Automatic Reviewer

### Predicting Whether a Paper will be Accepted or Rejected

### CS 521 Statistical Natural Language Processing, Fall 2012

Hong Wang ([hwang207@uic.edu](mailto:hwang207@uic.edu)) Shuyang Lin ([slin38@uic.edu](mailto:slin38@uic.edu))

# Abstract

# Introduction

After submitted a paper to a conference, you, as the author of the paper, usually are most interested in whether your paper will be accepted or not. However, it is the decision made by the reviewers of conference. One question rises here is that “can we predict the result?” or more specifically, “can we find the criteria of a certain conference?” In this paper, we introduce Automatic Reviewer which may give you the answers, even before you submit the paper to the conference.

This paper studies the following two problems: 1. What statistical features of papers are correlated to the decision of the conference? 2. To what extent can an automatic program predict whether a paper will be accepted or not?

We extract several features which range from paper metadata to paper topics. We use supervised learning algorithm (SVM) to learn our system model, and use cross-fold validation to evaluate our method.

# Related Work

Predicting has been studied in many areas, especially in e-commerce. Online store like Amazon uses data mining techniques to predict what the customer would like to buy based on her/his previous purchase history, and other customers’ purchase behavior. Such techniques are known as association analysis [1]. Hal Varian [2] studied using Google queries to help predict economic activity. In last few years, with the great success of social networking, more and more researches focus on that area too. Devavrat Shah and Stanislav Nikolov recently announced their new algorithm that can predict trending topics on Twitter as much as five hours in advance and with an accuracy of 95% [3]. However, as far as we know, no similar study has been done in predicting conference paper acceptance. The only related study focus on the visual structure of conference papers [4].

# Dataset

Currently, we only focus on ACL conference papers. We also limited our study to the papers which were published in year 2007, 2010, and 2012. The reason is that ??

The full papers are used as positive samples. Due to lack of real rejected papers, we assume that all the workshop papers are not as good as the full papers, and use those which have more than 8 pages as negative samples in our study. The statistics of dataset is listed as below:

|  |  |  |  |
| --- | --- | --- | --- |
| Year | Positive | Negative | Total |
| 2007 | MISS |  |  |
| 2010 |  | MISS |  |
| 2012 |  |  | MISS |
| Total |  |  |  |

We use several features: some of them are directly extracted from the paper; the others are calculated by using more sophisticated techniques.

* Metadata

Metadata are those information extracted from paper itself, like the number of pages, total numbers of tables/formulas/figures, number of tables/formulas/figures per page, max number of tables/formulas/figures per page. In our current experiment, we only use total number of tables/formulas/figures as features.

* Author Ranking

We extracted top 2000 authors in “Natural Language & Speech” area from Microsoft Academic Search[[1]](#footnote-1), and use this extracted list as the authors’ ranking. If author doesn’t appear in the list, the rank will be zero.

* Popular techniques mentioned in the paper

We also manually created a list of popular techniques which may be used in the NLP research. The terms of techniques and its synonyms, abbreviations are group together. We check each technique in the paper, to see if it has been mentioned or not.

* Words in the Title (TF-IDF Score)

Based on the assumption that novel ideas are favored in the conference, and papers with novel ideas may have titles that contain infrequent terms which are more attractive to the reviewers, we count the TF-IDF score for the title terms, and use score as one of features.

* Topic – LDA (Latent Dirichlet Allocation )

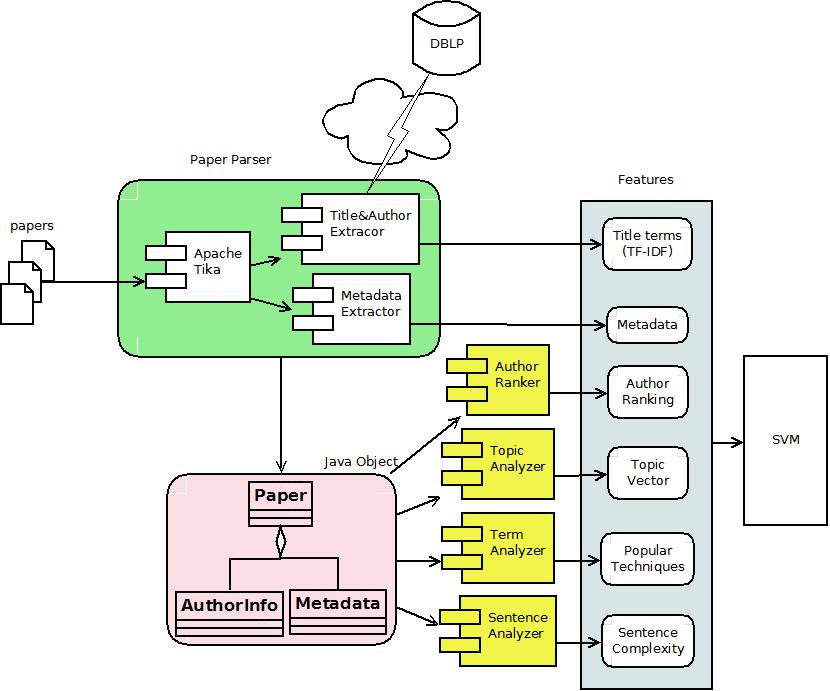
LDA[??] is used to extract topics from all the papers. In our experiment, top 20 topics are extracted, and the probability distribution vector of topics is used as one feature.

* Sentence complexity

All the content sentences within papers are parsed into phrase structure tree by using Stanford Parser[[2]](#footnote-2). The depth of tree is used as complexity of the corresponding sentence. We use the frequency vector of all complexities range from 0 to 40 as a feature vector (those complexities which larger than 40 are counted as 40).

# Approaches

In this study, we first downloaded all the papers from ACL web-site. Then we send them to Automatic Reviewer. There are three main components in Automatic Reviewer: 1. paper parser; 2. feature extractors; and 3. modeling and evaluation component. The architecture is showed as below:



Papers are first parsed into structured data (Java objects). Metadata from the paper are used directly as features. Then the parsed papers are processed by 4 feature extractors. Finally, the feature vectors are sent to Support Vector Machines (SVM) for model learning and evaluation.

### Paper parser

There are three sub-components in the paper parser: Apache Tika[[3]](#footnote-3), Metadata Extractor, and Title & Author Extractor.

The original papers which downloaded from ACL web-site are first sent to Apache Tika which is a PDF file parser. It can parse PDF file into HTML-like structured data. The paper is parsed into several <page> tags which contain <p> tags that denote the raw paragraphs in the original paper.

Because the parsing is not very accurate, the raw paragraphs contain all the texts in the paper, including the page foot, like “Proceedings of …”, “BioNLP 20...”. After all these noise paragraphs are discarded by regular expression matching, the first raw paragraph is considered as the candidate title. The paragraph starts with “abstract” is the abstract paragraph. If there’s no paragraph starts with “abstract”, the first paragraph which contains more than 300 English letters is considered as the abstract paragraph.

We consider all the paragraphs between title and abstract as raw author info which contains author names, affiliations, and email addresses. All paragraphs after the key-word “reference” or “references” are considered as paper references. The other paragraphs between abstract and references are considered as content paragraphs.

Content paragraphs are then sent to Metadata Extractor. Metadata Extractor uses regular expression *“^(fig\.|figure)\s\*\d+”* and *“^(tab\.|table)\s\*\d+”* to find the figures and tables in each page. Continuous paragraphs which contain more than 30% non- alphabet English letters are considered as one formula. These approaches may not be very accurate. However, since both training and testing papers are parsed based on this same rule, we think the inaccuracy is acceptable. Using the methods above, all the metadata are generated.

Candidate title and raw author information are processed by Title & Author Extractor. This extractor utilizes DBLP’s web service API[[4]](#footnote-4). This API supports fuzzy query, so we send the raw title to the API and retrieve the most reliable one as the real title for the paper. If a paper does not exist in DBLP, the raw title is accepted. Together with the title, author names are also returned from DBLP’s web API. The extractor uses regular expression to match the email addresses within raw author info, the text between the author name and matched email addresses are considered as the author’s affiliation. Currently, only title and author names are used in our experiment.

Through the paper parser, the PDF paper is parsed into structured Java “Paper” object. “Paper” contains one “Metadata”, several “Author” objects, and all sentences in each paragraph. The corresponding UML is attached in the Appendix.

### Feature Extractors

Besides the metadata feature which directly comes from the parsed Metadata object, other features are generated from the following 4 feature extractors.

* Author ranker

Author ranker extracts top 2000 authors in “Natural Language & Speech” area from Microsoft Academic Search. Then the author names, affiliations and ranking are stored in a Lucene[[5]](#footnote-5) index. Because abbreviation in names may appear in the paper, we send fuzzy query of author name to the Lucene index to retrieve the ranking information. Also an empirical threshold matching score is set to filter out those unlikely matching.

* Term analyzer

Term analyzer tries to catch technical concepts from paper at term-level. Right now this analyzer mainly focuses on the popular technique terms mentioned in the paper. Unlike topics of paper, these technique terms only concentrate on technique aspects of the paper content. We manually created a list of technique terms and their synonyms and abbreviations. Term analyzer uses this term list to generate a Boolean feature vector. Each value in the vector corresponds to the existence of each technique term.

|  |  |
| --- | --- |
| Term | Possible forms |
| LDA | LDA; Latent Dirichlet allocation |
| HMM | HMM; HMMs; Hidden Markov model |
| MaxEnt | MaxEnt; maximum entropy |
| MEMM | MEMM; maximum-entropy Markov model; maximum entropy Markov model |
| CRF | CRF; CRFs; Conditional random fields |
| NER | NER; named-entity recognition; named entity recognition; entity identification; entity extraction |
| SVM | SVM; SVMs; support vector machine; support vector machine; support vector network; support vector networks |
| LogisticRegression | logistic regression |
| LinearRegression | linear regression |
| LSI | LSI; latent semantic indexing; SVD; singular value decomposition; LSA; latent semantic analysis |
| KLdivergence | KL-divergence; KLdivergence; Kullback-Leibler divergence; Kullback Leibler divergence; information gain |
| MutualInformation | mutual information; |

* Topic analyzer

We believe that during a period of time, certain topics are more interested in certain research area, and those papers studied such topics have more chances been accepted by the conference in that research area. In order to find out what topics are more favored in the conference, we use LDA (Latent Dirichlet allocation) to extract 20 topics from all the training data. The probability distribution of the topics of paper is used as a feature vector.

* Sentence analyzer

Because there is a limit in the number of pages of conference paper, we think the sentence would be more complicated if the paper contains more information, and such paper are more valuable than others. Based on this assumption, we use sentence analyzer to analyze the complexities of sentences in each paper. The paragraphs in paper are split into sentences first. And then these sentences are parsed by Stanford Parser. The entire parsing process is extremely slow. It costs more than 27 hours in our experiment machine (Intel Core i5, 2.27GHz, 6G RAM for JVM) with 5 working threads in parallel. After we got the parsed phrase structure tree from Stanford Parser, the depth of each sentence tree is used as the measurement of sentence complexity.

# Results and discussion

We use the SVM with 10-fold-cross validation to evaluate our Automatic Reviewer system. The result of all positive predictions is used as the baseline.

// TODO

# Discussion and Conclusion // may be changed according to new evaluation result

Form the result of evaluation, we can see that our model can predict whether a paper is accepted or not with good accuracy. The metadata and the topic distribution learned by LDA model are the best features. This result proved our assumption that certain topics are more favored by certain conference. Also, tables number is the best of all three metadata, which indicates that the reviewers of paper are more willing to see the well structured result of research rather than plain text. // TODO

Features like popular techniques and author ranking do not work very good. This phenomena indicates that the quality of ACL conference is very high, the papers are thoroughly studied before accepting/rejecting, and the conference cares more about the content of paper rather than whether the techniques used are popular or not. Because the reviewing is blind, the author’s ranking cannot influence the decision of the reviewers.

We find that our model seems work better on the more balanced dataset (i.e. the numbers of positive and negative samples are close to each other)

// TODO

In the future, we want to study how the topic/style of papers changes over the time. And we also want to extend our research to other conferences in other area. We want to find out the main difference between the accepted papers in different conferences, and different areas.

Currently, the negative samples are workshop papers which may not be representative. We are considering using papers come from lower-level conferences within the same research area as the negative samples.

# References

[1] Bing Liu (2011), Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data, Second Edition. Springer

[2] CHOI, H. and VARIAN, H. (2012), Predicting the Present with Google Trends. Economic Record, 88: 2–9. doi: 10.1111/j.1475-4932.2012.00809.x

[3] Predicting what topics will trend on Twitter, A new algorithm predicts which Twitter topics will trend hours in advance and offers a new technique for analyzing data that fluctuate over time. <http://web.mit.edu/newsoffice/2012/predicting-twitter-trending-topics-1101.html>

[4] Paper Gestalt, Carven von Bearnensquash, <http://vision.ucsd.edu/sites/default/files/gestalt.pdf>

# Appendix

### Additional materials

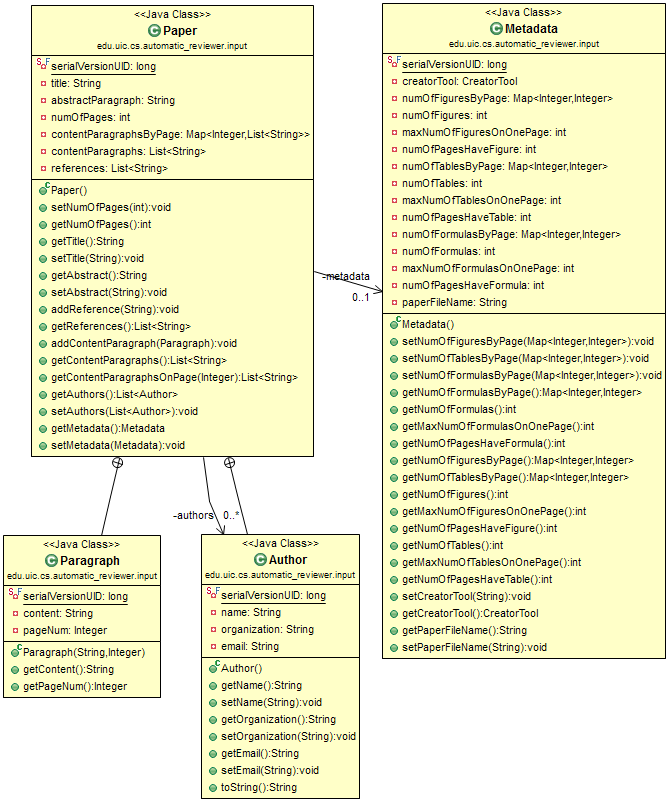
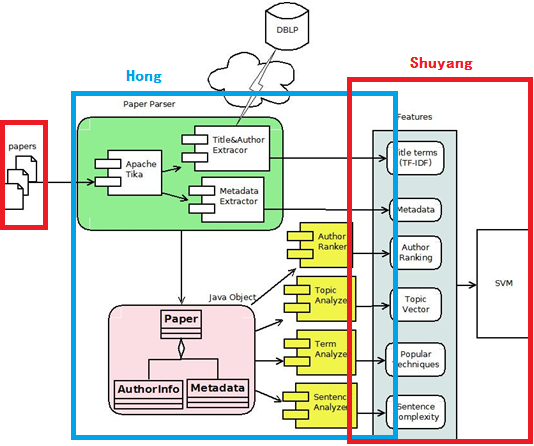


Fig. ?: UML for the parsed Java Paper object

### Project workload

* The blue parts are done by Hong Wang.
* The red parts (including original paper gathering and evaluation) are done by Shuyang Lin.
* This project report and previous presentation slides are done by them together.



1. http://academic.research.microsoft.com/RankList?entitytype=2&topDomainID=2&subDomainID=9 [↑](#footnote-ref-1)
2. http://nlp.stanford.edu/software/lex-parser.shtml [↑](#footnote-ref-2)
3. Apache Tika, http://tika.apache.org/ [↑](#footnote-ref-3)
4. http://www.dblp.org/search/api/?q=[TITLE], where "[TITLE]" is the paper title to search. [↑](#footnote-ref-4)
5. Apache Lucene, http://lucene.apache.org/ [↑](#footnote-ref-5)