Domain-Specific Automatic Scholar Profiling Based on Wikipedia

Ziang Chuai, Qian Geng, Jian Jin
jinjian.jay@bnu.edu.cn
Department of Information Management, Beijing Normal University
Beijing 100875, China

ABSTRACT

Scholar profiling is usually invited to provide junior researchers with domain-specific concepts systematically, but the organization of domain literatures is usually time-consuming. Then, an automatic scholar profiling method is preferred. Firstly, to extract some properties of a given scholar, structured data, like infobox in Wikipedia, are often used as training datasets. But it may lead to serious mis-labeling problems, such as institutions and alma maters, and a Fine-Grained Entity Typing method is expected. Thus, a novel Relation Embedding method based on local context is proposed to enhance the typing performance. Also, to highlight critical concepts in *selective bibliographies* of scholars, a novel Keyword Extraction method based on Learning to Rank is proposed to bridge the gap that conventional supervised methods fail to provide junior scholars with relative importance of keywords. Categories of experiments were conducted to evaluate the Relation Embedding method and the Keyword Extraction method, which show that the proposed approaches outperform existing baseline methods notably.

CCS CONCEPTS

• **Information systems → Wikis**.

KEYWORDS

scholar profiling, named entity typing, relation embedding, keyword extraction, learning to rank

ACM Reference Format:
Ziang Chuai, Qian Geng, Jian Jin. 2020. Domain-Specific Automatic Scholar Profiling Based on Wikipedia. In *Companion Proceedings of the Web Conference 2020 (WWW '20 Companion), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3366424.3383565

1 INTRODUCTION

With the size and dimension of data growing rapidly, scholar profiling is invited to provide junior scholars with a deep understanding of numerous unstructured information in a specific domain systematically[1, 30]. But the organization and reconceptualization of domain relevant literatures is usually time-consuming. To facilitate this task, an automatic approach to extract domain-specific information is expected. Accordingly, taking domain of information science as an example, a framework for scholar profiling is built in this study, focusing on the extraction of scholars' personal research experiences, i.e., alma maters and working institutions, as well as famous concepts that are highly relevant to their studies. Note that, to clarify the domain margin of information science, an RDF query is conducted in open knowledge bases, like Wikidata, to retrieve renowned information scientists who have a webpage in Wikipedia.

Personal research, as a kind of basic information for scholar profiling, can be utilized as guidelines for junior scholars' further studies. To extract such information, conventional Named Entity Recognition (NER) methods are usually based on Probabilistic Models and Deep Neural Networks (DNN), including Hidden Markov Model (HMM)[20], Conditional Random Fields (CRFs)[2] and CRF-based DNN[12] etc. Such supervised methods require a large and accurate training dataset, and Wikipedia is often utilized as a reliable and validate information source. Along with detailed descriptions, tabular information, known as *infobox*, is also presented in some entries in Wikipedia to give descriptions in a category structure, which is often utilized as given labels to generate training datasets for supervised methods[18, 29]. Specifically, if a term appears as the corresponding value of an attribute in an *infobox*, it will be labeled as the attribute wherever it appears in the article. However, such assumption ignores the local context, which can lead to a severe mislabeling problem and have a negative impact on the performance of supervised NER methods. For instance, in the Wiki-page about Gerard Salton, some sentences describing Salton's *institutions*, like 'Salton initiated the SMART Information Retrieval System when he was at Harvard University', might be mis-labeled as his *alma mater*, since *Harvard University* appears as the corresponding value of the attribute *alma mater* in his *infobox*. This problem involves all the probably ambiguous entity pairs, e.g. alma mater and institution, residence and nationality etc. and challenges many NER studies that leverage *infobox* as training data. To alleviate this problem, an embedding-based entity discrimination method is proposed, based on the intuition that terms that are semantically close should have similar representations.

Meanwhile, section called *selective bibliographies* in renowned scholars' Wiki-pages often covers a large amount of critical concepts. Accordingly, several tools to handle the bibliographic information were proposed, including Scholia, a user interface based on Wikidata[24]. To help junior researchers know what have been achieved in one domain, these critical concepts are expected to be highlighted and included in the profile. Thus, a proper Keyword Extraction method is needed. Although Keyword Extraction has been investigated in many studies, there are still some flaws in existing methods[15]. For instance, supervised methods, like KEA[32], always classify phrases into *keywords* or *non-keywords*, but the relative importance of concepts are ignored. For unsupervised methods, graph-based unsupervised methods, like TextRank[22], are usually time-consuming due to the traversing across the whole graph[26], and statistical feature-based unsupervised methods, like RAKE[28], mainly focus on the co-occurrence of terms, while semantic analyses are usually ignored[25]. In order to construct a well-performed supervised ranker, the techniques of Learning to Rank are introduced for Keyword Extraction[14].

Particularly, in this study, an embedding model named *translation with penalty* (TransP) is proposed to represent entities as well as their relations within a low dimensional vector space, in which initial vectors of entities and the semantic distance of entities and relations are firstly taken into considerations. It helps to better represent entities and their relations, which empowers a TransP-based approach for Fine-Grained Entity Typing. Besides, a Learning to Rank algorithm based on Adaptive Boost and Logistic Regression is proposed, referred to as AdaLogistic, to extract and rank keywords in a relative sense inspired by Jiang et al.[14]. In addition, some features aimed for *selective bibliographies* are defined to enhance the overall performance.

Specifically, main contributions are listed as follows: (1) a novel scoring function with penalty is proposed for the embedding model to force the embedding vectors stay semantically close to their descriptions; (2) Learning to Rank techniques are invited for Keyword Extraction, and a ranking algorithm based on AdaBoost and Logistic Regression is proposed to alleviate the problem that supervised methods can only return binary classification results.

In the rest of this study, related studies are summarized in Section 2. The research problems are formally defined in Section 3, and the framework is presented in Section 4. Detailed algorithms of fine-grained entity typing and keyword extraction are illustrated in Section 5 and Section 6 respectively. All the methods are evaluated in Section 7. Section 8 concludes this study.

2 RELATED WORKS

2.1 Entity Representation

To address the mis-labeling problem caused by *infobox* mentioned in Section 1, a series of methods for Named Entity Typing are proposed to discriminate similar entities into fine-grained types, and a proper relation embedding method is needed to make terms that are semantically close have similar representations so that entities from different types are easier to be distinguished[6, 27].

There have been various representation learning methods to embed relational knowledge into low dimensional vectors. For instance, RESCAL was proposed using a tensor factorization model[23], which aims to minimize a regularized loss function. In order to reduce the number of parameters and make the model fit for a large dataset, Bordes et al. proposed a canonical embedding model based on translation, named TransE[3]. The scoring function in TransE is defined as $f_r(\mathbf{h}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$, which means that $\mathbf{h}$+$\mathbf{r}$ should be close to $\mathbf{t}$ if triplet (*head*, *relation*, *tail*) appears in the corpus. Even though the intuition is simple, TransE has been proved to be powerful and outperforms many state-of-the-art methods notably. Based on TransE, a series of studies argued that entities and relations should be represented in distinct vector spaces, including TransH[7], TransR/CTransR[19] and TransD[13].

However, all the methods fail to consider the semantic distance of entities and relations to be embedded. In this way, entities and relations can be semantically far from their original meanings, which is against the purpose of knowledge embedding.

2.2 Keywords Extraction

Keyword Extraction has been investigated in a number of studies, which usually includes two main steps. In the first step, keyword candidates are generated from documents according to stop-words distribution, POS patterns, etc. Then, all the keyword candidates are divided into *Keywords* and *Non-Keywords* in the second step.

Many supervised methods are based on binary classifiers and manually-defined features. Wu et al. proposed a keyword extraction method based on naive Bayes, named KEA[32]. In order to integrate external knowledge into the model, He et al. proposed a keyphrase extraction model based on the prior possibility of a phrase being a keyphrase[9]. What is more, active learning method was utilized to reduce the amount of required training data for supervised methods by Auer et al.[4]. On the contrary, unsupervised methods rank candidate keywords according to scores of heuristic statistical features. Inspired by PageRank algorithm, several graph-based ranking models, including TextRank and multi-centrality index for graph-based keyword extraction method (MCI), are proposed to decide the importance of a term, which is represented as vertex in the graph[22, 31]. Based on statistical features of terms, e.g. the frequency and the sum of times that a word co-occurs with other words, Rose et al. proposed a ranking method, named RAKE[28].

However, there are still some flaws in these methods as mentioned in Section 1. To alleviate those problems, Learning to Rank algorithm was introduced to Keyword Extraction problem by Jiang et al.[14], where RankSVM algorithm was applied[10]. Since then, different Learning to Rank algorithms were invited and these algorithms were argued to perform better than RankSVM, including AdaRank algorithm[33], SVM-based AdaBoost algorithm[11].

2.3 A Brief Summary

To address the mis-labeling problem caused by *infobox*, embedding models are usually invited by Named Entity Typing methods to represent entities as well as their relations via low-dimensional vectors. But the performances of existing models drop when they are applied as an embedding learner for entity typing in the task of scholar profiling. Firstly, many Trans models initialize embedding vectors randomly, which ignores the semantic meanings in a certain scenario. Secondly, many Trans models have no restrictions for embedding vectors during the translating process, which means that embedding vectors can be adjusted freely in the whole vector space. In this way, embedding vectors might be far from their semantic meanings after a few epochs. In this study, a method named *translation with penalty* (TransP) is proposed for entity typing.

In addition, despite of the fact that both AdaRank and SVM-based AdaBoost achieve better performance than RankSVM in Information Retrieval tasks, there are some flaws when they are adopted for Keywords Extraction. Firstly, SVM-based AdaBoost assumes keyword extraction as a binary classification problem and fails to generate scores or ranks for keywords candidates. Secondly, AdaRank algorithm ignores weights of candidates, which leads to poor performance in Keyword Extraction due to the noises in keyword candidates. Thirdly, each weak ranker of AdaRank is trained using the feature that has the optimal weighted performance among all features, which means the final model is based on only a few features that have strong discriminating abilities. In this way, generality and performance of the final strong ranker in specific tasks can be badly influenced. To address these problems, a Learning to Rank model based on Adaptive Boost and Logistic Regression, referred to as AdaLogistic, is proposed for Keyword Extraction.

3 PROBLEM DEFINITION

To help junior scholars understand research records of renowned scholars, a three-step Named Entity Typing method, which includes coarse-grained entity detection, relation embedding and relation clustering, is proposed to alleviate the mis-labeling problem caused by *infobox*. Specifically, given an introduction sentence about a renowned scholar, denoted as $X = x_1, \cdots, x_n$, in which x_i is the i^{th} term in sentence X, it aims to identify fine-grained labels for each x_i, denoted as y_i. Also, in order to acquire critical concepts efficiently, a Keyword Extraction method is needed to analyze the section of *selective bibliographies* in Wiki-pages of renowned scholars. Specifically, the set of bibliographies $B = \{b_i\}, i = 1, 2, \cdots, n$ is taken as input, and keyword candidates $C_i = \{c_{ij}\}, j = 1, 2, \cdots, n(i)$ are expected to be extracted from each bibliography b_i. Afterwards, a ranked set of keywords $K_i = \{k_{ij}\}, j = 1, 2, \cdots, n(i)$ is identified for each bibliography b_i by ranking candidates in set C_i.

4 FRAMEWORK

In this section, a two-phase framework for scholar profiling based on Wikipedia is proposed, as presented in Figure 1.

In phase I, information of renowned scholars' educational backgrounds and working experience is extracted via a fine-grained Named Entity Typing method, which consists of three steps. In the first step, a list of triplets, (h, v, t), is generated as the corpus for entity typing, in which h and t respectively represent names of scholars and organizations, i.e. unidentified alma maters and institutions in our specific task, and v represents the verb phrase that modifies the relation between h and t. In the second step, TransP model is proposed to embed entities and verb phrases into low-dimension vectors for further identification. In the third step, a clustering approach is applied to type the coarse-grained results into expected fine-grained categories, and corresponding labels are given to each category according to pre-defined seeds.

In phase II, *selective bibliographies* of renowned scholars are given as input, from which keywords are extracted by a ranking-based method as academic concepts. Firstly, keyword candidates are generated by stop-words distribution and POS patterns. Then, a ranking algorithm is proposed based on AdaBoost algorithm and Logistic Regression, referred to as AdaLogistic, to decide the importance of being selected as keyword.

5 FINE-GRAINED ENTITY TYPING

5.1 Coarse-Grained Entity Detection

CRFs achieve great performance in NER[16]. When dealing with NER tasks, a specific label is assigned to each word in the input sequence, indicating both the boundary and the type of an entity, and the BIO schema is utilized in this study[5, 17]. Table 1 shows the set of features used in CRF.

Table 1: Features for CRF

Features	Description
Word Features	Word(n)
	Word(n-1)Word(n)
	Word(n-2)Word(n-1)Word(n)
POS Features	POS(n)
Orthographic Features	*Capitalized/Uppercase*(n)
	Contains Slashes/Numbers(n)
	Prefix/Suffix(n)
	POS(n-1)POS(n)
Syntactic Features	*Head Word*(n)
Boolean Features	e.g. *University*(n)
Gazetteer Label	*Gazeteer*(n)

Besides word features, POS features and Orthographic features, features about syntactic dependency are utilized to describe *head words* of each word in the dependency tree of a given training sentence. Function $HeadWord\,(n)$ returns a head token for $Word\,(n)$ in the dependency tree, which itself is a *Verb* or a *Noun*. *Null* will be returned if the expected token is not found. In addition, a few task-specific Boolean features are defined, like whether a word is *'university'* etc. In order to consider the category label of the current word, gazetteers-related features are utilized, which includes common names, temporal expressions, countries etc.

5.2 TransP for Relation Embedding

5.2.1 Preliminaries. With entities regarding organizations obtained by CRF, a list of triplets is built as the corpus for TransP model to be embedded. To clarify the proposed approach, some notations of TransP are introduced. h denotes head entity, t denotes tail entity and v denotes verb phrase between them in a triplet. Note that the bold letters $\mathbf{h}$, $\mathbf{v}$, $\mathbf{t}$ represent the corresponding vectors. $\mathcal{G}$ denotes the golden triplets, and $\mathcal{G}'$ denotes the corrupted triplets.

5.2.2 Initial Values. In the existing Trans models, random vectors are utilized as initial values which ignore the semantic meaning of entities and relations. To overcome this problem, a novel initializing method for vector $\mathbf{h}$, $\mathbf{v}$ and $\mathbf{t}$ is proposed.

Note that, Wikidata is a free and open knowledge base which contains descriptions for each entity in it. For instance, scholar *Gerard Salton* has *'American computer scientist'* as his description. Rich representation information is hidden behind these descriptions and, in this study, critical words in descriptions are utilized to initialize an entity representation. Specifically, three tokens with highest *tf-idf* value for each entity from its descriptions context are extracted, and weighted sum of the pre-trained word embedding vectors $\mathbf{w}$ of the selected tokens are utilized as the description vector $\mathbf{d}_i$ of entity i. Mathematically, it can be represented as,

$$\mathbf{d}_i = \sum_{j}^{3} tf - idf_{ij} \times \mathbf{w}_{ij} \qquad (1)$$

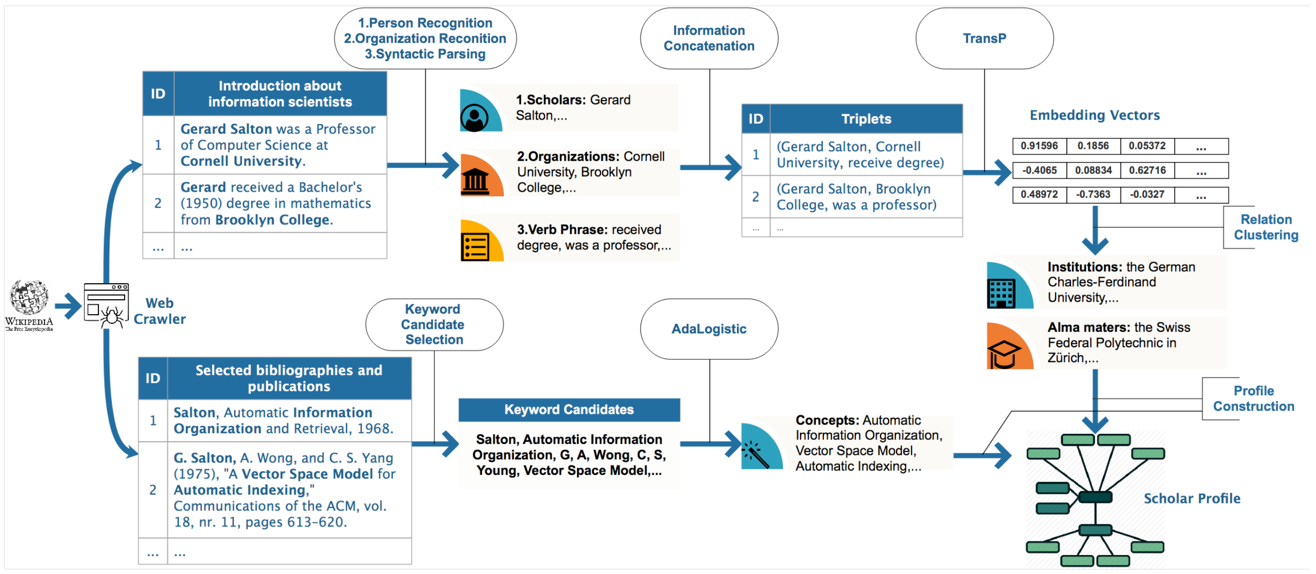
1.Person Recognition
2.Organization Reconition
3.Syntactic Parsing
Information Concatenation
TransP
ID | Introduction about information scientists
1 | Gerard Salton was a Professor of Computer Science at Cornell University.
2 | Gerard received a Bachelor's (1950) degree in mathematics from Brooklyn College.
... | ...
1.Scholars: Gerard Salton,...
2.Organizations: Cornell University, Brooklyn College,...
3.Verb Phrase: received degree, was a professor,...
ID | Triplets
1 | (Gerard Salton, Cornell University, receive degree)
2 | (Gerard Salton, Brooklyn College, was a professor)
... | ...
Embedding Vectors
0.91596 | 0.1856 | 0.05372 | ...
-0.4065 | 0.08834 | 0.62716 | ...
0.48972 | -0.7363 | -0.0327 | ...
Relation Clustering
Institutions: the German Charles-Ferdinand University,...
Alma maters: the Swiss Federal Polytechnic in Zürich,...
Profile Construction
WIKIPEDIA The Free Encyclopedia
Web Crawler
Keyword Candidate Selection
AdaLogistic
ID | Selected bibliographies and publications
1 | Salton, Automatic Information Organization and Retrieval, 1968.
2 | G. Salton, A. Wong, and C. S. Yang (1975), "A Vector Space Model for Automatic Indexing," Communications of the ACM, vol. 18, nr. 11, pages 613–620.
... | ...
Keyword Candidates
Salton, Automatic Information Organization, G, A, Wong, C, S, Young, Vector Space Model,...
Concepts: Automatic Information Organization, Vector Space Model, Automatic Indexing,...
Scholar Profile

Figure 1: Scholar Profiling

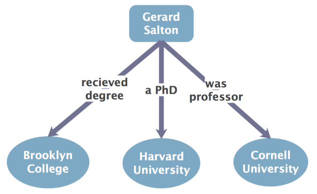

Figure 2: Neighbors of Gerard Salton

Next, to take the neighbor information into considerations for an entity representation, for a specific head entity h^*, the sum of description vectors of all tail entities, $\mathbf{d}_t$, are used as initial vectors of $\mathbf{h}_0^*$, and vice versa. Figure 2 shows an example of neighbors.

$$\mathbf{h}_0^* = \sum_{(h^*,v,t)\in\mathcal{G}} \mathbf{d}_t \qquad \mathbf{t}_0^* = \sum_{(h,v,t^*)\in\mathcal{G}} \mathbf{d}_h \tag{2}$$

As for initial values for relations, the pre-trained word embedding vector of the verb in a triplet is used.

5.2.3 Penalty for Embedding. One benefit to embed terms into a vector before typing to a fine-grained level is that entities that are semantically close can be embedded as similar representations and, it potentially helps to improve the performance of entity typing. But all the embedding models mentioned above such as TransE, TransH etc., has no restriction schemes for embedding vectors. In this way, embedding vectors from these approaches might be semantically far from their descriptions after a few epochs, and there will be no improvement for entity typing if these embedding vectors are utilized. To make vectors stay semantically close to their descriptions, a penalty scheme is needed for each entity in the scoring function f_v, with its description vector being the center, i.e. $\mathbf{h}_c = \mathbf{d}_h$ and $\mathbf{t}_c = \mathbf{d}_t$. Specifically, the penalty of an entity should be defined as $\|\mathbf{h} - \mathbf{h}_c\|_2^2$ or $\|\mathbf{t} - \mathbf{t}_c\|_2^2$.

5.2.4 A Translation Model with Penalty. In order to cope with the issue of initial values and lacking of restrictions, a translation model with penalty is proposed based on TransE. As explained, the scoring function can be defined as:

$$f_v(\mathbf{h}, \mathbf{t}) = \|\mathbf{h} + \mathbf{v} - \mathbf{t}\|_2^2 + \lambda_1\|\mathbf{h} - \mathbf{h}_c\|_2^2 + \lambda_2\|\mathbf{t} - \mathbf{t}_c\|_2^2 \tag{3}$$

$\lambda_1, \lambda_2 \in [0, 1]$ are tuning hyper parameters that are used to control the degree of supervision from $\mathbf{h}_c$ and $\mathbf{t}_c$.

The score $f_v(\mathbf{h}, \mathbf{t})$, as a kind of distance between $\mathbf{h}$+$\mathbf{v}$ and $\mathbf{t}$, should be lower for a golden triplet and higher for a corrupted one. With $\mathbf{h}_c$ and $\mathbf{t}_c$ being constants, the number of parameters of TransP are the same with TransE, which guarantees TransP for a comparable time-complexity against TransE while taking semantic information of entities and relations into considerations. To encourage the difference between golden triplets and corrupted ones, the following loss function is utilized:

$$\mathcal{L} = \sum_{(h,v,t)\in\mathcal{G}} \sum_{(h\prime,v\prime,t\prime)\in\mathcal{G}\prime} [\gamma + f_v(\mathbf{h}, \mathbf{t}) - f_{v\prime}(\mathbf{h}\prime, \mathbf{t}\prime)]_+ \tag{4}$$

where $[x]_+ = max(x, 0)$, γ is the margin separating golden and corrupted triplets. The embedding vectors of entities and relations can be obtained via optimizing the loss function $\mathcal{L}$.

5.3 Relation Clustering

After the embedding vectors $\mathbf{v}$ are obtained, the k-means algorithm is applied to group verb phrases into discriminative clusters. In order to alleviate the impact of initial cluster centers, the Canopy algorithm is applied to generate proper initial centers for k-means[21]. Seeds are selected in advance to determine the label of each cluster.

6 KEYWORD EXTRACTION

6.1 Keyword Candidate Selection

Intuitively, words in a keyphrase should carry meanings, which are described as content bearing, while stop-words are considered to be uninformative. Therefore, to select a list of keyword candidates from ungrammatical and short context like *selective bibliographies*, given document texts are split into sequences of words based on phrase delimiters, stop-words and POS patterns, instead of dependency parsing, and each sequence is considered as a keyword candidate[28].

6.2 Keyword Candidate Ranking

6.2.1 Preliminaries. To clarify the problem, some notations and functions are introduced in this subsection.

Let $D = \{d\}$ denote the training document set, and $l_d \in \{0,1\}^{n(d) \times 1}$ be the indicator vector of document d, where $l_{dp} = 1$ means that candidate p in document d is a keyword and, $n(d)$ represents the number of candidate keywords in document d. F denotes the manually-defined feature set.

In addition, some functions are defined. Given an indicator vector l_d and a score sequence of keyword candidates $l_d\prime$ of a document d, one evaluation metric, *Averaged Precision*, is defined as, $E(l_d, l_d\prime) = \frac{\sum_{p \in d} P_d(p) l_{dp}}{\sum_{p \in d} l_{dp}}$, in which $P_d(p) = \frac{\sum_{q: r_d(q) < r_d(p)} l_{dq}}{r_d(p)}$ represents the precision calculated by top $r_d(p)$ candidates, and $r_d(p)$ represents the rank of candidate p in document d.

6.2.2 Features. To rank a list of candidates, a discriminative scoring function that evaluates the importance of each candidate is needed. Particularly, in this study, four features are reckoned to build such scoring function, i.e.,

1. Average embedding vector of words in a candidate;
2. The RAKE score[28] of a candidate;
3. The *tf-idf* value of a candidate;
4. Proportion of uppercase word in the candidate.

These four features are specially designed for short and ungrammatical corpus like *selected bibliographies* since that, they are independent with grammar or syntactic rules of corpus. Pre-trained embedding vectors capture the semantic meanings of a candidate, while both RAKE scores and *tf-idf* values are statistical features that consider the importance of keyword candidates based on mutual information in a document. Additionally, the proportion of uppercase word intends to model the observation that proper noun usually appears in a uppercase form.

6.2.3 Algorithm. Inspired by AdaRank and SVM-based AdaBoost, a novel ranking method is proposed based on Adaptive Boost algorithm and Logistic Regression which is referred as AdaLogistic. To be clear, AdaLogistic is no more than a framework, besides four defined features in the previous subsection, some task-specific features can be defined and extracted to enhance the overall performance.

As shown in Algorithm 1, AdaLogistic takes a training document set $D = \{d\}$ as input and pre-defined functions E, Δ, Ψ and feature set F as parameters. AdaLogistic runs $|F|$ rounds, and a weak ranker ϕ_i $(i = 1, 2, \cdots, |F|)$ is trained in each round. The final strong ranker Φ is output as the weighted linear combination of the weak rankers, in which the weight of weak ranker ϕ_i, denoted as α_i, represents the goodness of ϕ_i in terms of the performance measure E. Additionally, distributions of weights over documents in the training dataset D as well as phrases in a document d, denoted as W_{Dd} and W_{dp} respectively, are maintained at each round of AdaLogistic, so that documents and phrases that are of better discriminating abilities are focused in the next round of training.

Algorithm 1 AdaLogistic Algorithm

```
Require: The training document set, $D = \{d\}$; Pre-defined functions, $E, \Delta, \Psi$; Pre-defined feature set, $F$;
Ensure: A ranked list of keywords, $K = \{k\}$.
1: Initialize weight for each document $d$ in set $D$ as $W_D = \{W_{Dd}^{(0)} \mid W_{Dd}^{(0)} = \frac{1}{n}, d \in D\}$; and weight for each phrase $p$ in a specific document $d$ as $W_d = \{W_{dp}^{(0)} \mid W_{dp}^{(0)} = \frac{1}{n(d)}, p \in d\}$
2: for each $i$ in $1 : |F|$ do
3:     Train a weak ranker $\phi_i$ based on loss function $\mathcal{L}$ and the $i^{th}$ feature, and calculate the corresponding score vector $l_d^{(i)}\prime$, in which $\phi_i\left(z^{(i)}\right) = \frac{1}{1+\exp\left(-\left(\omega_i z_{dp}^{(i)} + \beta_i\right)\right)}$ and $\mathcal{L} = \sum_{d \in D} W_{Dd}^{(i)} \sum_{p \in d} W_{dp}^{(i)} \left(l_{dp} - \frac{1}{1+\exp\left(-\left(\omega_i z_{dp}^{(i)} + \beta_i\right)\right)}\right)^2$
4:     Calculate the evaluation metric $E_{\phi_i} = \sum_{d=1}^{n} W_{Dd}^{(i)} E_i\left(l_d, l_d^{(i)}\prime\right)$;
5:     Update the weight of $i^{th}$ weak ranker $\alpha_i = \frac{1}{2} \ln \frac{1+E_{\phi_i}}{1-E_{\phi_i}}$;
6:     Update the weight of document $W_{Dd}^{(i)} = \frac{W_{Dd}^{(i-1)} E_i\left(l_d, l_d^{(i)}\prime\right)}{Z}$, in which $Z$ represents the normalization factor;
7:     Update the weight of phrase $W_{dp}^{(i)} = \frac{\sum_{q: r_d(q) > r_d(p)} \Delta_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right) \Psi_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right)}{Z}$, in which $Z$ represents the normalization factor;
8: end for
9: Obtain the strong ranker $\Phi = \sum_{i=1}^{|F|} \alpha_i \phi_i\left(z^{(i)}\right)$.
```

In Algorithm 1, first of all, the weights for each document d and phrase p in d, denoted as $W_{Dd}^{(0)}$ and $W_{dp}^{(0)}$, are initialized equally in *line 1*. As illustrated in *line 2* and *line 3*, the algorithm runs $|F|$ times, and at the i^{th} round, a weak ranker ϕ_i is built based on i^{th} feature. In this way, all the features are used iteratively. Once ϕ_i is trained, the performance of this weak ranker is evaluated as E_{ϕ_i} in *line 4*, and the corresponding weight α_i of ϕ_i is calculated according to E_{ϕ_i} in *line 5*. Intuitively, a weak ranker that achieves a higher score in terms of pre-defined evaluation metrics should be given a higher weight, which is the basic idea of Adaptive Boost algorithm. To get prepared for the next round, weights for each document d and phrase p in d are updated in *line 6* and *line 7*, so that discriminative training samples are given higher weights and their losses are assigned to more importance in the next round. Specifically, *line 7* focuses on the weight of each candidate keyphrase, in which Δ represents the cost shown in evaluation metrics of ranking a pair of phrases p and q incorrectly, while Ψ describes the score difference between a pair of phrase p and q. Finally, a strong ranker is obtained as the weighted linear combination of weak rankers, and the ranked keyword candidate list can be obtained. With the strong ranker, a list of candidates that are extracted from *selective bibliographies* of renowned scholars can be ranked according to the probability of being a concept word.

For the maintenance of weights over documents, inspired by the updating function utilized in AdaBoost that aims to accumulate the influence of former weak rankers[8], the updating scheme is defined as $W_{Dd}^{(i)} = \frac{W_{Dd}^{(i-1)} E\left(l_d, l_d^{(i)\prime}\right)}{Z}$ in *line 6*. In addition, unlike the AdaRank algorithm, the weight in phrase-grain is also taken into considerations in *line 7*, to alleviate the negative impact brought by noise terms that have poor discriminating abilities. Under this motivation, $\Delta_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right)$, denoting the cost of sorting candidate p and q incorrectly, is defined as $\Delta_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right) = E_i\left(l_d, l_d^{(i)\prime}\right) - E_i\left(l_d, l_d^{(i)\prime}(p/q)\right)$, in which z_{dp} denotes the feature values of phrase p in document d, and $l_{d'}(p/q)$ is the score sequence obtained by swapping candidates p and q. $\Psi_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right)$, defined as $\Psi_i\left(z_{dp}^{(i)}, z_{dq}^{(i)}\right) = \ln\left(1 + \exp\left(\phi_i\left(z_{dp}^{(i)}\right) - \phi_i\left(z_{dq}^{(i)}\right)\right)\right)$, is to tune the weight assigned to candidate p according to the difference between score of p and other candidates that rank lower. In one word, the updating scheme of phrase-grain weights in *line 7* is under the intuition that if candidate p ranks higher than candidate q, the bigger the difference between their scores, i.e. $\phi_i\left(z_{dp}^{(i)}\right)$ and $\phi_i\left(z_{dq}^{(i)}\right)$, the higher the weight should be given to candidate p, based on the loss of failure to rank candidate p and q correctly.

7 EXPERIMENTS

7.1 Dataset

In phase I, the *infobox* structure in Wikipedia is utilized to label correlated entities to generate a training dataset for CRF. Accordingly, 957 sentences that describe famous scholars' *alma maters* and *institutions* are gained from Wikipedia as training dataset and, each sentence has a labeled *'organization'* entity as mentioned in Section 4. To build the scholar profile in a specific domain, the trained CRF model is firstly utilized to detect alma maters and institutions without distinguishing them from the Wiki-page of 29 renowned information scientists, which are carefully selected according to the result of RDF query in Wikidata mentioned in Section 1. Next, the coarse-grained results are utilized as the training corpus for TransP. In this way, 130 related entities and 144 triplets are obtained to train a TransP model. With the TransP model, embedding vectors for entities and relations to be further identified are generated. Finally, embedding vectors are clustered and, clusters are assigned with predicted labels according to pre-defined seeds. In this experiment, 130 *organizations* are manually divided into *alma maters* and *institutions* according to the local context for evaluation.

In phase II, *selective bibliographies* in the introductions of the 29 renowned information scientists in Wikipedia are manually double-checked to label keywords by postgraduates majored in Information Science who have a solid understanding of critical concepts in this domain, referred as Dataset 1. Besides, maui-semeval 2010 is utilized as the benchmark dataset for Keyword extraction, which contains 244 papers in computer science as well as corresponding keywords. This dataset is divided randomly into a training set that contains 200 papers and a testing set that contains 44 papers, with 6718 candidates obtained for training and 1414 for testing.

7.2 Evaluation Metrics

In this study, *precision*, *recall* and *F1* metrics are utilized to evaluate the performance of TransP model and AdaLogistic algorithm.

In addition, *Mean Average Precision* is also utilized to evaluate the ranking performance of AdaLogistic, which is defined as:

$$MAP = \frac{1}{|D|} \sum_{d \in D} AP_d \tag{5}$$

AP_d is defined as $AP_d\left(y_d, y_{d'}\right) = \frac{\sum_{j=1}^{n(d)} P_d(j) y_{dj}}{\sum_{j=1}^{n(d)} y_{dj}}$, and P_d is defined as $P_d(j) = \frac{\sum_{k: r_d(k) < r_d(j)} y_{dk}}{r_d(j)}$. Intuitively, $P_d(j)$ represents the precision calculated by top j candidates, and AP_d means the average precision of observed samples once a positive sample is recalled.

7.3 Result Analysis

7.3.1 Entity Typing. First of all, the benefit of embedding vectors for typing is presented by an visualization technique. To visualize the distribution of learned embedding vectors, which is in multi-dimensional space, a Dimension Reduction method is utilized to project vectors into a 2-dimensional space.

For instance, Figure 3 presents the distribution of *alma maters* and *institutions* by visualizing vectors with pre-trained word embedding vectors and embedding vectors from TransP respectively, given that vectors are projected into a 2-dimensional space. As seen from the figure of Default Vectors, with improper entity embedding, centers of different clusters are close to each other without a visual margin, which imposes much more difficulties on entity typing algorithms. In contrast, vectors of *alma maters* and *institutions* can be easily distinguished by a linear hyperplane if they are embedded by TransP, as shown in the figure of Embedded Vectors.

Despite of the fact that k-means achieves state-of-the-art performance as a simple clustering method, its performance can be badly influenced by the choice of hyper-parameter, k. Thus, sensitivity analysis is firstly conducted before the evaluation on Entity Typing. Figure 4 shows that the performance is not sensitive much to the choice of k, and the best choice of k should be 3 or 4. In this way, all the relation vectors are divided into 4 groups by k-means, and 7 pre-defined seeds are used to merge the groups into 2 clusters. Expected labels are given to each cluster according to the seeds as well. In addition, the Canopy algorithm is conducted to generate initial cluster centers for k-means, as illustrated in Section 5.

The overall performance of Entity Typing is presented in Table 2. It can be seen that TransP outperformed other translating models in the specific task of Entity Typing notably. Also, it implies that, without the considerations of the semantic distance between entities and relations, the performances of other Trans models are not improved obviously from TransE to TransD. In addition, due to the poor quality of limited number of training data obtained from coarse-grained entity detection, the number of parameters to be trained can also influence the over performance. As shown in Table 2, the simplest translating model with the smallest number of parameters, TransE, outperformed other models, except for TransP.

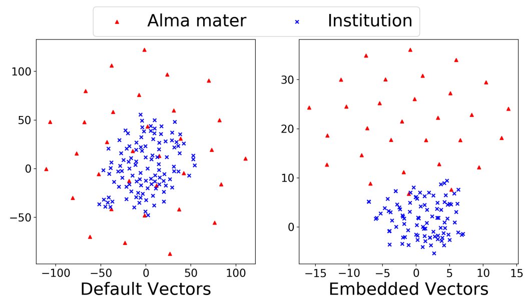

Figure 3: 2-Dimensional Distribution of Vectors

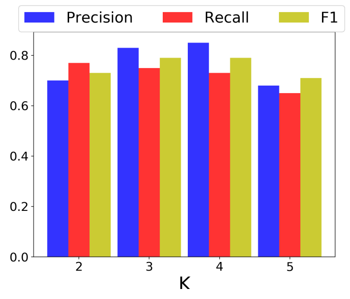

Figure 4: Sensitivity Analysis of K-means

This side-result is also supported by comparing the performance of TransD with TransR. TransD outperforms TransR and has a comparable performance with TransE model, probably because TransD has less parameters to train and has no matrix multiplication to operate, which makes it more robust to the quality of training dataset.

Table 2: Performance of Entity Typing

Models	Precision	Recall	F1
TransE	0.56	0.57	0.56
TransP	**0.85**	**0.73**	**0.78**
RESCAL	0.49	0.49	0.49
TransH	0.45	0.44	0.44
TransR	0.49	0.48	0.48
TransD	0.56	0.56	0.56

7.3.2 Keyword Extraction. The proposed AdaLogistic algorithm is applied for Keyword Extraction by ranking keyword candidates according to the score describing how likely a candidate could be selected as a keyword. Besides, a threshold is needed to decide the proportion of candidates, which are convincing enough to be selected as a keyword. Thus, a series of experiments about sensitivity analysis were conducted on two datasets to decide the threshold value of the ranking-based keyword extraction method and, results are presented in Figure 5. Considering the generality, in the following experiments, the threshold is selected as 33% where all evaluation metrics gain considerable values.

In addition, the Precision-Recall Curves of different keyword extraction methods on both specific task and benchmark dataset are also presented in Figure 6. As seen from these figures, compared with other four benchmark algorithms, the proposed AdaLogistic algorithm performs the best. Also, on Dataset 1, in which many sentences are short and ungrammatical, RAKE score achieved much better performance than other baseline methods. It implies that statistical features like RAKE score are especially suitable for ungrammatical short corpora, which supports the utilization of RAKE score as a critical feature in AdaLogistic. While on maui-semeval 2010, where the corpora are generated from academic papers in computer science and obey common grammar rules, the advantage of RAKE score is not obvious.

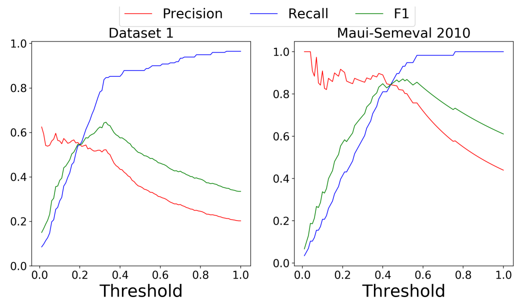

Figure 5: Sensitivity Analysis Based on Precision, Recall, F1

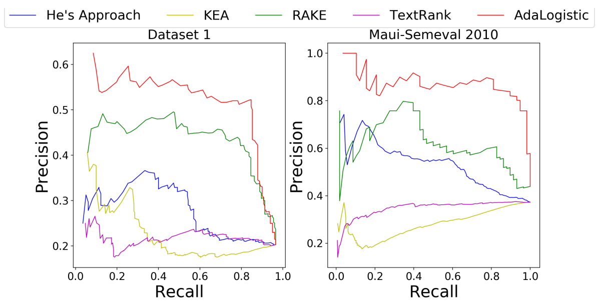

Figure 6: P-R Curves on Dataset 1 and Maui-Semeval 2010

With top 33% ranked candidates selected, AdaLogistic was evaluated on both Dataset 1 and maui-semeval 2010, comparing to other baseline methods for Keyword Extraction. According to Table 3 and Table 4, AdaLogistic algorithm achieves the best among all other methods on both two datasets. In addition, as presented in Table 3 and Table 4, the *precision* of TextRank is improved largely, while a comparable *recall* is achieved, when it is utilized on maui-semeval 2010. Utilizing a graph-based method, the score of a keyword candidate in TextRank is calculated recursively by traversing across the whole graph through semantic links[22]. Accordingly, it is found that the performance of TextRank strongly depends on the effectiveness of links between nodes in the graph constructed from the context. Since documents in maui-semeval 2010 are generated from academic papers, semantic links between terms are much more convincing than those in Dataset 1, which results in better performance of TextRank on maui-semeval 2010.

Table 3: Performance of Keyword Extraction on Dataset 1

Models	Precision	Recall	F1	MAP
AdaLogistic	**0.52**	**0.89**	**0.65**	**0.37**
He's approach[9]	0.31	0.51	0.39	0.2
KEA	0.18	0.7	0.29	0.17
RAKE	0.46	0.78	0.58	0.33
TextRank	0.2	0.33	0.25	0.15

Table 4: Performance of Keyword Extraction on Maui-Semeval 2010

Models	Precision	Recall	F1	MAP
AdaLogistic	**0.86**	**0.66**	**0.75**	**0.79**
He's approach[9]	0.55	0.49	0.51	0.53
KEA	0.18	0.16	0.17	0.28
RAKE	0.61	0.47	0.53	0.56
TextRank	0.36	0.32	0.34	0.35

8 CONCLUSION

This study aims to present an automatic framework for scholars' profile construction, covering one's research records and famous concepts that are highly relevant to them, which helps junior researchers catch critical concepts in a particular domain. To discriminate ambiguous entity pairs, a novel embedding model named TransP is proposed, in which initial vectors and semantic distance of entities and relations are taken into considerations. Meanwhile, a Keyword Extraction algorithm based on AdaBoost algorithm and Logistic Regression is proposed to extract concepts from *selected bibliographies* of renowned scholars and return the relative importance of concepts as well. Experiments show that the newly proposed methods outperformed other benchmarked methods.

In the future, more information will be covered in the scholar profile, including interpersonal relations between renowned scholars, like co-authors, colleagues etc., and concepts from relevant disciplines that share common research interests with information science domain, like computer science, mathematics etc.

9 ACKNOWLEDGEMENTS

The work described in this paper was supported by Grant from the National Nature Science Foundation of China (project no. NSFC 71701019/G0114) and National Social Science Foundation of China (Grant.19ATQ005).

REFERENCES

[1] Bahram Amini, Roliana Ibrahim, Mohd Shahizan Othman, and Mohammad Ali Nematbakhsh. 2015. A reference ontology for profiling scholar's background knowledge in recommender systems. *Expert Systems with Applications* 42 (2015), 913–928.
[2] Sonia Bergamaschi, Andrea Cappelli, Antonio Circiello, and Marco Varone. 2017. Conditional Random Fields with Semantic Enhancement for Named-entity Recognition. In *WIMS '17*. 28:1–28:7.
[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data, In NIPS'13. *Neural Information Processing Systems*, 1–9.
[4] Arthur Brack, Jennifer D'Souza, Anett Hoppe, Sören Auer, and Ralph Ewerth. 2020. Domain-independent Extraction of Scientific Concepts from Research Articles.
[5] Hongjie Dai, Po-Ting Lai, Yung-Chun Chang, and Richard Tzong-Han Tsai. 2015. Enhancing of chemical compound and drug name recognition using representative tag scheme and fine-grained tokenization. *Journal of cheminformatics* 7 (2015), S14.
[6] Lance De Vine, Shlomo Geva, and Peter Bruza. 2017. Efficient Analogy Completion with Word Embedding Clusters. In *ADCS'2017*. 4:1–4:4.
[7] Jianlin Feng. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *AAAI'14*. 1112–1119.
[8] Yoav Freund and Robert E Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. System Sci.* 55 (1997), 119 – 139.
[9] Guoxiu He, Junwei Fang, Haoran Cui, Chuan Wu, and Wei Lu. 2018. Keyphrase Extraction Based on Prior Knowledge. In *JCDL '18*. 341–342.
[10] Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. Large margin rank boundaries for ordinal regression. *Advances in Large Margin Classifiers* 88 (2000), 115–132.
[11] Kuo-Wei Hsu. 2017. Integrating Adaptive Boosting and Support Vector Machines with Varying Kernels. In *IMCOM '17*. 88:1–88:8.
[12] Chaoyi Huang, Youguang Chen, and Qiancheng Liang. 2019. Attention-Based Bidirectional Long Short-Term Memory Networks for Chinese Named Entity Recognition. In *ICMLT'2019*. 53–57.
[13] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *JCNLP'15*. 687–696.
[14] Xin Jiang, Yunhua Hu, and Hang Li. 2009. A Ranking Approach to Keyphrase Extraction. In *SIGIR '09*. 756–757.
[15] Wei Jin and Corina Florescu. 2018. Improving Search and Retrieval in Digital Libraries by Leveraging Keyphrase Extraction Systems. In *JCDL '18*. 419–420.
[16] John Lafferty, Andrew Mccallum, and Fernando Pereira. 2002. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML'02*. 282–289.
[17] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 260–270.
[18] Qi Li, JunQi Dong, Jiang Zhong, Qing Li, and Chen Wang. 2019. A neural model for type classification of entities for text. *Knowledge-Based Systems* 176 (2019), 122–132.
[19] Y. Lin, Zhiyuan Liu, M. Sun, Y. Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI'15*. 2181–2187.
[20] R.M. Syachrul M.A.K., Moch Arif Bijaksana, and Arief Fatchul Huda. 2019. Person Entity Recognition for the Indonesian Qur'an Translation with the Approach Hidden Markov Model-Viterbi. *Procedia Computer Science* 157 (2019), 214–220.
[21] Andrew McCallum, Kamal Nigam, and Lyle H. Ungar. 2000. Efficient Clustering of High-dimensional Data Sets with Application to Reference Matching. In *KDD '00*. 169–178.
[22] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text.. In *EMNLP'04*. 404–411.
[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML'11*. 809–816.
[24] Finn Nielsen, Daniel Mietchen, and Egon Willighagen. 2017. Scholia, Scientometrics and Wikidata. In *ESWC'17*. 237–259. https://doi.org/10.1007/978-3-319-70407-4_36
[25] Eirini Papagiannopoulou and Grigorios Tsoumakas. 2018. Local word vectors guiding keyphrase extraction. *Information Processing and Management* 54 (2018), 888–902.
[26] Qinjun Qiu, Zhong Xie, Liang Wu, and Wenjia Li. 2019. Geoscience keyphrase extraction algorithm using enhanced word embedding. *Expert Systems with Applications* 125 (2019), 157–169.
[27] Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. 2016. Label Noise Reduction in Entity Typing by Heterogeneous Partial-Label Embedding. In *KDD '16*. 1825–1834.
[28] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*. Wiley, 1–20.
[29] Pum-Mo Ryu, Myung-Gil Jang, and Hyun-Ki Kim. 2014. Open domain question answering using Wikipedia-based knowledge model. *Information Processing and Management* 50 (2014), 683–692.
[30] Jie Tang. 2016. AMiner: Mining Deep Knowledge from Big Scholar Data. In *WWW '16 Companion*. 373–373.
[31] Didier A. Vega-Oliveros, Pedro Spoljaric Gomes, Evangelos E. Milios, and Lilian Berton. 2019. A multi-centrality index for graph-based keyword extraction. *Information Processing and Management* 56 (2019), 102063.
[32] Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. 2005. Domain-specific Keyphrase Extraction. In *CIKM '05*. 283–284.
[33] Jun Xu and Hang Li. 2007. AdaRank: A Boosting Algorithm for Information Retrieval. In *SIGIR '07*. 391–398.