physical sciences, but the relative importance of these methods varied. They studied how scholars found, used and disseminated information in science, especially physics, and discussed the differences for publishing, collaborations and sharing data varied by discipline, territory and career periods (Gaston, 1971; Vertesi and Dourish, 2011; Collins, 2015). Focusing on the behaviors of researchers using the arXiv as the preprint repository and considering the situations they are in, there are some interesting findings. Data derive the value not only from the context of utilize and exchange, but also from specific content of production, integral to the studies (Vertesi and Dourish, 2011; Velden, 2013). It is called "data economy" of which we need a great understanding. In physics field, the findings were carried out, which deepened understanding of how widespread behaviors within the community of physicists (Collins, 2015).

Computer science is viewed as one of the most important subjects in the arXiv. This paper aims to bring out some interesting findings in behaviors and priorities of computer scientists and patterns of using arXiv in computer science research. Compared with the important academic conferences, we focus on the differences of publication behaviors across disciplines in computer science.

Chapter 3

Datasets

In order to explore the features of the submissions on arXiv in computer science research area, we need to collect the arXiv papers in computer science field at first. Then we need some computer science top tier conferences in different sub-areas and get their accepted papers. In this chapter, we introduce in detail the datasets we used in this project. The datasets in our project are mainly divided into two parts, the first part is the arXiv and conference papers, which is used for computing the similarities of paper pairs (in one pair, one paper is conference paper and one paper is arXiv paper). While the second part is the labeled training dataset, which is randomly produced by the processed first part and then label the data according to the fact whether the paper pairs are really matched.

Firstly, we introduce the tools we used to collect data, and then is the detailed way of how to collect them. Next, after preprocessing, the first part of the datasets is well formed, and is ready to execute by using the matching algorithm. Finally, after matching the conference papers with the arXiv papers, we get paper pairs that contain similarity information. A small number of data are randomly chosen from the similarity paper pairs, and we labeled them to prepare for the evaluation of the algorithm and as the training data to build the model.

3.1 Tools

Python: the programming language has been chosen to do this data analysis project. It is widely used and supports object-oriented, imperative and functional programming. The syntax of Python allows programmers use fewer lines of codes to express the same concepts as C++ or Java. Most importantly, it meets the requirements of this project:

web data crawling, natural language processing, machine learning and statistical analysis. As one of the most popular programming languages for scientific computing, its scientific libraries and high-level interactive nature become the best choice for the algorithmic development and exploratory data analysis.

It integrates a lot of modules such as:

Scrapy: it is an open source and collaborative framework for extracting the data from websites. As we all know, websites have rich unstructured text source that can be mined and turned into useful data. This useful information can be extracting by web scraping, while using the scrapy is a very simple, fast but extensible way to achieve web harvesting. It is easy to setup, to use and has great documentation. The solutions are mature and focused and support to export data to CSV, XML, JSON (Hassan, 2016).

Xmlsax: a package provides a number of modules which implements SAX interface for Python. SAX (Simple API for XML) is an event-driven online algorithm for parsing XML documents. While there is another algorithm for parsing XML documents DOM (Document Object Model), it needs build a tree for XML file and load the whole file into memory. We have a large number of data, so SAX is chosen to parse the XML files since it just need load a small part of a file while parsing, though DOM is faster, it may cause out of memory error. Python provides this interface, so we use xml.sax package to solve the xml parsing problem. The make_parser() function of xml.sax and its ContentHandler make it easier and convenience to parse the XML files.

NLTK: (Natural Language Toolkit) is written in Python programming language for natural language processing (for English). As a leading platform, NLTK has been called “a wonderful tool for teaching, and working in, computational linguistics using Python,”and “an amazing library to play with natural language. It has many corpora”, lexical resources and provides a number of interfaces that are easy to use. Many text processing libraries for tokenization, stemming can be used for the matching algorithm. The way of using this toolkit to process the data is explained in chapter 4.

Scikit-learn: a Python module that has a wide range of simple and efficient tools for data mining and data analysis and integrates a great number of machine learning algorithms. Using a general purpose high-level language, it put emphasis on ease of use, performance, documentation and API consistency. This package is open source, builds on Numpy, SciPy and matplotlib, has minimal dependencies and distributed under BSD license, encouraging both academic and commercial use (Pedregosa et al., 2011). We will use this package and apply the logistic regression to build a model to

compute whether a paper-pair is matched or not, and this is described in chapter 5.

Numpy: the fundamental package for scientific computing with Python and the base data structure used for data and model parameters. It is very useful in the data training and model building.

Matplotlib: a python 2D plotting library which produces publication quality figures. This library makes it easy to generate plots, histograms, bar charts and so on. The interface pyplot achieves the function of Matlab, it is very convenience and useful to those who are familiar with Matlab.

OAI-PMH: (The Open Archives Initiative Protocol for Metadata Harvesting) provides an application-independent interoperability framework based on metadata harvesting. Metadata are the data that provide information about other data. A single metadata scheme may be expressed in a number of different markup or programming language, such as XML, HTML, RDF or plain text. For instance, XML, the metadata scheme is hierarchical, so the elements are nested and there are parent-child relationships between metadata elements. OAI-PMH is such a good tool and a preferred way to access metadata and harvest the collections of articles.

3.2 Data collection

3.2.1 ArXiv data

(1) Bulk data harvesting

Open access permits the computation of articles and human access to individual articles, while the results of the computation include tools to find, browse, use and access articles. Because of the practical and financial constraints, arXiv does not allow user to crawl data. However, the data access mechanisms provided are grouped into two parts: metadata and full-text.

In this project, we just need abundant article data to explore the features in the use of arXiv in computer science, so the full-texts are not needed. Due to this demand, metadata are better to collect. While the arXiv provides three kinds of bulk metadata access: OAI-PMH, API and RSS.

From table 3.1, the RSS feeds are intended primarily for human consumption, API is better for real-time access and OAI-PMH can access for all articles. The interface to API is quite simple, and the results can be obtained by using search_query. The prefixes in search_query field that can be searched contains title, author, abstract, cat-

Table 3.1: ArXiv bulk metadata access

Bulk Metadata Access
OAI-PMH
arXiv supports the OAI protocol for metadata harvesting (OAI-PMH) to provide access to metadata for all articles, updated daily with new articles. This is the preferred way to bulk-download or keep an up-to-date copy of arXiv metadata.
API
arXiv supports real-time programmatic access to metadata and our search engine via the arXiv API. Results are returned using the Atom XML format for easy integration with web services and toolkits.
RSS
arXiv provides RSS feeds of new updates each day. These are intended primarily for human consumption but do use well defined XML formats and thus might be useful to machine applications.

egory except date. Also, if use id to search, the number of results for one request is very limited. A request with max_results> 30000 will result in an HTTP 400 error. Actually, the problem is this API does not provide the exact total number of records we can get. Large results request will take a long time to render. We need collect all article information before 2016 and the number of papers is more than 100,000 in fact, so that we need request smaller slices for many more times. Also because we need collect the title, authors, submit time and category information, OAI-PMH interface is more suitable.

Arxiv supports and participates in the Open Archive Initiative (OAI), and is a registered OAI-PMH data provider and provides metadata for all submissions. The baseURL that arXiv supports is `http://export.arxiv.org/oai2`. Each article in arXiv is modeled as an item in the OAI-PMH interface, and each item is wrapped in a pair of record tags where all article information needed are nested. A record is returned as an XML-encoded byte stream in response to a protocol request to disseminate a specific metadata format from a constituent item. There are several metadata formats (oai_dc, arXiv, arXivRaw) for an item and all of them support all articles. The format examples are available in the arXiv help page. For convenience, the arXivRaw format has been chosen in order to get the first submission date for the later data analysis.

Each metadata item for OAI-PMH is associated with a datestamp, which is the last modification time of the article not the original submission time or replacement times for older articles, since arXiv updates metadata records in bulk on several occasions. Therefore, the OAI-PMH interface does not support selective harvesting based on submission dates, except for selective harvesting based on subject area (there are sets for match, cs, nlin, q-bio and sub-sets in physics archives). This means we can use cs set to get all articles in computer science field without a datestamp range request.

The verb that used in the request URL is ListRecords to harvest records form repository and the optional arguments for selective harvesting are set and metadataPrefix to harvest records in cs subject and in arXivRaw metadata format. The request URL is like following:

```
http://export.arxiv.org/oai2?
    verb=ListRecords&set=cs&metadataPrefix=arXivRaw
```

However, due to the flow control mechanism, the repository replies to a request with an incomplete list and a resumptionToken. The maximum number of records for one request is 1000. Hence, in order to get the complete list of records, a sequence of requests is needed to issue and must with resumptionToken argument except the initial request. Once the initial harvest has been completed, the copy is maintained by making incremental harvesting request, and the value of resumptionToken elements in the response can be used as the value of resumptionToken arguments in the subsequent request.

For example, the value of resumptionToken in the response by using the request URL above is:

```
<resumptionTokencursor="0"completeListSize="108745">1219662|1001
</resumptionToken>
```

This is the resumptionToken element with cursor and completeListSize attributes in the initial response. The value of cursor attribute means the start index of the record in the complete record list for this response. The value of completeListSize mens the total number of records in the complete record list. A non-empty value of resumptionToken has been returned, indicating that the list is incomplete and one or more request is needed to be issued in order to harvest complete list. And the value of resumptionToken element can be used in the next request like the following URL:

```
http://export.arxiv.org/oai2?
    verb=ListRecords&resumptionToken=1219662|1001
```

From this URL, we can see that in order to retrieve next portion of the complete list, the subsequent requests can just use verb of ListRecods and resumptionToken argument but without other arguments in the first request. While each value of resumptionToken can be obtained from the previous response. Because there are 108,745 records in the replied complete list, each response will get 1000 records in max. Then we can make sure that we need 109 requests in total, and the value of resumptionToken is certain that the number of the suffix of this value should increase 1,000 at each time while the prefix number remains the same.

(2) XML data parsing

All arXiv article metadata in computer science area have been harvested through the OAI-PMH interface are saved as XML files. Then for each article, the useful information wrapped in the metadata item can be extracted by parsing the XML files.

There are two types of XML parser for an XML document, one is DOM and another is SAX. DOM is the Document Object Model, which relies on two main concepts: a) The XML processor construct the complete XML document tree in memory; b) The XML application issues DOM library calls to explore and manipulate the XML tree, or to generate new XML trees. The advantages of DOM are that it is easy to use, once in memory, no tricky issues with XML syntax anymore. Also, all DOM trees serialize to well-formed XML, even after arbitrary updates. However, the biggest problem is it uses lots of memory.

While, SAX (the simple API for XML) processor need constant space, regardless of the XML document size. This XML parser supports users to build their own data structure. It provides a low level access to the data, which is in tag by tag and text-node by text node order. The document is read sequentially and once only. Moreover, SAX is an event-driven algorithm, where there is no need to build a tree data structure in the communication between the SAX processor and backend XML application. Therefore, SAX is memory efficient.

For a large amount of data needed parsing, the SAX parser is better to choose. It reports fine-grained parsing events for an XML document, such as startDocument, endDocument, startElement, endElement, characters and so on. The ContentHandler and the parse functions of the xml.sax package in Python makes it quite convenient to parse the metadata.

Figure 3.1 shows the one article record of the harvested CS arXiv data. The data for each record in XML file that we need are all wrapped in metadata element. The value of title, authors, categories elements can be obtained in the characters-event directly after corresponding startElement-event. The categories element provides the sub-area of computer science for the article. As for the first submit time, it is wrapped in the version element where the attribute is version=‘v1’ and we can get the submission time from the value of date element in this version element. Then the extracted data can be export to a text file and wait for processing.

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLoca
<responseDate>2016-06-12T14:19:13Z</responseDate>
<request verb="ListRecords" from="1993-01-01" metadataPrefix="arXivRaw" set="cs">http://export.arxiv.org/oai2</request>
<ListRecords>
<record>
<header>
 <identifier>oai:arXiv.org:0704.0002</identifier>
 <datestamp>2008-12-13</datestamp>
 <setSpec>cs</setSpec>
</header>
<metadata>
 <arXivRaw xmlns="http://arxiv.org/OAI/arXivRaw/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation
 <id>0704.0002</id><submitter>Louis Theran</submitter>
  <version version="v1"><date>Sat, 31 Mar 2007 02:26:18 GMT</date><size>377kb</size><source_type>D</source_type></version>
  <version version="v2"><date>Sat, 13 Dec 2008 17:26:00 GMT</date><size>136kb</size><source_type>D</source_type></version>
  <title>Sparsity-certifying Graph Decompositions</title>
  <authors>Ileana Streinu and Louis Theran</authors>
  <categories>math.CO cs.CG</categories>
  <comments>To appear in Graphs and Combinatorics</comments>
  <msc-class>05C85; 05C70; 68R10; 05B35</msc-class>
  <license>http://arxiv.org/licenses/nonexclusive-distrib/1.0/</license>
  <abstract>  We describe a new algorithm, the $(k,\ell)$-pebble game with colors, and use
it obtain a characterization of the family of $(k,\ell)$-sparse graphs and
algorithmic solutions to a family of problems concerning tree decompositions of
graphs. Special instances of sparse graphs appear in rigidity theory and have
received increased attention in recent years. In particular, our colored
pebbles generalize and strengthen the previous results of Lee and Streinu and
give a new proof of the Tutte-Nash-Williams characterization of arboricity. We
also present a new decomposition that certifies sparsity based on the
$(k,\ell)$-pebble game with colors. Our work also exposes connections between
pebble game algorithms and previous sparse graph algorithms by Gabow, Gabow and
Westermann and Hendrickson.
</abstract></arXivRaw>
</metadata>
</record>
```


Figure 3.1: One record of arXiv data

3.2.2 Conference data

(1) Conference list

After harvesting arXiv data, articles in conferences are needed to collect in order to compare how many conference papers have been submitted to arXiv. But the most important thing before collecting conference data is to determine which conferences are worth collecting or should be harvested.

There are too many conferences in computer science area, but the top tier conferences are more typical to explore the data features. Also, we need to measure the differences of the use of arXiv in different sub-areas. Hence, several conferences in different computer science sub-areas have been chosen from a set of A+ CS conferences.

As is show in table 3.2, these 17 famous A+ CS conferences are chosen from different sub-areas or sub-sub-areas. They are all held in every year and have accepted papers from 2006 to 2015. Other filtered conferences are because that they may be held in every two years or difficult to harvest data. AAAI, AAMAS, ACL, ICML and NIPS are all in artificial intelligence sub-area, but the latter three are in different sub-sub-areas. ACL is in natural language processing area while the ICML and NIPS are

Table 3.2: Chosen conference list

Category	Acronym	Name
AI	AAAI	National Conference of the American Association for Artificial Intelligence
AI	AAMAS	National Conference of the American Association for Artificial Intelligence
NLP	ACL	Association of Computational Linguistics
ML	ICML	International Conference on Machine Learning
ML	NIPS	Advances in Neural Information Processing Systems
Architecture	ASPLOS	Architectural Support for Programming Languages and Operating Systems
Architecture	ISCA	ACM International Symposium on Computer Architecture
Data Mining	SIGKDD	ACM International Conference on Knowledge Discovery and Data Mining
Database	SIGMOD	ACM Special Interest Group on Management of Data Conference
Graphics	SIGGRAPH	ACM SIG International Conference on Computer Graphics and Interactive Techniques
HCI	CHI	International Conference on Human Factors in Computing Systems
IR	SIGIR	ACM International Conference on Research and Development in Information Retrieval
PL	OOPSLA	ACM Conference on Object Oriented Programming Systems Languages and Applications
PL	PLDI	ACM SIGPLAN Conference on Programming Language Design and Implementation
Security	CCS	Architectural Support for Programming Languages and Operating Systems
Theory	FOCS	International Conference on Human Factors in Computing Systems
Theory	STOC	ACM Symposium on Theory of Computing

in machine learning area. Besides, other conferences are chosen from architecture, data mining, database, graphics, human-computer interaction, information retrieval, programming language, security and theory these different areas.

2Scrapers for data crawling ArXiv started from 1991, and not all conference papers were submitted to arXiv. The earlier, the smaller number of papers might be uploaded to arXiv. Therefore, it is reasonable to collect recent 10 years conference papers. For each conference, finding the main webpage URLs from 2006 to 2015, and crawling the accepted papers for each year. Because of the different HTML codes for conference in each year, 10 different spiders are needed to write for each conference.

Scrapy is used for writing the spiders to crawl the conference papers. Due to the information of accepted papers on webpage contains title, authors, an item of one paper should be generated that includes year, title, authors. The spider class in scrapy helps a lot to create spiders. The spider of each conference in each year has a unique spider name, and corresponding start_urls. After downloading the data through the URLs, a response object will be generated and delivered to parse function as the unique parameter. Then this parse function is in charge of parsing the response data, extracting information and yielding item.

For parsing and extracting data, the scrapy selector use XPath to select useful information from part of the HTML document. XPath is an expression language to be used

in another host language and is the most important XML query language used in many technologies (Abiteboul et al., 2011). It uses path expression to select nodes or node sets in an XML document, while it can be also used in HTML document. XPath allows the description of paths in a XML/HTML tree and the selected nodes are matched these paths. There are several types of nodes: document, element, attribute and text nodes. Document is the root node of the XML/HTML tree, but in scrapy spider, it is better to use the relevant paths of element nodes rather than the absolute paths. The attribute nodes or the values of attribute nodes can be used as the conditions of filters. Then, the exact text nodes contain the information that needed extract. The predicates and axes of XPath contribute to write the correct paths expression. Predicates (i.e. filter) are used to find the specific nodes or nodes that contain a specific value and an axis defines a set of nodes relative to the current node. The response data can be parsed and extracting required information from the HTML document through good XPath expressions.

After parsing the response data, the useful data are extracted and can be saved in the item container. Because scrapy supports export data to JSON, then we can save all these items into json files.

(3) Testing on a small data set There are too many scrapers that need to build, so the matching algorithm can test on a small data set after collecting a small number of conference data. In this way we can modify the algorithm much more quickly according to the errors or problems. After evaluating the algorithm that works well on the data, then we can collect more conference data and apply the algorithm to the full conference data. Besides, the deadlines of each conference from 2006 to 2015 should be collected at the same time for the latter data analysis.

3.3 Well-formed data set

In order to facilitate the latter matching algorithm, model building and data analyzing, as well as reducing the troubles of data inconsistency, the arXiv data and conference data need to be processed into uniform form before the follow-up works. Since the data are obtained from different sources, the information in the same item field may be expressed in different forms. That is to say, the data of arXiv papers may be expressed in different ways, and the data of different conferences papers may have different forms meanwhile different from the arXiv papers. For example, some papers provide the universities or institutes of authors but others do not. Thus, we need preprocess the data,

filter out the noise information and try to transform the data into a uniform format. Besides, all data have been saved in json files, each item in the json list is a dictionary that represents a paper. The fields of arXiv item data are title, authors, submit (submission time) and categories, while the fields of conference item data are title, authors, year. This well-formed data set is prepared for executing the matching algorithm to calculate the similarities of the arXiv paper and conference paper.

3.4 Labeled data

After computing the similarities of paper pairs, the data items that contain similarity information should be saved. The information of a new data item includes the arXiv paper data, conference paper data, title similarity and authors similarity. Then a number of random samples should be generated from the new data items.

In order to have a well-proportioned sample set, the new data items can be divided into 3 groups by the value of title similarities. The value of the similarity is not bigger than 1. The two titles are more similar, if the value of similarity closer to 1. If the value much smaller, the two papers are less likely the same one. Since most titles of conference papers have small changes if the paper was uploaded to arXiv, the papers are more likely unmatched papers if the title similarity is less than 0.5. Thus, the groups can be [0, 0.5], [0.5, 0.75], [0.75, 1], and a small number of samples can be randomly chosen from each group. So that we can guarantee that we have samples in all ranges rather than crowd in a small range or do not have data in a small range due to the complete random.

Choosing a conference from each sub areas, and for each chosen conference, generating samples from each group and exporting into sample files. Checking each paper pair, if the two papers are really matched, then label the item and insert a matched information into that item. This labeled dataset is used for examining the performance of the matching algorithm and then building the model.

Chapter 4

Similarity algorithm

4.1 Similarity algorithm design

4.1.1 Word vector

For two papers with title, authors and date information, if we want to know whether they are the same paper, things can be treated as checking whether the two strings are matched. Thus, an approach to compare two paper strings is processing the strings into word vectors at first, and then use a formula to compute the similarity between the two strings.

The first step is paper string processing. Two strings, one represents conference paper, and another represents arXiv paper. Converting every word in these two strings into lower case and then using NLTK to remove the stop words, tokenize them and word stemming.

Stop words are the words that we want to remove before performing a natural language processing task. For example, those words with high probability that may appear in most of the sentences can influence the similarities of two paper strings. NLTK provides a list of common stop words for English language as one of its datasets, and we need update this stop-words list by adding the common used punctuation characters that may appear in the title or authors of a paper.

Tokenization is the process of breaking a string into words, phrases, symbols or other meaningful elements. After tokenizing, the string convert to a list of words, there, we call it word vector.

A further step processes the word vector is word stemming. Words can appear with many different syntactic forms that change their morphology. For example, the verb

did is the past tense of do, and doing is the present progressive tense. If we want to count the frequency of the verb do, regardless of which form it takes, then we could normalize all forms to one common form before counting. One of the most common methods for English word stemming is Porter Stemmer, which can be applied to text by a method of NLTK.

Then, the word vectors are prepared well for calculating the similarities.

4.1.2 Similarity measures

Computing the similarity of two papers is now converting to compute the similarity between two word vectors. We use the idea of Jacca rd similarity coefficient, which is a statistic method for comparing the similarity and density of sample sets. It measures the similarity between set A and B, and is defined as the intersection of A and B divided by the union of these two sets:

$$J(A,B) = \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcap B|}{|A| + |B| - |A \bigcap B|}, \tag{4.1}$$

$$0 \leq J(A,B) \leq 1 \text{ (If A and B both empty, we define } J(A,B) = 1)$$

There are two word vectors, conference word vector $Con = (c_1, c_2, c_3,)$ and arXiv word vector $Arx = (a_1, a_2, a_3,)$. The similarity between Con and Arx can be measured by using the same idea. Counting the number of conference words that appear in the arXiv word vector and then dividing the total number of words that just contains co-occurrence words once:

$$Similarity(Con,Arx) = \frac{|Con \bigcap Arx|}{|Con \bigcup Arx|} = \frac{len(t_1,t_2,...)}{len(Con) + len(Arx) - len(t_1,t_2,...)}, \tag{4.2}$$

$$0 \leq Similarity(Con,Arx) \leq 1$$

where t_i is the co-occurrence word, $(t_1,t_2,...)$ is the co-occurrence word vector, $len()$ is the function that compute the length of a vector. Also, if the similarity is much closer to 1, then these two papers are more similar, while if the similarity is much closer to 0 which means they are more likely the different papers.

4.1.3 Authors and title similarities

If a paper string that uses for comparing contains both its title and authors, a situation may exist that if two papers are not the same in fact and the authors of two papers are the same and contain many words, but the titles have very small number of words, then the similarity will be very high and these two papers will be thought as the matched pair. In order to avoid this problem, it is necessary to separate title and authors, and thus, two types of similarity are needed for a pair of papers: one is title similarity and another is authors similarity. These two similarities can be used as the inputs of the matching model.

The process of calculating the similarities is:

1) Focusing one conference paper, computing title similarities of this conference paper with all arXiv paper in the dataset.

2) Choosing top 10 paper pairs sorted by the value of the title similarities in a descending order.

3) Computing authors similarities for these 10 paper pairs and choosing the top 1 paper pair.

4) Implementing the first three steps for every conference paper, then we will get all pairs for conference papers eventually.

4.2 Similarity algorithm evaluation

Using the labeled dataset, controlling the thresholds of title and authors similarities, and then plotting P-R curves to show the performance.

P-R curve is Precision-Recall curve,

$$Precision = \frac{TP}{TP+FP} \tag{4.3}$$

$$Recall = \frac{TP}{TP+FN}, \tag{4.4}$$

where TP=True Positive, FP=False Positive and FN=False Negative. Precision and Recall are inversely related, i.e. as precision increases, recall decreases and vice versa. The trade-off between precision and recall can be observed using P-R curve.

Figure 4.1, 4.2 and 4.3 are the P-R curves on NIPS, FOCS and SIGMOD respectively. They are in different sub areas, and the curves in each figure are changing on title thresholds for 6 fixed authors thresholds (the “a” in the title of each subplot means the authors threshold). In order to get higher precision and higher recall at the same

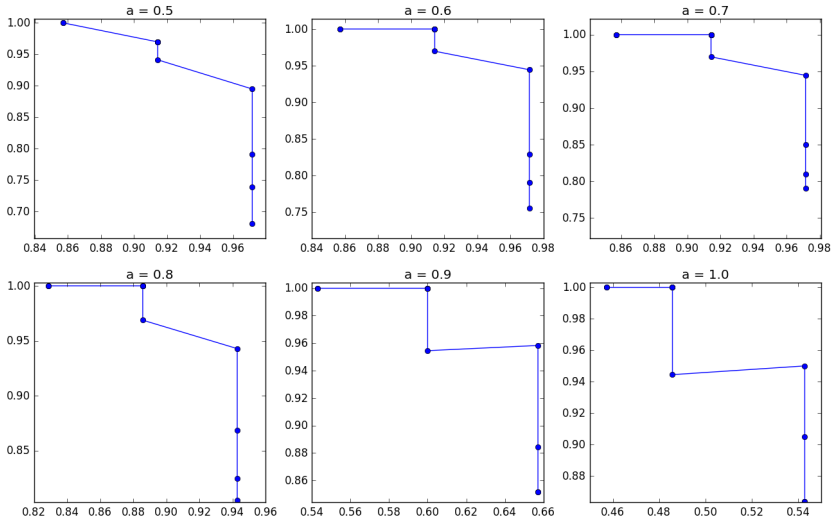

Figure 4.1: P-R curves on NIPS

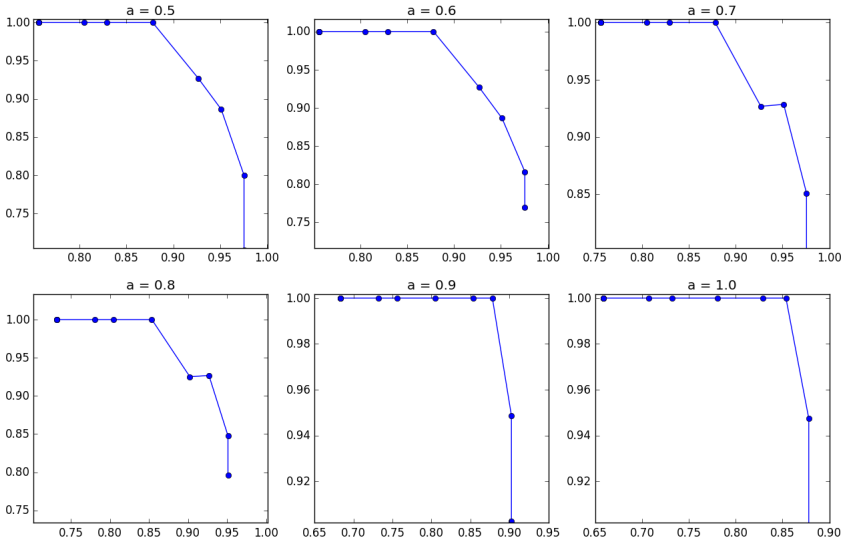

Figure 4.2: P-R curves on FOCS

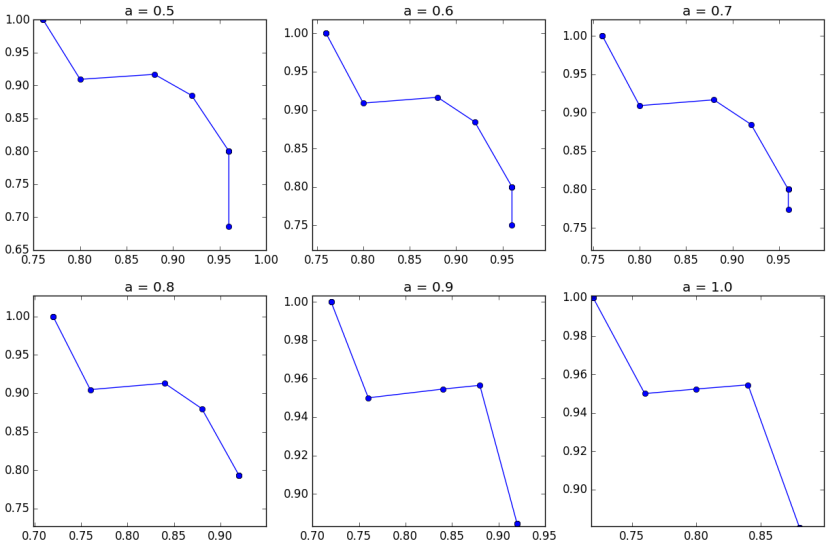

Figure 4.3: P-R curves on SIGMOD

time, we found that in each figure, the point at the knee of the plot when a=0.7 can reach an appropriate balance between precision and recall. The balance point lets precision bigger than 0.9 and recall bigger than 0.85, where the title thresholds are all around 0.8 for different conferences, though they are belonging to different sub areas.

Chapter 5

Logistic paper matching model

In the evaluation of the similarity algorithm, the thresholds change in discrete values and different conferences or sub areas have different thresholds, though they are generally consistent. A better way is to build a matching model, the title and authors similarities of a paper pair are the input arguments for this model, while the output result is binary 0 or 1, where 0 means they are not the matched pair and 1 means they are matched. This is a two class discriminative-classification task, and the number of input arguments is two, hence, we choose logistic regression method to build the model. Using the labeled dataset and separate data into two parts, one for training and one for testing. Using the training data to train the model in order to get the weights of title and authors similarities. Then apply the weights to the test data in order to evaluate the performance of the model. After that, we can use this model to compute and predict the class result of the whole well-formed dataset with two similarities.

5.1 Paper matching model

5.1.1 Logistic regression

Logistic regression is used to describe the relationship between a binary variable and one or more independent variables. Consider a binary output variable and we want to model the conditional probability $P(y = 1|\mathbf{x}) = g(\mathbf{x};\mathbf{w})$ as a function of $\mathbf{x}$, g must be between 0 and 1, and any unknown parameters w in g are to be estimated by the maximum likelihood. While the logistic (or sigmoid) function $g(a) = \sigma(a) \equiv \frac{1}{1+exp(-a)}$ provides a means for this, and consider the linear regression $a = f(\mathbf{x}) = b + \mathbf{w}^T\mathbf{x}$, then $p(y = 1|\mathbf{x}) = \sigma(f(\mathbf{x})) = \sigma(b + \mathbf{w}^T\mathbf{x})$. The linear regression plus a logistic trick (use of

sigmoid squashing) becomes logistic regression which gives us a linear classifier. The decision boundary is $b + \mathbf{w}^T\mathbf{x} = 0$ where $p(y = 1|\mathbf{x}) = p(y = 0|\mathbf{x}) = 0.5$.

The bias parameter b shifts the position of the hyperplane, and the direction of vector w affects the angle of the hyperplane.

Assume data is independent and identically distributed. For parameters θ, the log likelihood is $logp(D|\theta) =$

$$\sum_{n=1}^{N} y^n logp(y = 1|\mathbf{x}^n) + (1 - y^n)log(1 - p(y = 1|\mathbf{x}^n)) \tag{5.1}$$

Then we can calculate the gradients of the log likelihood and use gradient descent to find the maximum. We obtain the parameters w and b.

5.1.2 Training the paper matching model

The input vector $\mathbf{x} = (a_{sim}, t_{sim})$ where a_sim is the authors similarity and t_{sim} is the title similarity. Weight vector $\mathbf{w} = (w_1, w_2)$, bias parameter is b and output variable y is 0 or 1. Thus, $p(y = 1|\mathbf{x}) = \sigma(w_1 * a_{sim} + w_2 * t_{sim} + b)$.

Scikit-learn provides LogisticRegression module to build the logistic regression classifier. The parameter C is the inverse of regularization strength, smaller values specify stronger regularization and we set it as 1.0.

Using 80 percent of the labeled data as the training dataset, inputting authors and title similarities, then the provided function will train the model and return the weights and bias parameters. Saving this model and applying it to the whole well-formed dataset.

5.2 Matching model evaluation

We first use cross validation to evaluate the paper matching model. Cross validation is a model evaluation method which gives us an indication of how well the classifier will do. There are several methods to do the cross validation, while we choose the 5-fold cross validation method. The training dataset is divided into 5 subsets. Each time, one of 5 subsets is used as the test set and the other are put together to form a training set. Repeating 5 times, the average error across 5 trials is computed. The accuracy of this 5-fold cross-validation evaluation method is 98.23%.

As 80 percent of the labeled data are used as the training data, the rest 20 percent labeled data can be used as the test data. Applying the model trained by the training

data to test dataset, the accuracy we got is 97.37%.

The accuracies of the evaluation show that this classifier has a really good performance. Then we can use this classifier to predict whether the paper pair is matched and analyze the data.

Chapter 6

Data analysis

In this chapter, we analyze the data mainly from three aspects:

- Extracting features from the number of arXiv papers in different sub areas.
- Analyzing the trend of different conference papers submitted on arXiv:

  - The trends of the total number of papers submitted on arXiv during recent 10 years for different conferences.

  - The trends of the number of papers submitted on arXiv for different conferences.

  - The trends of number of papers submitted on arXiv for different groups (different sub-fields).

- Finding out when do researchers submitted their conference papers to arXiv, and analyzing the features of the time gap between submitted date and the conference deadlines:

  - The features based on different conferences

  - The features based on different years

6.1 Features of CS papers on arXiv

Figure 6.1 shows the number of arXiv papers submitted in different sub areas in an ascending order from the start of arXiv to 2015. The x axis presents the CS categories on arXiv and y axis is the number of papers. The lowest 5 categories are GL (General Literature), OS (Operating Systems), SD (Sound), GR (Graphics), ET (Emerging

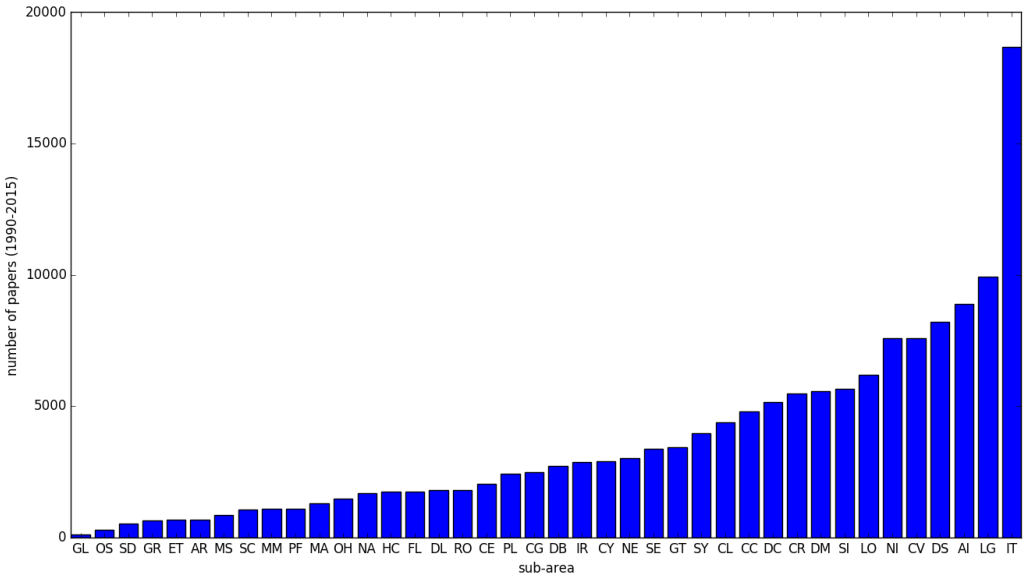

Figure 6.1: Number of arXiv papers in different sub areas

Technologies). These are the categories developing for many years but do not have breakthroughs in recent decades. The highest 5 categories are IT (Information Theory), LG (Learning), AI (Artificial Intelligence), DS (Data Structures and Algorithms), CV (Computer Vision and Pattern Recognition). These categories have great progresses in recent years because of the rapid development of the computer hardware and distributed systems as well as the fast speed of computer computation. Therefore, a great number of researches have been taken in these sub areas, which leads to occurrence of a large number of papers on these researches.

6.2 The trend of different conference papers submitted on arXiv

6.2.1 The rates of using arXiv for 10 years among different conferences

During the recent 10 years, the total accepted papers, the number of papers submitted on arXiv and the proportion of the matched papers over published papers for different top tier CS conferences are listed in table 6.1 While the figure 6.2 is the bar chart which shows the 10 years number of matched papers and accepted papers of different conferences obviously.

Table 6.1: Overall arXiv usage situation in recent 10 years for different conferences

Conference	Number of published papers	Submit on arXiv	Proportion
ISCA	452	1	0.22%
SIGGRAPH	1001	4	0.40%
PLDI	461	11	2.39%
SIGIR	843	11	1.30%
OOPSLA	437	13	2.97%
ICSE	1205	15	1.24%
CHI	3249	16	0.49%
SIGMOD	950	31	3.26%
CCS	741	32	4.32%
ACL	2294	70	3.05%
SIGKDD	1250	83	6.64%
AAMAS	2188	92	4.20%
AAAI	3597	203	5.64%
FOCS	746	330	44.24%
STOC	853	338	39.62%
ICML	2037	548	26.90%
NIPS	3073	607	19.75%

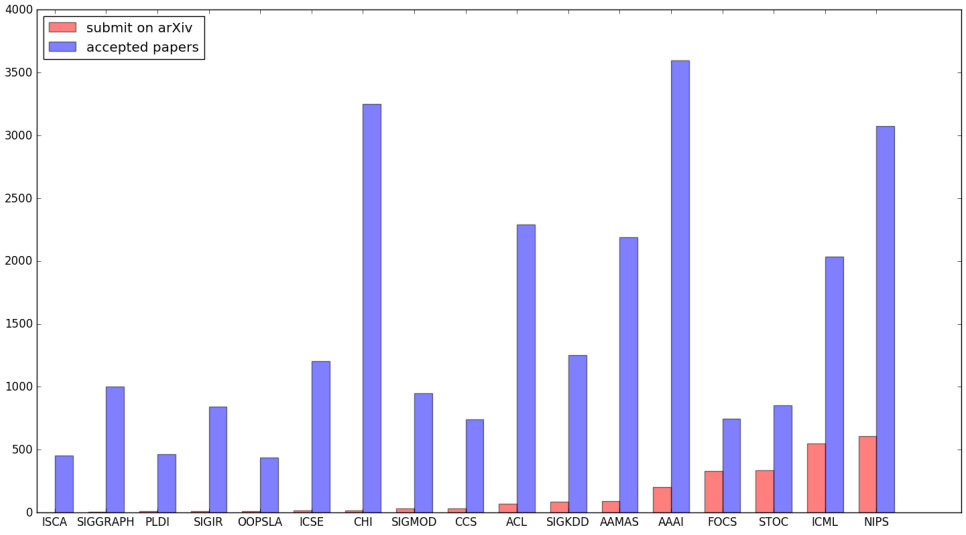

Figure 6.2: The number of matched papers and accepted papers for different conferences

From the bar chart and the digits of the proportion, we can see that FOCS and STOC have nearly two fifths of the accepted papers submitted on arXiv. About one fifth of the ICML and NIPS accepted papers are submitted on arXiv. SIGMOD, CCS, ACL, SIGKDD, AAMAS and AAAI have 3% - 6% published papers submitted on arXiv in recent ten years, while the proportions of other conferences are under 3%.

This phenomenon provides the evidence that researchers of computer theory are more willing to use arXiv and the use of researchers in machine learning area is in the second place. Next are the fields of AI, data mining, database and computer security. However, arXiv is not that popular in other fields of CS.

The computer theory is much closer to physics and mathematics and the arXiv is quite popular in these two subjects, which affects the researchers in the field of theory, so this might be the reason of why researchers in this field like using arXiv. Machine learning becomes one of the most popular directions in recent years, so that many more studies are underway or have been done.

6.2.2 The trend of the arXiv using rates for different conferences

Table 6.2 shows the rates (proportions: matched papers over published papers) of using arXiv for different conferences in recent 10 years, and the figure 6.3 presents the trends of the arXiv using. In order to show the trends clearly, the lines in the bottom in figures are plotted in three other separate plots.

Table 6.2: The rates of using arXiv for different conferences in each year

Conference	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015
ISCA	0	0	0	0	0	0	0	0	2.17%	0
SIGGRAPH	0	0	0	0	0	0	0	0	0.79%	2.54%
PLDI	0	0	0	0	2.44%	1.82%	2.08%	8.70%	3.57%	3.39%
SIGIR	2.63%	0	1.18%	0	1.14%	0	3.06%	0	1.22%	4.29%
OOPSLA	0	0	0	0	0	4.92%	5.08%	2%	1.92%	9.43%
ICSE	1.08%	0	0	0	1.03%	1.16%	0.67%	2.58%	3.03%	1.71%
CHI	0	0.55%	0.46%	0.36%	0.33%	0.24%	0.27%	0.26%	0.43%	1.45%
SIGMOD	0	1.08%	2.08%	0	3.06%	0	1.52%	7.69%	7.38%	6.40%
CCS	0	0	7.84%	0	1.85%	5%	2.47%	6.67%	6.14%	6.25%
ACL	1.36%	1.53%	0.84%	0.47%	1.30%	1.03%	0.87%	1.22%	4.56%	12.23%
SIGKDD	0.85%	0	2.54%	2.16%	2.97%	4.30%	6.02%	8.80%	12.75%	19.25%
AAMAS	0.38%	2.46%	1.97%	4.65%	1.84	4%	4.61%	6.15%	6.33%	7.35%
AAAI	1.56%	1.91%	3.09%	0	2.88%	4.65%	3.92%	7.43%	8.02%	11.87%
FOCS	13.56%	7.14%	18.99%	36.99%	48.15%	49.41%	63.29%	59.49%	60.87%	65.12%
STOC	10.13%	15.38%	9.76%	20.25%	37.18%	52.38%	52.81%	51%	62.64%	70.97%
ICML	1.43%	1.33%	4.43%	5.63%	8.18%	12.50%	91.09%	24.38%	31.29%	38.51%
NIPS	0.98%	2.76%	4.40%	7.25%	9.93%	14.05%	17.93%	26.39%	35.04%	47.64%

The features of the digits in table 6.2 and the trends depicted in the figure 6.3, 6.4, 6.5 and 6.6 can be extracted based on the current list of conferences:

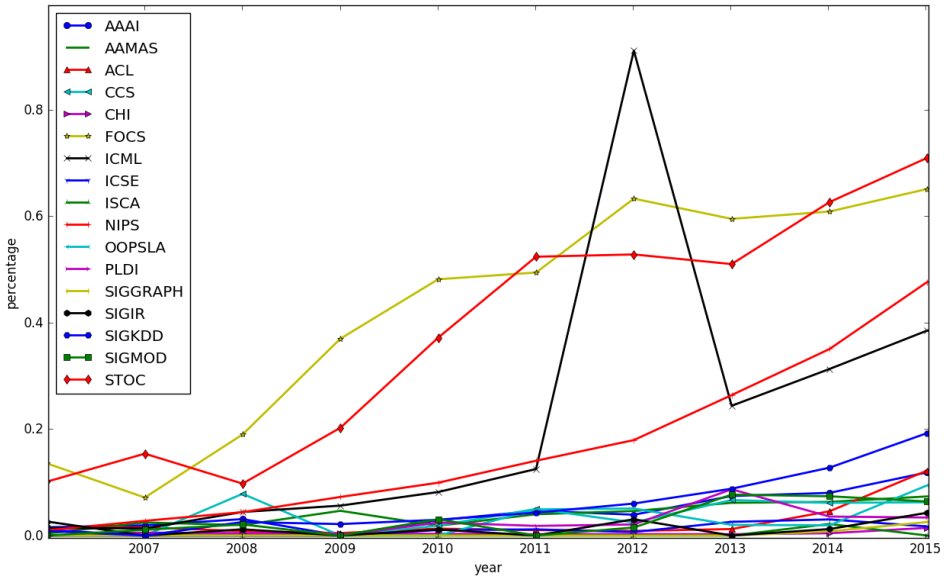

Figure 6.3: The number of matched papers and accepted papers for different conferences

a) Researchers in computer architecture and graphics just start using arXiv from 2014, which is reflected at least by the data of ISCA and SIGGRAPH in table 6.2 (more reliable results should be drawn by collecting more data in these CS sub areas).

b) As is shown in figure 6.3, FOCS and STOC, NIPS and ICML (expect 2012, it is explained by figure 6.7 and figure 6.8 have the similar trends and take the first and second places respectively, which are consistent with the results of the recent 10 years total proportions. All of them are increasing year by year, while the NIPS and ICML are increasing in an exponential trend where the growth rate is much bigger than that of FOCS and STOC.

c) In figure 6.4, AAAI, AAMAS and SIGKDD are all increasing exponentially. They have the similar trend from 2006 to 2011, but from 2012, SIGKDD increases much faster than AAAI and AAMAS. ACL keeps a low and relatively steady arXiv using rate for a long period from 2006 to 2013, but with an extremely high increasing rate after 2013.

In the same field AI, different sub fields have different trends, but same sub fields have similar trends. For example, NIPS and ICML are similar, AAMAS and AAAI are similar, but machine learning (NIPS, ICML), AI (AAMAS, AAAI) and NLP (ACL) have different trends, where arXiv is more popular in machine learning than in natural language processing.

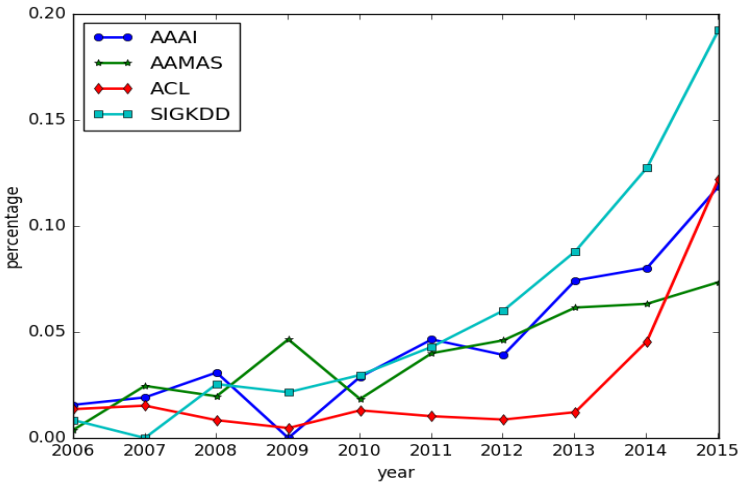

Figure 6.4: Trends of AAAI, AAMAS, ACL and SIGKDD

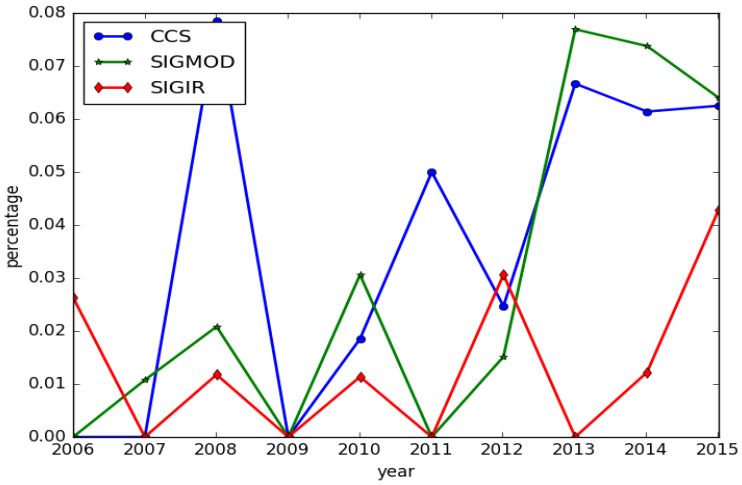

Figure 6.5: Trends of CCS, SIGMOD and SIGIR

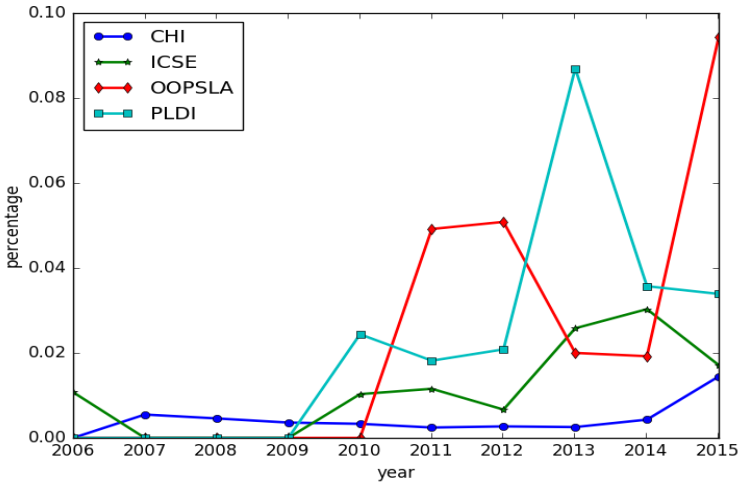

Figure 6.6: Trends of CHI, ICSE, OOPSLA and PLDI

d) Figure 6.5 shows the trends of CCS, SIGMOD and SIGIR, and they all have a vibrated feature, that they will get a peak value in every two or three years. The peak values will be in an increasing trend over years.

e) Though the using rates of conferences in figure 6.6 are all under 10% so as figure 6.5, they still have an increasing trend in the use of arXiv. CHI has a very steady and low using rate from 2007 to 2014, but has a smaller increase in 2015. The arXiv using rates of ICSE, OOPSLA and PLDI all start rising from 2009, 2010 and are in fluctuated increasing trends. ICSE is under the PLDI, and OOPSLA is a phase later than PLDI although they are both in programming language field.

dates	number of matched papers
2011-06	3
2011-08	1
2011-09	2
2011-10	1
2011-11	1
2011-12	1
2012-02	5
2012-03	4
2012-04	2
2012-05	14
2012-06	191

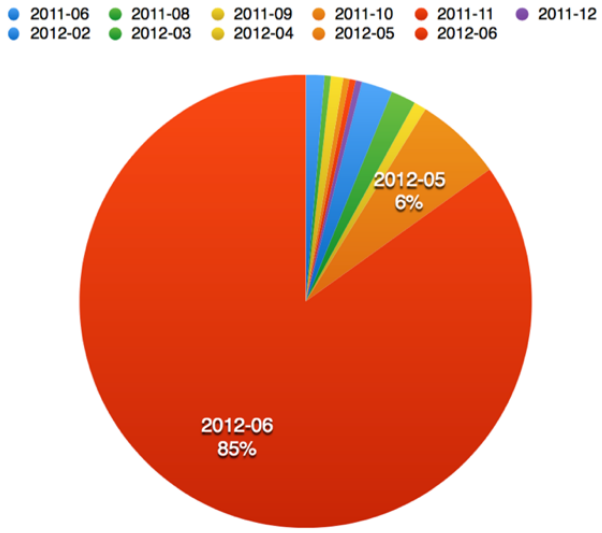

Figure 6.7: The dates of the ICML 2012 submissions on arXiv

In order to explain the data of ICML 2012, we extract the all the submitted time of arXiv matched papers of ICML 2012. The date of ICML main conference in 2012 is from June 27th to 29th, and figure 6.7 that 85 percent of the matched papers were submitted on June of 2012 and 6 percent on May of 2012, i.e. more than 90 percent of the submissions were uploaded on May or June in 2012. The further information of the June submissions extracted in figure 6.8 indicates that almost all were submitted on one of the two dates: 2012-06-18 and 2012-06-27, which are all before or on the first day of the main conference. This point maybe because that papers were uploaded on arXiv together by some official reasons.

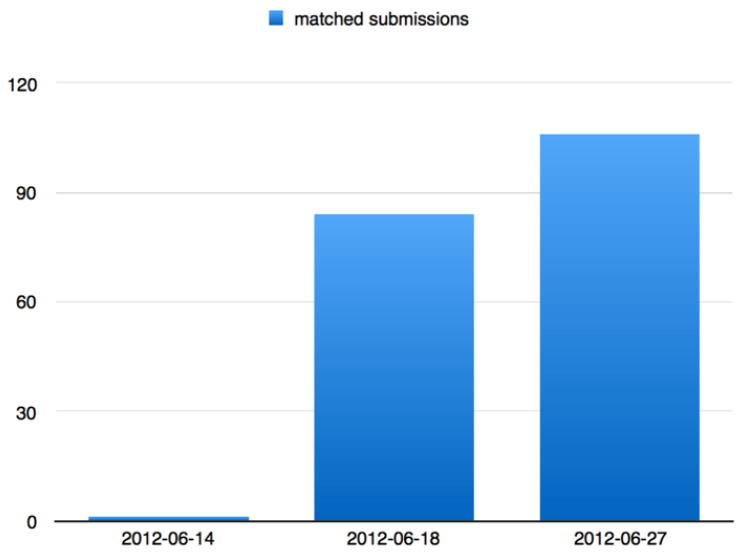

Figure 6.8: The number of arXiv submissions on June 2012 of ICML

6.2.3 Trend of groups

Then we analyze the trend in different categories where we group the conferences into 8 categories (AR, AI, CG, CS, DM, HCI, SE and TH). The categories of conferences are listed in table 6.3

Table 6.3: Groups of conferences.

Class	Conference
Architecture (AR)	ISCA
Artificial Intelligence (AI)	AAAI, AAMAS, ACL, ICML, NIPS
Computer Graph (CG)	SIGGRAPH
Computer Security (CS)	CCS
Data Mining (DM)	SIGMOD, SIGKDD, SIGIR
Human-Computer Interaction (HCI)	CHI
Software Engineering (SE)	OOPSLA, PLDI, ICSE
Theory (TH)	FOCS, STOC

Figure 6.9 depicts the trends of the arXiv-using rates for different categories. Due to the influence of physics and math, the arXiv is preferred by the researchers in computer theory than researchers in other CS sub areas. Affected by the popularities of deep learning, data science in recent years, the artificial intelligence and its sub-fields

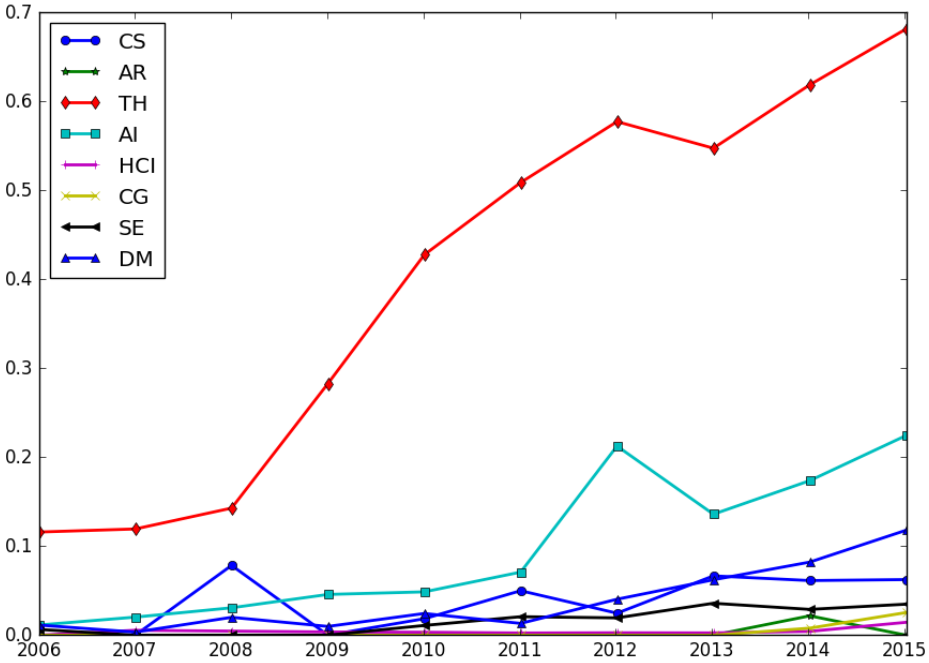

Figure 6.9: Trends of the rates for different groups

as well as data mining related fields have more and more researches, which boosts the use of arXiv.

6.3 The time of CS researchers submit their papers on arXiv

6.3.1 Comparison by conferences

We get lists of paper submission deadlines in every year for different conferences, and then measuring the time differences about when researchers submit their papers on arXiv, before or after the deadlines. ISCA and SIGGRAPH are ignored because their arXiv submissions are less than 5 over 10 years.

Figure 6.10 plots the probability distributions of the time differences compared by conferences. Each subplot is a distribution of a conference, y axis represents the probabilities, while numbers on the x axis represents the differences between the time submitted on arXiv and the conference deadlines, where negative number means how many months earlier than the deadlines, the positive numbers means the months later than the deadlines, and 0 means the month of the submit time is the same as the deadline. They are the discrete distributions, the shape of most of the distributions are sim-

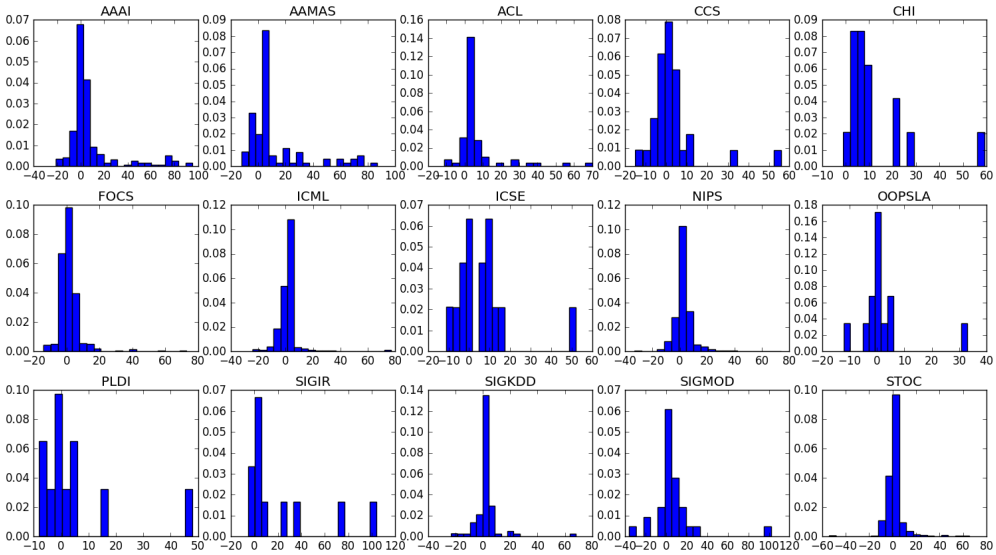

Figure 6.10: Trends of the rates for different groups

ilar, and the center of the distributions shows the time gap that most of the researchers submitted on arXiv.

Table 6.4: Average time difference for different conferences

Conference	STOC	FOCS	OOPSLA	ICML	NIPS	SIGKDD	CCS	PLDI
Mean	1.16	1.33	2.0	2.17	2.29	2.65	3.0	5.09
Conference	SIGMOD	ICSE	ACL	AAAI	AAMAS	CHI	SIGIR	
Mean	6.19	6.67	7.01	9.03	12.09	12.38	22.82	

Table 6.4 lists the average time differences for different conferences. If the mean is positive, that means the right side of 0 is longer than the left side, and vice versa. If the value of mean is 0, then it means that the distribution is center symmetric. The degree of the value illustrates the length of the tails and also about how long most of them submit their papers on arXiv before (negative value) or after (positive value) the deadline. If the value is much closer to 0, then the center of the distribution is much closer to 0.

6.3.2 Comparison by year

Figure 6.11 plots the probability distributions of time difference like fig. 6.10, but it is compared by years from 2006 to 2015. The shape of the distribution change from long right tail to long left tail. This phenomenon indicates that in early years, most of the

researchers submit their papers on arXiv a long time after the deadline. This time gap is curtailed over years, and became the long left tail gradually. However, the left tail will not increase as long as the early right tails, because researchers may not at a too early time to arXiv paper before deadlines.

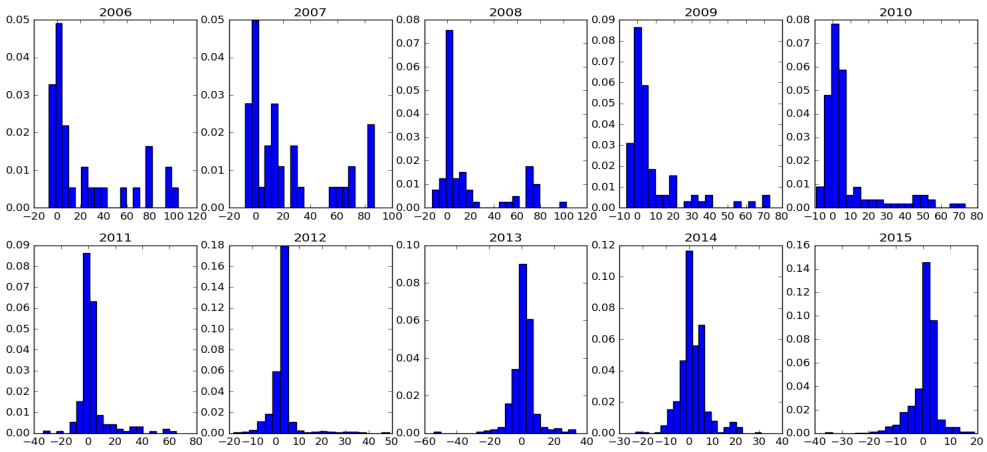

Figure 6.11: Probability distributions of time difference between arXiv submission time and conference deadlines (compared by years)

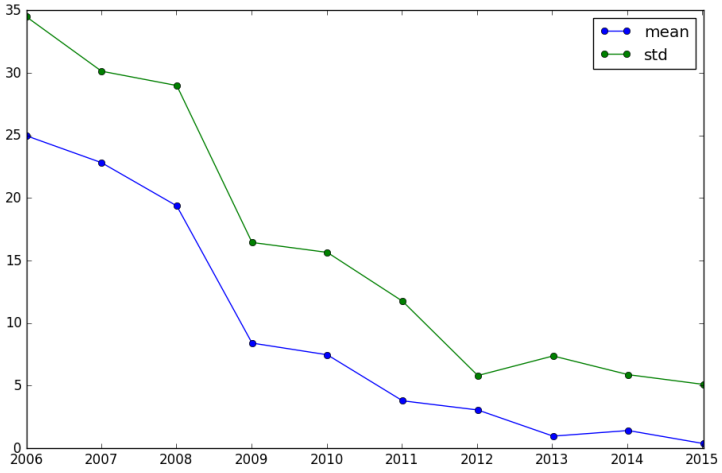

Figure 6.12: The trends of mean and standard deviation

The trends of mean and standard deviation over years are shown in figure 6.12 and both of them decrease. The dropping mean shows that the center of the distribution is getting closer to 0, while the declining std indicates the shape of the distribution will become tighter and tighter, which means most of the researchers will arXiv paper on the center time.

Chapter 7

Conclusion

The preprint repositories become more and more popular for scientists, and because of the popularity in the use of open access, many studies have been done on the use of open access in various subject fields, such as physics and mathematics. ArXiv is one of the most important open access, and has an increasing number of submissions every day. This paper aims to measure the behaviors of computer science researchers in the use of arXiv. In order to achieve the objectives, the following contributions have been made:

- We collect all arXiv papers through the OAI-PMH interface provided by arXiv in computer science subject from the start of arXiv to 2015. After parsing the XML files, we extract useful information from metadata. Then the differences and features of the submissions in different sub-areas of computer science have been drawn. From the highest number of submissions in theory, data science, artificial intelligence and computer vision, we can know that there are great progress and breakthroughs in these fields, while development of other fields with very small number of submissions may be in a stagnated period.

- Not only do different subjects have different features in the use of arXiv, but there are differences among different sub areas in the same subject. In order to find the evidence of the arXiv using features in different sub areas, we decide to choose a list of top tier computer science conferences in different sub areas and measure the number of conference papers had been submitted on arXiv. Then we can know the features of the behaviors of researchers in different fields. Writing scrapers to harvest the accepted papers for conferences in recent decade, and processing the data into an uniform format.

- Aiming at the conference paper, designing a similarity algorithm, and calculating the title similarities between a conference paper and every arXiv paper by using the algorithm. Next computing the authors similarity for the top 10 paper pair in title similarity and selecting the top 1 authors similarity paper pair. Then We get title similarity and authors similarity for each paper pair that constitutes the dataset we need to measure.

- Producing a random sample from the dataset, then check the sample whether the paper pairs are matched and label the matched data. This labeled data set can be used to evaluate the performance of the similarity algorithm. Also it can be used to build the matching model.

- The two similarities are the inputs of the logistic matching model, while the output is a binary variable (1 means matched, 0 means not matched). Using the labeled dataset, separating it into two parts, one for training the model, another is as the test dataset to evaluate the matching model.

- Applying the model to the whole dataset, then we can get matched submissions on arXiv for different conferences. The overall ratio of arXiv submissions and accepted papers for conferences in the recent ten years shows that arXiv is more popular in computer theory and machine learning. The computer theory has the highest using rate that may be because this sub area closes to physics and math, then it affects the arXiv using behaviors of researchers.

- Getting the proportions of arXiv submissions over accepted papers for each conference and plotting the trends over years. Different conferences in the same field may have similar trends (such as FOCS and STOC). In the same field, for example AI, different sub-fields may have different trends (such as machine learning and natural language processing).

- Grouping the conferences and finding out the features from the trends of groups. Besides the computer theory, the popularities of the deep learning and data science boost the use of arXiv in artificial intelligence and data mining related areas.

- Collecting the deadlines for conferences in each year, then the time differences between the submission time and deadline can be analyzed. We get a group of probability distributions for each conference as well as a group of probability distributions for each year. The distributions change in different conference, and

also they change in different year. The shape of the distribution change from long right tail to center symmetric then to long left tail over years, which indicates more and more researchers submit papers on arXiv before the conference deadlines but may not be too early. The trends of the mean and standard deviation provides evidence about the changes of the time differences and can also be used to predict the subsequent mean and standard deviation that affects the shape of the distribution.

We believe that further analysis can be done base on our works. More interesting findings and important experiments might be triggered by the current analysis on measuring the use of arXiv in computer science research through matching the top tier conference papers with arXiv papers. In the future, we can analyze the arXiv submissions by a low level conference. Comparing the features of the arXiv use on high level and low level venues. Then measuring the differences of behaviors of the researchers in different levels.

Bibliography

Abiteboul, S., Manolescu, I., Rigaux, P., Rousset, M.-C., and Senellart, P. (2011). *Web data management*. Cambridge University Press.

Academia (2014). Why upload to academic preprint sites like arxiv? `http://academia.stackexchange.com/questions/16832/why-upload-to-academic-preprint-sites-like-arxiv`.

Collins, E. (2015). Information practices in the physical sciences. a quantitative study to evaluate the impact of digital technologies on research in the physical sciences. Technical report.

Craig, I. D., Plume, A. M., McVeigh, M. E., Pringle, J., and Amin, M. (2007). Do open access articles have greater citation impact?: a critical review of the literature. *Journal of Informetrics*, 1(3):239–248.

Fowler, K. K. (2011). Mathematicians' views on current publishing issues: A survey of researchers.

Fry, J. and Talja, S. (2007). The intellectual and social organization of academic fields and the shaping of digital resources. *Journal of information Science*, 33(2):115–133.

Gaston, J. (1971). Secretiveness and competition for priority of discovery in physics. *Minerva*, 9(4):472–492.

Hassan, K. (2016). Extracting data from websites using scrapy. https://kaismh.wordpress.com/2016/04/29/extracting-data-from-websites-using-scrapy/.

Kling, R., McKim, G., and King, A. (2003). A bit more to it: scholarly communication forums as socio-technical interaction networks. *Journal of the American Society for Information Science and Technology*, 54(1):47–67.

Kling, R., Spector, L. B., and Fortuna, J. (2004). The real stakes of virtual publishing: The transformation of e-biomed into pubmed central. *Journal of the American Society for Information Science and Technology*, 55(2):127–148.

LeCun, Y. (2013). Proposal for a new publishing model in computer science. `http://yann.lecun.com/ex/pamphlets/publishing-models.html`.

Li, X., Thelwall, M., and Kousha, K. (2015). The role of arxiv, repec, ssrn and pmc in formal scholarly communication. *Aslib Journal of Information Management*, 67(6):614–635.

Moed, H. F. (2007). The effect of 'open access' on citation impact: An analysis of arxiv's condensed matter section. *Journal of the American Society for Information Science and Technology*, 58(13):2047–2054.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Suber, P. (2007). Open access overview. *Retrieved from Peter Suber's website: http://www. earlham. edu/˜ peters/fos/overview. htm*.

Tennant, J. (2016). What if you could peer review the arxiv?

Velden, T. (2013). Explaining field differences in openness and sharing in scientific communities. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 445–458. ACM.

Vertesi, J. and Dourish, P. (2011). The value of data: considering the context of production in data economies. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 533–542. ACM.