Contextual Rule-based Feature Engineering for Author-Paper Identification

Erheng Zhong[†], Lianghao Li[†], Naiyan Wang[†], Ben Tan[†], Yin Zhu[†], Lili Zhao[†], Qiang Yang[†‡]
[†]Hong Kong University of Science and Technology, Hong Kong
[‡]Huawei Noah's Ark Lab, Hong Kong
{ezhong,llias,nwangab,btan,yinz,lzhaoae,qyang}@cse.ust.hk

ABSTRACT

We present the ideas and methodologies that we used to address the KDD Cup 2013 challenge on author-paper identification. We firstly formulate the problem as a personalized ranking task and then propose to solve the task through a supervised learning framework. The key point is to eliminate those incorrectly assigned papers of a given author based on existing records. We choose Gradient Boosted Tree as our main classifier. Through our exploration we conclude that the most critical factor to achieve our results is the effective feature engineering. In this paper, we formulate this process as a unified framework that constructs features based on contextual information and combines machine learning techniques with human intelligence. Besides this, we suggest several strategies to parse authors' names, which improve the prediction results significantly. Divide-conquer based model building as well as the model averaging techniques also benefit the prediction precision.

1. INTRODUCTION

One task of KDD Cup 2013 is to determine which papers in an author's profile were truly written by a given author. Due to the author-name ambiguity, one paper may be assigned to an author incorrectly. This can be caused by two factors: firstly, authors may use different variations of their own names and secondly, names of different authors may be similar or even the same. Thus, this task is also closely related to name disambiguation which is important for many applications, especially in literature organization [7]. This competition is challenging due to the following reasons:

- The data are very noisy and different data files can be inconsistent with each other. We need to clean the data to discovery truth.
- The author sets of training and test data are disjoint and thus we cannot exploit the knowledge of each author directly.
- The given information may be limited. Some important factors are missing, such as the bibliography.

During the competition, we propose to solve this task using supervised learning. Specifically, we aim to represent each given author-paper pair as an instance and then build a classification model to predict whether the paper is written by the author or not. In addition, according to the evaluation target, where results are evaluated by Mean Average Precision of all authors, we need to rank the papers wrote by the author before other papers. Thus, we formulate the task as a personalized ranking problem, where author is analogous to query and paper is analogous to document. There are two challenges to achieve this goal: how to extract features from noisy raw data to represent each given author-paper pair as a feature vector and how to build a classifier to determine the correctness of the given author-paper pairs.

To cope with these challenge, we firstly propose a feature engineering framework called CRFC, Contextual Rule-based Feature Construction, where each feature is defined as the probability of one event happening under the given contexts and events are defined by rules. As described in the following, during the competition, our main workload is to define rules and contexts as many as we can. The constructed features can be divided into three categories: author-related features, that describe authors' characteristics only, such as the number of papers that wrote by the author; paper-related features, that extract from papers' profile only, such as the number of authors of the paper; and the most important author-paper features, that represent the similarity between the given author and paper. At the end of the competition, we totally construct 40 features. Consequently, considering its superior performance in other ranking competitions [3], we employ Gradient Boosted Tree [4], GBT, as the main classification model which takes author-paper feature vectors as input and outputs whether this pair is correct or not. GBT can be considered as a point-wise ranking model. In addition, by discovering some "super" features that can distinguish the data well, we propose a divide-conquer strategy that partitions the data into disjoint parts and then builds models in different data subsets. As demonstrated later, this strategy can improve the prediction precision by 0.001. In order to further improve the robustness and accuracy of our results, we average a series of boosted tree models together as our final submission.

We organize this paper as follows. Problem formulation and learning framework are described in Section 2. We proceed the feature engineering from noisy row data before building classification models. This step is described in Section 3. In details, we firstly state a feature construction framework that formally summarizes our feature engineering process during the competition, and then present the important rules and features we constructed. In Section 4, we highlight several techniques that we used in the contest, which help improve the prediction precision. Finally, we explain our exploration on the validation dataset and summarize our final submission and experimental results in Section 5.

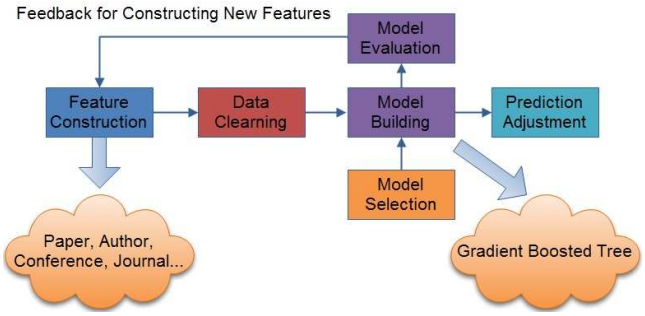

Figure 1: Main Flow

2. PROBLEM FORMULATION

In this section, we formulate the task and the objective we aim to achieve. Let $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^{n}$ denote the author set, where each author can be represented as a k_u-dimension feature vector and n is the number of authors; $\mathbf{P} = \{\mathbf{p}_i\}_{i=1}^{m}$ denote the paper set, where each paper can be represented as a k_p-dimension feature vector and m is the number of papers; $\mathbf{A} = \{\mathbf{a}_i\}_{i=1}^{q}$ denote the set of given author-paper pairs, where each pair can be also represented as a k_a-dimension feature vector and q is the number of pairs. By combining all these three parts together, we can represent each given author-paper pair between author i and paper j as $\mathbf{x}_{ij} = \{\mathbf{u}_i, \mathbf{p}_j, \mathbf{a}_{ij}\}$. In addition, we use $y_{ij} \in \{0, 1\}$ to denote the label of $\mathbf{x}_{ij}$, representing whether author i wrote paper j. According to the criterion, we have to rank those papers wrote by the author before other papers, i.e., $\sigma(\mathbf{x}_{ij}) \succeq \sigma(\mathbf{x}_{ik})$ if $y_{ij} > y_{ik}$ for each author i, where σ denotes the rank position. Thus, we aim to build a model f to minimize the following objective

$$\min_{f} \sum_{i=1}^{n} \sum_{j,k,y_{ij}<y_{ik}} [f(\mathbf{x}_{ij}) - f(\mathbf{x}_{ik})] + \lambda R(f) \qquad (1)$$

where $R(f)$ denotes the model complexity of f and $[\cdot]$ is an indicator function that $[\cdot] = 1$ if $\cdot > 0$ and $[\cdot] = 0$ otherwise. In this competition, we use Gradient Boosted Tree to build f. The main flow of the whole learning process is shown in Figure 1. The first and most important step is to construct features, in order to present author-paper pairs as feature vectors. As the raw data are very noisy and inconsistent, we have to do data cleaning during the feature construction process. These processes are followed by model building. Then the built models are evaluated through model selection and plotted to find those important kinds of features. Finally, those models that with superior performance are averaged to produce the final submission.

3. CONTEXTUAL RULE-BASED FEATURE ENGINEERING

As we stated above, the most critical point is to represent each author-paper pair as a feature vector. This is also our main workload during the competition. As the data contain different kinds of knowledge, including author and paper profiles, conference and journal information, and the text extract from the head of each paper, there may be infinite way to extract features from such raw data. In our previous work [14], we present a framework, Contextual Feature Construction, where each feature is defined as the probability of one event happening under the given contexts, to guide the feature construction process. This framework help us won the Nokia Mobile Data Challenge [5]. However, the data in KDD Cup competition are more complex, that means the events are hard to define. Thus, we extend this framework by using rules, where human can firstly define a set of rules and then use these rules to generate events.

3.1 Contextual Rule-based Feature Construction Framework

Different from a standard supervised learning problem, data from the KDD Cup are raw data, which cannot be taken as inputs for supervised models. Specifically, data of authors and papers are stored in multiple csv files, where each file contains one kind of knowledge. Thus, the first and the most important process is to construct features with high discriminability from these raw data and represent each author-paper pair as a feature vector, i.e., $\mathbf{x} = \{x_i\}_{i=1}^{k}$, where x_i represents the i-th value of the instance $\mathbf{x}$ and k is the number of features. Since there are infinite ways to perform feature construction, to make the feature construction process tractable, we propose to construct features based on events under different contexts. As papers that wrote by the author and other papers may have different probabilities of performing the same event under given contexts, and thus make it possible to discriminate those incorrectly assigned papers. For example, for a given author-paper pair, we can define one event as the number of papers wrote by the author and the context is the year when the paper was published, and then we can construct a feature as the number of papers wrote by the authors in the year when the paper was published. Thus, if the author wrote most papers in 2000s, then for a paper that was published before 1990s is unlikely wrote by the author.

Based on this motivation, we propose a unified framework: Contextual Rule-based Feature Construction (CRFC). CRFC defines a feature as the probability of one kind of event happening under given contexts. Formally, each feature is defined as $Pr(a|u, \Theta)$, where a is one event, u is a given author and Θ is the set of contexts, such as year/conference/journal. Based on this definition, we divide CRFC into three components:

- Enumerating possible events $\{a_i\}_{i=1}^{A}$ as many as possible
- Defining appropriate contexts, Θ
- Computing the conditional probabilities, $Pr(a|u, \Theta)$

As not all events can be defined easily and some may be complex, we propose to extract these events through rules. We firstly define a series of indicators and then use the combination of these indicators to produce rules and then events. For example, we have two indicators, one is whether the current author's name is capitalized and the other is whether the co-authors' names of the current paper are abridged. Then we can combine these two indicators as a rule with an "and" operation, and this rule is considered as an event. In next subsection, we will summarize those useful rules extracted from the data. Another issue is to define contexts, according to the data, we extract six basic contexts as follows

- No context, e.g., the number of papers wrote by the current author.
- Paper, e.g., the number of co-authors in this paper.
- Author, e.g., the number of cooperations between the current author and another author.
- Year, e.g., the number of papers in this year.
- Conference, e.g., the number of papers in this conference.
- Journal, e.g., the number of papers in this journal.

These contexts can be combined to produce more complex contexts, which we will describe in next subsections.

Table 1: Rule Summary

Index	Description
1	The last name of the current author is different from that of all other coauthors
2	The name and affiliation of the current author are the same as another coauthor in the paper
3	The affiliation of the current author is full but the affiliations of all other coauthors are empty
4	The affiliation of the current author is empty but the affiliations of all other coauthors are full
5	The name of the current author is full but the names of all other coauthors are abridged
6	The name of the current author is abridged but the names of all other coauthors are full
7	The name of the current author is capitalized but those of all other coauthors are not
8	The author ID has multiple records in a paper but the affiliations of each author ID are different
9	The author name and affiliation on the paper and author profile can exactly match
10	The value of Year, Conference ID or Journal ID is negative
11	The paper is published before 1965
12	The address of the current author is complete but the addresses of all the other coauthor are not
13	The address of the current author is not complete but the addresses of all the other coauthor are complete

Using the Bayesian rule, we obtain the following computing equations to compute the values of features:

$$Pr(a|u,\Theta) = \frac{Pr(a,\Theta,u)}{Pr(\Theta,u)} \propto \frac{n_{a,u,\Theta}}{n_{u,\Theta}} \quad (2)$$

where $n_{u,\Theta}$ is the number of records under contexts Θ, $n_{a,u,\Theta}$ denotes the number of events a that author u takes under contexts Θ. Based on the conclusion in [8], this estimation is also the result of maximum likelihood estimation.

In addition, to make the computational process more efficient, we formulate the original data into an entity-relation model, where one kind of information is an entity (corresponding to one csv file) and the relation between two entities is built according to IDs, such as author ID, paper ID, etc. Then, each event is related to one query.

3.2 Rule Summary

We summarize some important rules in this section, as shown in Table 1. Most rules are conducted by analyzing the writing styles of names and affiliations of authors in each paper. For example, we found that if the name of the current author is full but those of all other co-authors are abridged, then this paper is unlikely to be written by the current author. These rules are combined from some atom indicators and then evaluated on the training data to check whether they have discriminative abilities. Finally, we extract 10 rules and then each author-paper pair can be represented as an external 10-dimensional vector. These constructed vectors as well as the corresponding labels are used to build a decision tree. This decision tree outputs a probability that the given author wrote the paper and we use this probability as an additional feature, which is the combination of the extracted rules.

3.3 Feature Summary

We summarize the extracted features in Table 2. Basically, these features are derived from the CRFC framework and can be divided into three groups: author-related features, that describe authors' characteristics only, such as the number of papers that wrote by the author; paper-related features, that extract from papers' profile only, such as the number of authors of the paper; and author-paper features, that represent the similarity between the given author and paper. Among them author-related features are constructed without contexts, paper-related features are constructed with paper contexts, author-paper related features are constructed with author/conference/journal/year contexts and their combinations, and hence have the most discriminative powers.

4. MODEL BUILDING PRACTICES

Besides the main flow of the learning process, there are some other key factors to improve the prediction precision.

4.1 Author Name Parser

As discussed in Section 1, it is difficult to assign one paper to an author if he or she publishes under different author names. In this subsection, we discuss how to match the different variations of author-name using only the author-name itself. For example, identifying that "Larry Page" is the same person as "Lawrence Page". This task is not as easy as it sounds. Since the name variation may be caused by many factors, such as nicknames, misspellings, hyphenations, reverse, and etc., we propose to extract a set of matching rules to find the same author-name. Suppose there are n different author-name records in the dataset, in order to find the matched author-names we have to examine all $\Theta(n^2)$ pairs of author-name records which is intractable when the author-name records are in large volume, say hundreds of thousands in this competition. As we manually matched hundreds of author-names, we found that there are very few name variations occur in the last name. So in our solution we first group all author-name records according to their last names, and then find the matched name in each group separately. For matching the author-name in each group, we adopted a set of matching rules obtained from domain experts. We also used some existing name matching rules in open source software "Lingua-EN-MatchNames" [1]. Table 3 summarizes the matching rules used in our solution.

In addition, we also consider these names as the same author if (a) the edit distance between the first name and middle name of two authors is less than a threshold. (b) the length of the first name of one author is two. The first character can match the other's, and the second character appears later. In our solution, all non-alphabetical characters in the names beforehand.

4.2 Content Similarity

Since an author tends to publish papers under a limited number of research topics, the papers an author published may share many common domain vocabulary. So in our solution, we propose to use the content similarity between the title of a paper and the common used terms of an author as a content feature. For each paper, we apply the standard text preprocessing (i.e., removing stop words, converting letters to low case, stemming) on it and then represent it as a bag-of-word vector. For each author, we merge all the papers

[1]http://search.cpan.org/ brianl/Lingua-EN-MatchNames-1.12/

Table 2: Feature Summary

Index	Description
	Author-Paper Pairs
0	Number of cooperation times between the current author and other coauthors in the paper
1	Number of papers between the current author and other coauthors in the paper
2	Number of cooperation times among the coauthors Number of shared papers among the coauthors
4	Number of coauthors that have duplicated author-paper pairs
5	Whether the name on the paper is the same as the name in the author profile
6	Whether the affiliation on the paper is the same as the name in the author profile
7	The similarity between the author and paper based on affiliation using simhash
8	Number of papers published in the current conference
9	Number of papers published in the current journal
10	The sum of the 9th and 10th features
11	Number of words in the title used by the author
12	Number of words in the keywords used by the author
13	Number of words conference/journal name used by the author
14	The ratio of cooperation times between the coauthors who exist in duplicated records and other coauthors
15	The ratio of shared papers between the coauthors who exist in duplicated records and other coauthors
16	The ratio of shared papers between the coauthors who exist in duplicated records and other coauthors in current year
17	The ratio of shared papers between the coauthors who exist in duplicated records and other coauthors in current conference
18	The ratio of shared papers between the coauthors who exist in duplicated records and other coauthors in current journal
19	The ratio of used vocabularies in the title between the coauthors who exist in duplicated records and other coauthors
20	The ratio of used vocabularies in the keywords between the coauthors who exist in duplicated records and other coauthors
21	Similarity between the name on the paper and the name in the author profile
22	Similarity between the affiliation on the paper and the affiliation in the author profile
23	Tree induced rule-based feature described in last section
24	Similarity between the text of the author and paper based on bag-of-words
	Paper
25	Number of authors
26	Year
27	Conference ID
28	Journal ID
29	Number of records that exist multiple times
30	Averaged number of papers of all authors
	Author
31	The simhash value of the affiliation
32	Number of conference papers
33	Number of journal papers
34	Number of all papers
35	Averaged number of coauthors in all wrote papers
36	Number of records that exist multiple times
37	Number of distinct journals
38	Number of distinct conferences
39	Number of distinct years

he or she wrote to obtain the bag-of-word representation for an author. The cosine similarity between bag-of-word vectors is used as the content feature.

4.3 Divide-Conquer Model Building

One property of the dataset is that, there is a super "feature" based on the number of duplicated author-paper records. That means if one author-paper pair exists multiple times, this paper is wrote by the author with high probability. According to our statistic, 99% pairs are truly assigned if they have multiple records. This motivates us to design a divide-conquer model. As for different data parts, the effective features can be different. Specifically, we firstly separate the author-paper pair instances into two parts according to their number of duplicated records. Then for each data subset, we build a GBT model. For all test data instances, they are partitioned firstly and then the correspondent GBT model will produce a probability that how likely this author-paper pair is assigned correctly. Finally, we generate the ranking list based on these probabilities.

4.4 Model Averaging

Finally, we perform model averaging to produce submitted predictions. The motivation is that, on the one hand, labeled data are noisy so the model may overfit; on the other hand, different kinds of models describe data from different aspects and may provide more robust results if we combine their predictions. Formally, the final model $h(\mathbf{x})$ is defined as

$$h(\mathbf{x}) = \frac{1}{M}\sum_{f} f(\mathbf{x}) \tag{3}$$

where M is the number of models for averaging. We analyze that the averaged model can reduce the prediction variance as follows. Let f^* denote the ideal model and $d(f, f^*)$ denote the difference between f and f^*. We obtain

$$d(f, f^*) = \mathbf{E}_f\big(f(\mathbf{x}) - f^*(\mathbf{x})\big)^2 = \mathbf{E}_f\big(f(\mathbf{x})^2 - 2f(\mathbf{x})f^*(\mathbf{x}) + f^*(\mathbf{x})^2\big) \tag{4}$$

On the other hand, the difference $d(h, f^*)$ is

$$\begin{aligned} & d(h, f^*) = \mathbf{E}\big(\mathbf{E}_f(f(\mathbf{x}) - f^*(\mathbf{x}))\big)^2 \\ = \;& \mathbf{E}\Big(\big(\mathbf{E}_f f(\mathbf{x})\big)^2 - 2\big(\mathbf{E}_f f(\mathbf{x})\big)f^*(\mathbf{x}) + f^*(\mathbf{x})^2\Big) \\ \leq \;& d(f, f^*) \qquad as \qquad \mathbf{E}[f]^2 \leq \mathbf{E}[f^2] \end{aligned} \tag{5}$$

This provides us the theory guarantee to perform model averaging.

Table 3: Summary of name matching rules

Index	Description	Examples
1	Letter case	“Homer Simpson” vs. “HOMER SIMPSON”
2	Hyphenation	“Hong-Hu Zhu” vs. “Honghu Zhu”
3	Nickname	“Bill Gates” vs. “William Gates”
4	Misspelling	“Grene ” vs. “Green”
5	Name chunks	“Ralph Mac Nally” vs. “RalphMac Nally”
6	Prefix (Dr., Prof., and etc.)	“Dr. Harrt Dankowicz” vs. “Harrt Dankowicz”
7	Suffix (Jr., Senior, and etc.)	“Kenneth Powell Jr.” vs. “Kenneth Powell”
8	Abbreviation	“Robert A. Granat” vs. “R. A. Granat” and “Hong-Hu Zhu” vs “H.-H. Zhu”
9	Similar phonetics	“Hanson” vs. “Hansen”
10	Symbols	“O’Brien” vs. “Obrien”
11	English name for Asian	“Xiaojin Jerry Zhu” vs. “Xiaojin Zhu” and “Yeong C. Kim” vs. “Chonggun Kim”
12	Name reverse for Asian	“Binyin LIU” vs. “LIU Binyin”

Table 4: Performance Comparison

Model Description	MAP
Official Baseline	0.85046
Only Duplicated Information	0.95457
- Duplicated Information	0.97928
- Rule-based Feature	0.98254
- Content Similarity Feature	0.98317
Basic GBT	0.98459
+ Divide-Conquer Model Building	0.98489
+ Up-sampling	0.98515
+ Model Averaging	0.98543

Figure 2: Efficiency

5. EXPERIMENTS

In this section, we describe the experimental results we obtained on the validation dataset and evaluate the impotentness of each extracted feature. Basically, we use Gradient Boosted Trees as the main model. We test its performance on Mean Average Precision (MAP) with different feature sets, different model parameters and different model adjusting strategies. The description of the dataset can be found in [9].

5.1 Performance Comparison

The performance of different built models can be found in Table 4. We observe that, the largest improvement is achieved by adding the feature about duplicate information. By using only one feature, that represents the number of duplicated records of author-paper pairs, the MAP can exceed 95%. In addition, the rule-based and the content-similarity features improve the prediction precision further. Finally, by applying the divide-conquer strategy, up-sampling and model averaging, the performance can be boosted further. As a summary, the optimal parameters are as follows:

- Number of trees: 150

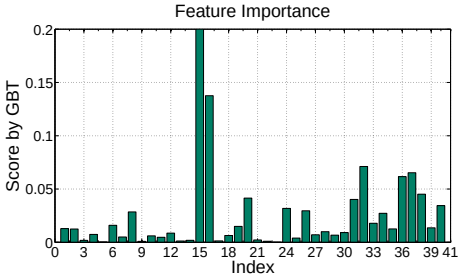

Figure 3: Feature Importance

- Tree height: 7
- Minimal number of instances to split: 10
- Sampling ratio: 0.95
- Learning rate: 1.0

We also studied the efficiency of the proposed framework. As shown in Figure 2, it takes about 47 minutes to complete the whole training process.

5.2 Feature Analysis

As we stated above, we construct new features based on the model performance in each round. Thus, we plot the feature importance according to GBT in Figure 3, where the indexes are correspondent to that in Table 2. We observe that Feature-14 about the duplicated information is the most important and the rule-based and content similarity features can significantly improve the prediction precision. We plot the first tree of GBT (with height 3) in Figure 4, which shows the hierarchy relations among features.

6. RELATED WORKS

This competition can be considered as a personalized ranking task. It is also an external problem of name disambiguation. As a state-of-the-art series of algorithms, learning-to-rank has been demonstrated to be effective in different application domains. We review these related works briefly in this section.

Name Disambiguation Disambiguating names is a very challenging problem in many applications, such as scientific literature management in this competition, social network analysis, people search, etc. To solve the disambiguation problem, much research has been conducted. Bilenko etc., employ method that combines multiple string similarity methods to capture different notions of similarity [1]. Tang etc., proposed a unified framework based on Hidden

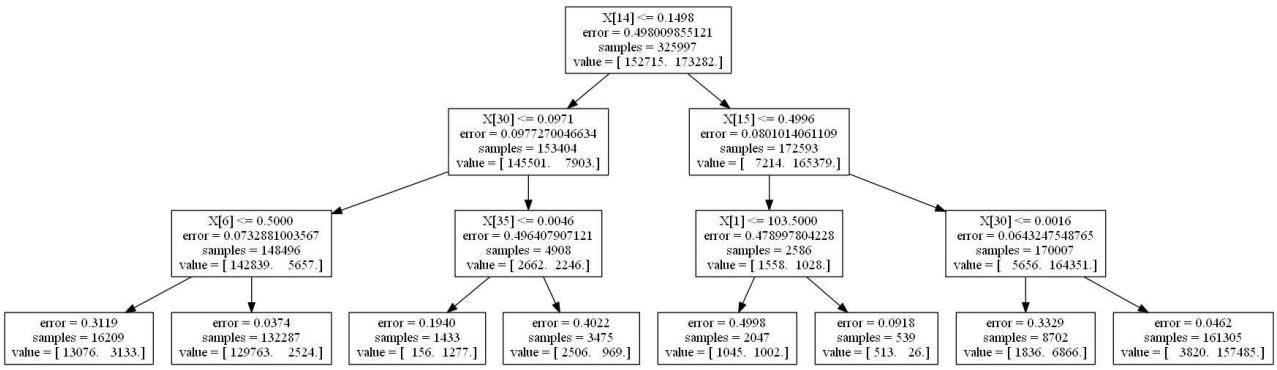

Figure 4: The First Decision Tree with Important Features

Markov Random Fields to model dependencies between scientific papers, then estimate the actually number of researchers who share the same name, using Bayesian Information Criterion [10].

Gradient Boosted Trees In our framework, we use Gradient Boosted Tree (GBT) as the main classification model. It is a well-known machine learning technique. The main idea is to compute a sequence of simple trees, where each successive tree is built to fit the residuals of the prediction of all previous trees combined [4]. GBT is highly adaptable and many different loss functions can be used during the boosting process. It can be also parallelized and able to deal with very large data set [12]. Due to its advantages, GBT has been widely used in many machine learning areas such as classification, regression, ranking, etc. It also serves well in many data mining competitions, including KDD CUP 2009 [11] and KDD CUP 2011 [13].

Learning to Rank As we formulate the problem of the competition as a ranking task, we also try some learning-to-rank algorithms, which rank the papers that are written by a given author before that are not. The main types of learning to rank algorithms are point-wise, pair-wise, and list-wise approaches, and a comprehensive survey can be found in [6]. The GBT algorithm used in our framework can be considered as point-wise ranking model. The ranking algorithms have been used well in many applications, such as document retrieval, sentiment analysis, collaborative filtering, etc. Some algorithms, like LambdaMART [2] which is a bagging GBT with pair-wise loss, contribute well to the winnings of Yahoo! Learning to Rank Challenge [3].

7. CONCLUSION

We have presented a supervised learning framework to predict whether one paper is wrote by a given author. Importantly, we proposed Contextual Rule-based Feature Construction (CRFC), a unified feature construction process, to build features from raw data for each author-paper pair. We formulate each feature as the conditional probability of one event under given contexts and events can be extracted from different rules. Consequently, these conditional probabilities can be computed through event counts in each csv file. We then preprocessed data, where each author-paper pair is represented as a feature vector. The products of the previous steps were taken as input for model building. Gradient Boosted Tree is exploited to construct models for prediction. To further improve the prediction precision, we exploit a divide-conquer process to build models. The final submission was based on model averaging because as analyzed, this strategy reduces the prediction variance and generalizes well.

Acknowledgement

We thank the support of Hong Kong CERG Projects 621211 and 620812.

References

[1] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, Sept. 2003.

[2] C. J. C. Burges. From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research, 2010.

[3] O. Chapelle and Y. Chang. Yahoo! learning to rank challenge overview. *Journal of Machine Learning Research - Proceedings Track*, 14:1–24, 2011.

[4] J. H. Friedman. Stochastic gradient boosting. *Comput. Stat. Data Anal.*, 38(4):367–378, Feb. 2002.

[5] J. K. Laurila, D. Gatica-Perez, I. Aad, J. Blom, O. Bornet, T. Do, O. Dousse, J. Eberle, and M. Miettinen. The mobile data challenge: Big data for mobile computing research. In *Mobile Data Challenge by Nokia Workshop*, Newcastle, UK, 2012.

[6] T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331, Mar. 2009.

[7] B.-W. On, D. Lee, J. Kang, and P. Mitra. Comparative study of name disambiguation problem using a scalable blocking-based framework. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '05, pages 344–353, New York, NY, USA, 2005. ACM.

[8] W. Pan, E. Zhong, and Q. Yang. Transfer learning for text mining. In C. C. Aggarwal and C. Zhai, editors, *Mining Text Data*, pages 223–257. Springer, 2012.

[9] S. B. Roy, M. D. Cock, V. Mandava, B. Dalessandro, C. Perlich, W. Cukierski, and B. Hamner. The microsoft academic search dataset and kdd cup 2013. In *KDD Cup 2013 workshop*, 2013.

[10] J. Tang, A. C. Fong, B. Wang, and J. Zhang. A unified probabilistic framework for name disambiguation in digital library. *IEEE Transactions on Knowledge and Data Engineering*, 24(6), 2012.

[11] J. Xie, V. Rojkova, S. Pal, and S. Coggeshall. A combination of boosting and bagging for kdd cup 2009 - fast scoring on a large database. *Journal of Machine Learning Research - Proceedings Track*, 7:35–43, 2009.

[12] J. Ye, J.-H. Chow, J. Chen, and Z. Zheng. Stochastic gradient boosted distributed decision trees. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 2061–2064, New York, NY, USA, 2009. ACM.

[13] H. Zhang, E. Riedl, V. A. Petrushin, S. Pal, and J. Spoelstra. Committee based prediction system for recommendation: Kdd cup 2011, track2. *Journal of Machine Learning Research - Proceedings Track*, 18:215–229, 2012.

[14] E. Zhong, B. Tan, K. Mo, and Q. Yang. User demographics prediction based on mobile data. *Pervasive and Mobile Computing*, (0):–, 2013.