Automatic Reviewer: Predicting Whether a Paper will be Accepted or Rejected

Hong Wang
Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607, USA
hwang207@uic.edu

Shuyang Lin
Department of Computer Science
University of Illinois at Chicago
Chicago, IL 60607, USA
slin38@uic.edu

Abstract

//TODO

1 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.

2 Related Work

Predicting has been studied in many areas, especially in economics.?? Hal Varian studied using Google queries to help predict economic activity // TODO more examples??

However, as far as we know, no similar study has been done in predicting conference paper acceptance. The only related study is focus on the visual structure of papers [??].

// TODO 4 related papers?!

3 Data Set

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 them as negative samples in our study. The statistics of dataset is listed as below:

We use different features: some of them are directly extracted from the paper; the others are calculated by using several different 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 [??], and use this extracted list as the authors ranking. If author doesnt 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 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 such papers may have titles that contain novel 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 [??]. The depth of tree is used as complexity of the corresponding sentence. //TODO how exactly we use it??

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 several feature extractors. Finally, the feature vectors are sent to Support Vector Machines (SVM) for model learning and evaluation.

4 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. model and evaluation component. The architecture is showed in Figure 1.

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 several feature extractors. Finally, the feature vectors are sent to Support Vector Machines (SVM) for model learning and evaluation.

4.1 Paper parser

There are three sub-components in the paper parser: Apache Tika [??], Metadata Extractor, and Title & Author Extractor. Downloaded original papers 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 contains ¡p¿ tags that denote the raw paragraphs in the original paper.

Because the parsing is not very accuracy, the raw paragraphs contain all the texts in the paper, including the page foot, like Proceedings of , BioNLP 20.... Those paragraphs are discarded. Then, the first raw paragraph is considered as the candidate title. The paragraph starts with abstract is the abstract paragraph. If theres no paragraph starts with abstract, the first paragraph which contains more than 300 English letters is considered as 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 are considered as 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 match 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 DBLPs web service API: http://www.dblp.org/search/api/?q=[TITLE] where [TITLE] is the paper title to search. 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. Together with the title, author names are also returned from this API. Then the extractor uses regular express to match the email addresses within raw author info, the text between the author name and email addresses are considered as the authors affiliation.

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

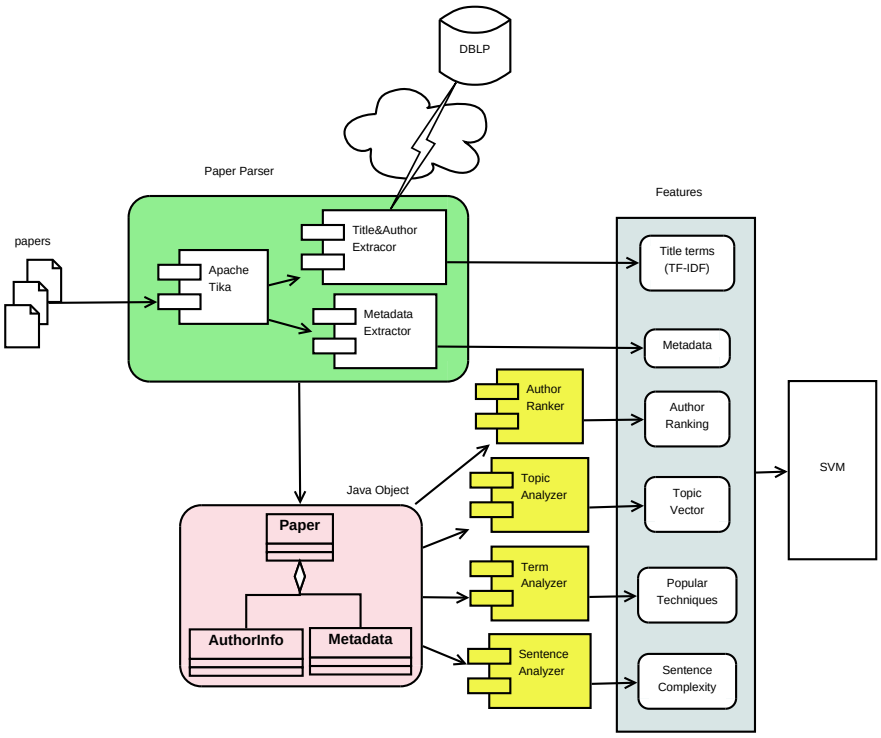

Figure 1: The architecture of the proposed approach

4.2 Feature Extractors

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

- **Author ranker** Author ranker extracts the top 2000 authors in Natural Language & Speech area from Microsoft Academic Search. Then the author names, affiliations and ranking are stored in a Lucene [??] index. Because there may be abbreviation in names which appeared in the paper, we send fuzzy query of author name to Lucene index to retrieve the ranking information. Also an empirical threshold value are set to filter those unlikely matching.

- **Term analyzer** Term analyzer tries to catch concepts from paper at term-level. Right now the analyzer mainly focuses on the popular technique terms mentioned in the paper. Not like topic of paper, these technique terms concentrate on technique aspects of paper content. We manually created a list of technique terms and their synonyms and abbreviations. Term analyzer use this term table to generate a Boolean vector feature, each value in the vector corresponds to the existence of each technique term.

- **Topic analyzer** We believe that during a period of time, some certain topics are more interested in a given 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 interested by 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 limitation of the number of pages in every conference, we think the sentence would be more complicated if the paper contains more information. Based on this assumption, we use sentence analyzer to analyze the complexities of sentences in each paper. The paragraphs in paper are first split into sentences, 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.

5 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.

5.1 Evaluation on Each Single Year Data

As the first part of our experiment, we evaluate the proposed apporach the papers of each single year. For each year of the years 2012, 2010 and 2007, we use 10-fold cross validation for the evaluation.

5.2 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 authors 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.

Additional materials

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 both of them.

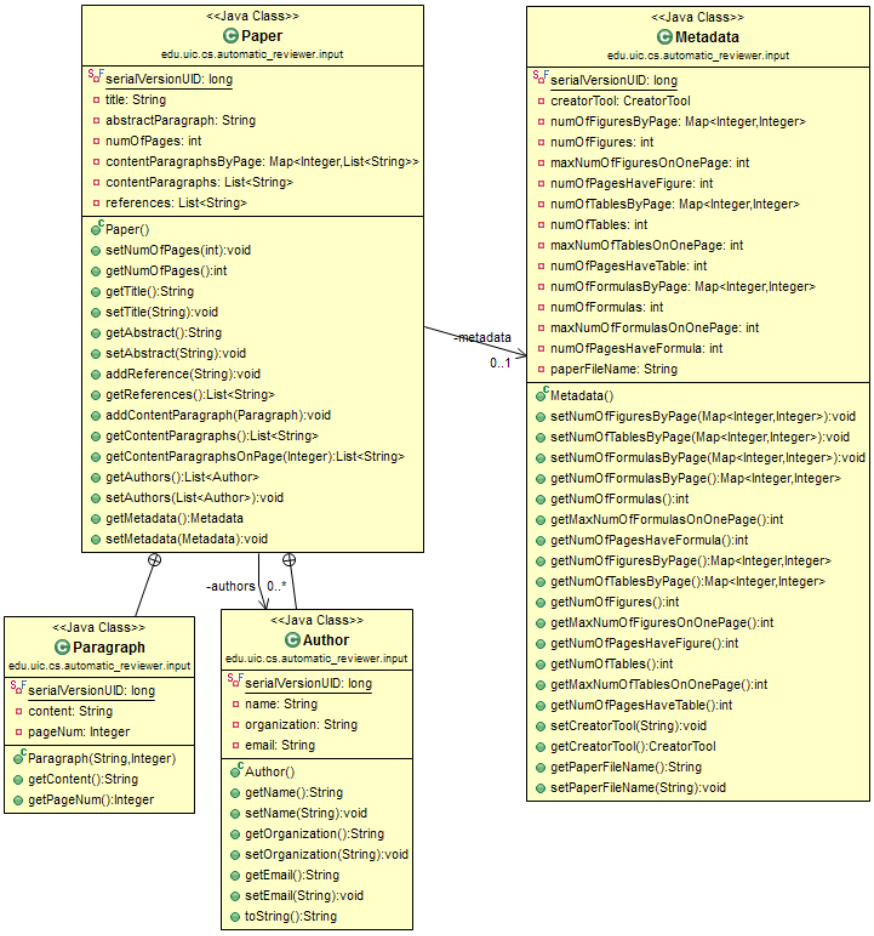

Figure 2: UML for the parsed Java Paper object

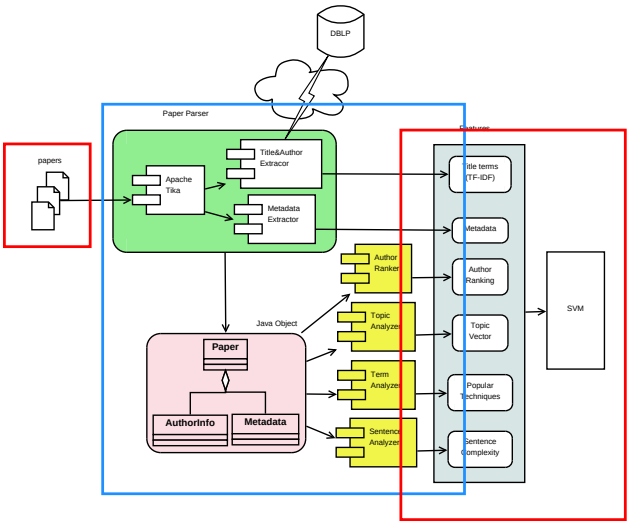

Figure 3: UML for the parsed Java Paper object