Context-Guided Learning to Rank Entities

Makoto P. Kato[1(✉)], Wiradee Imrattanatrai[2], Takehiro Yamamoto[3], Hiroaki Ohshima[3], and Katsumi Tanaka[2]

[1] University of Tsukuba/JST, PRESTO, Tsukuba, Japan
mpkato@acm.org
[2] Kyoto University, Kyoto, Japan
wiradee@db.soc.i.kyoto-u.ac.jp, tanaka.katsumi.85e@st.kyoto-u.ac.jp
[3] University of Hyogo, Kobe, Japan
t.yamamoto@sis.u-hyogo.ac.jp, ohshima@ai.u-hyogo.ac.jp

Abstract. We propose a method for learning entity orders, for example, safety, popularity, and livability orders of countries. We train linear functions by using samples of ordered entities as training data, and attributes of entities as features. An example of such functions is $f(\text{Entity}) = +0.5$ (Police budget) -0.8 (Crime rate), for ordering countries in terms of safety. As the size of training data is typically small in this task, we propose a machine learning method referred to as *context-guided learning* (CGL) to overcome the over-fitting problem. Exploiting a large amount of contexts regarding relations between the labeling criteria (*e.g.* safety) and attributes, CGL guides learning in the correct direction by estimating a roughly appropriate weight for each attribute by the contexts. This idea was implemented by a regularization approach similar to support vector machines. Experiments were conducted with 158 kinds of orders in three datasets. The experimental results showed high effectiveness of the contextual guidance over existing ranking methods.

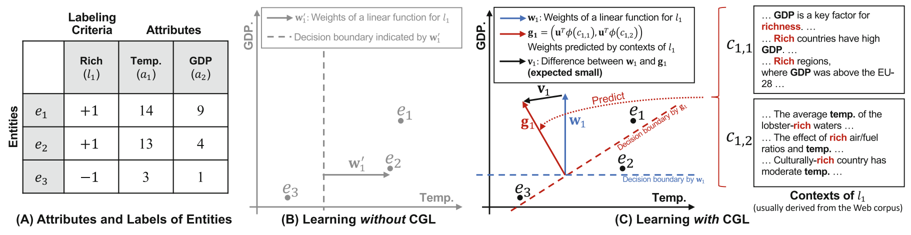
1 Introduction

Entity search is one of the emerging trends in major search engines [19,32], and has been powered by large-scale knowledge bases such as DBpedia, Wikidata, and YAGO. A wide variety of entity attributes are stored in knowledge bases and have enabled search engines to support entity search queries such as "european countries" and "movies starring emma watson".

On the other hand, the current entity search systems have not supported various kinds of rankings yet, which can be found on the Web, for example, the most *livable* countries, *innovative* companies, and *high-performance* cameras. If such diverse rankings were integrated into entity search and explained objectively with some evidences, users could be more efficient for accomplishing complex tasks such as decision making, comparison, and planning. For example, a user is planning to visit several European countries and inputs a query "european countries safety" to know how safe each country is. If an entity search engine

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12035, pp. 83–96, 2020.
https://doi.org/10.1007/978-3-030-45439-5_6

provided a list of countries ranked by public safety and factors used to determine the ranking (*e.g.* crime rate and police budget), they would be helpful for the user to make his/her travel plan.


Labeling Criteria
Attributes
Entities
(A) Attributes and Labels of Entities
$\mathbf{w}'_1$: Weights of a linear function for l_1
Decision boundary indicated by $\mathbf{w}'_1$
GDP
Temp.
e_1
e_2
e_3
$\mathbf{w}'_1$
(B) Learning *without* CGL
$\mathbf{w}_1$: Weights of a linear function for l_1
$\mathbf{g}_1 = (\mathbf{u}^T\phi(c_{1,1}), \mathbf{u}^T\phi(c_{1,2}))$ Weights predicted by contexts of l_1
$\mathbf{v}_1$: Difference between $\mathbf{w}_1$ and $\mathbf{g}_1$ (expected small)
$\mathbf{v}_1$
$\mathbf{g}_1$
$\mathbf{w}_1$
Predict
Decision boundary by $\mathbf{g}_1$
Decision boundary by $\mathbf{w}_1$
$c_{1,1}$
$c_{1,2}$
Contexts of l_1
(usually derived from the Web corpus)
(C) Learning *with* CGL

	Rich (l_1)	Temp. (a_1)	GDP (a_2)
e_1	+1	14	9
e_2	+1	13	4
e_3	−1	3	1

… **GDP** is a key factor for **richness**. …
… **Rich** countries have high **GDP**. …
… **Rich** regions, where **GDP** was above the EU-28 …

… The average **temp.** of the lobster-**rich** waters …
… The effect of **rich** air/fuel ratios and **temp.** …
… Culturally-**rich** country has moderate **temp.** …

Fig. 1. (A) Entities e_1 and e_2 are rich countries, and e_3 is not a rich country. They have only two attributes a_1 (temperature) and a_2 (GDP). (B) Every entity can be expressed as a point in a two dimensional space by their attribute values in this example. Our goal is to learn a linear function for the labeling criterion l_1, which is defined as $f_1(\mathbf{x}) = \mathbf{w}_1^T\mathbf{x}$. One of the possible weights that perfectly classify the training examples is $\mathbf{w}'_1 = (1, 0)$, but not necessarily effective for the other examples. (C) Contexts are used to produce a "rough" prediction $\mathbf{g}_1$ of the ideal weights. CGL determines the weights $\mathbf{w}_1$ such that $\mathbf{v}_1$, the difference between $\mathbf{w}_1$ and $\mathbf{g}_1$, is small and training examples are separated well. The weights $\mathbf{w}_1$ are expected to be effective for the other cases, since a strong correlation between richness and GDP is suggested by their contexts.

In this paper, we propose a method for learning orders of entities using samples of ordered entities as training data and attributes of entities as features. Entity orders are expressed in several forms on the Web: comparative sentences (*e.g.* "DiCaprio is taller than Pitt"), scores (*e.g.* "[Camera A] portrait: 9.2, landscape: 7.5, and sports: 8.5"), and rankings (*e.g.* "1st: Iceland, 2nd: Denmark, and 3rd: Austria"). These expressions can be interpreted with a uniform model, *i.e.* a subset of entity pairs that defines an entity order, and be used as training data to learn entity orders. The learned models can be used not only to rank entities but also to explain rankings by correlated attributes. We assume that entity orders can be represented as a linear function of attributes (denoted by f), primarily because of the high explanatory capacity for users. For example, given a list of entities ordered by labeling criterion "safety", (Iceland, Denmark, Austria), and their attributes such as "GDP", "Crime rate", and "Police budget", we learn function f(Entity) = +0.5 (Police budget) −0.8 (Crime rate).

A major challenge for this problem is the lack of training data. Many Web sites do not present all the ordered entities (see Table 1). Moreover, the size of training data might not be sufficiently large for some entity classes, even if all the ordered entities are described (*e.g.* only 50 states in the United States). As the number of attributes should be large enough to explain diverse orders, and can be increased easily with existing techniques [11,28], the problem of learning to rank entities can suffer from serious over-fitting problems.

To cope with this essential problem, we propose a learning method referred to as *context-guided learning* (*CGL*). This method uses not only ordered entities but also *contexts of labeling criteria and attributes* to learn the function f. A labeling criterion refers to a textual representation to determine labels (or an order in a ranking problem). The context can provide the models with additional information, and guide learning in the correct direction by preventing over-fitting. Figure 1 illustrates how CGL is applied to a classification problem. (As can be seen later, CGL is first explained for a classification problem and later extended to a ranking problem). Our goal in this example is to learn a linear function for the labeling criterion l_1 (richness), which is defined as $f_1(\mathbf{x}) = \mathbf{w}_1^T \mathbf{x}$ (an intercept is omitted for simplicity). When we simply apply an ordinary learning algorithm, learned weights can be $\mathbf{w}'_1 = (1, 0)$ in (B) of Fig. 1, indicating that the attribute a_1 is useful for this classification. Although these weights seem reasonable as their decision boundary perfectly separates positive (e_1 and e_2) and negative (e_3) examples, it is easy to anticipate that the attribute a_1 can be useless for the other cases if we know the meaning of the labeling criterion (*i.e.* richness) and attribute a_1 (*i.e.* temperature). CGL, on the other hand, incorporates contexts of the labeling criterion and attributes for making a "rough" prediction of the ideal weights, and expects the weights $\mathbf{w}_1$ to be close to the "rough" prediction (denoted as $\mathbf{g}_1$ in (C) of Fig. 1). Although the prediction based on contexts cannot be always accurate (indeed, the decision boundary of $\mathbf{g}_1$ fails to classify examples well), $\mathbf{g}_1$ suggests that the attribute a_1 is not strongly related to the labeling criterion, and guides the learning of the weights $\mathbf{w}_1$. Thus, the learning can be successful even if sufficient training data are not available. CGL does not require any annotations for the contexts. Alternatively, CGL learns multiple functions at the same time for learning the relationship between contexts and weights in the function f.

To the best of our knowledge, CGL is the first attempt to leverage contexts of labeling criteria and features directly in machine learning (ML) problems. CGL is a general ML method and can be applied not only to ranking problems but also to classification and regression problems as long as relations between labeling criteria and features are described in a particular corpus.

Our contributions in this paper can be summarized as follows: (1) we introduced the problem of learning to rank entities by using attributes as features, in order to rank entities by various criteria and precisely understand labeling criteria; (2) we proposed CGL, a general ML method using contexts of labeling criteria and features for preventing over-fitting; and (3) we conducted experiments with a wide variety of orders, and demonstrated the effectiveness of CGL in the task of learning to rank entities.

2 Related Work

We review related work on entity ranking and discuss the difference between CGL and existing ML methods, in particular, multi-task learning methods.

2.1 Entity Ranking

Entity ranking has been addressed in some tracks in INEX and TREC. The INEX Entity Ranking track held two tasks: entity ranking and entity list completion tasks [12–14]. The entity ranking task expected systems to return relevant entities in response to a given query, while the entity list completion task expected systems to return entities related to given example entities. The TREC Entity track offered related entity finding tasks, in which systems were expected to find entities related to a given entity, with the type of the target entity and nature of their relation [2–4]. Those tasks only expect that retrieved entities are ordered by the relatedness to given example entities, and do not expect different kinds of orders within related entities.

Apart from the evaluation campaigns, there are some work that addresses learning to rank entities. Kang *et al.* used a ranking algorithm based on a boosted tree model for finding entities related to a given query [24]. Tran *et al.* proposed a method of ranking entities based on salience and informativeness for timeline summarization of events [30]. Zhou *et al.* addressed a problem of finding entities that have a specified relation with an input entity [34]. They trained a ranker for each relation based on training queries and labeled entities by using features derived from search snippets regarding pairs of entities. Although this work and ours use contexts (or search snippets) for learning to rank entities, our rankers are built primarily on attributes of entities and does not use contexts of entity pairs. Jameel *et al.* proposed an entity embedding method for entity retrieval [22]. Their method is mainly based on the co-occurrence between entities and words, and does not directly model entity attributes.

Some NLP tasks are also related to our task. Iwanari *et al.* tackled a problem of ordering entities in terms of a given adjective by using some evidences extracted from texts [20]. Their task is similar to ours as both address entity ranking in terms of a particular labeling criterion. While their method uses contexts of labeling criteria and entities, our method uses contexts of labeling criteria and *attributes* of entities.

2.2 Multi-task Learning

The important characteristics of CGL are summarized as follows: (1) weights in the function f are learned based on labels as well as contexts regarding labeling criteria and features, and (2) multiple functions are learned at the same time to learn the relationship between the contexts and weights in the function f. Below, we review several ML methods and discuss their relationship to CGL.

Multi-task learning is an approach to improving learning in each task by learning multiple tasks simultaneously [9]. CGL is considered as an instance of multi-task learning. Regularized multi-task learning, which was proposed by Evgeniou and Pontil, assumes that weights of multiple tasks are similar [15]. As explained later, their model is a special case of our model when contexts are all the same. Other models assume that weights are sampled from a common prior [10,27,33]. Argyriou *et al.* used an assumption that weights are represented in a low subspace common to multiple tasks [1]. In contrast to these methods using an assumption that all the tasks are related, some work selectively decides which tasks are related and are expected to share similar weights [21,25]. Similarly, CGL uses contexts to measure the similarity between tasks implicitly, and tends to estimate similar weights for similar tasks. An interesting difference between CGL and the other multi-task learning methods is that *CGL still works even if any pairs of tasks are not similar*. CGL only requires that some contexts are similar among multiple tasks. Thus, the applicable scope of CGL is not limited to problems targeted by existing multi-task learning methods.

3 Methodology

In this section, we first explain the problem of learning to rank entities from samples of ordered entities with attributes. We then introduce CGL, apply it to our problem, and explain some approaches to modeling contexts for CGL.

3.1 Problem Definition

Letting E be a set of entities of a particular class, we define an entity order as a total order on E, denoted by $\preceq_k$. Each order has a labeling criterion (or an ordering criterion in this case) denoted by l_k. For example, labeling criteria could include "livability", "innovativeness", "beauty", and "performance". A set of all $(e_i, e_j) \in E \times E$ for which $e_i \preceq_k e_j$ holds is called a graph[1] of an entity order, denoted by $G_{\preceq_k}$. Orders are usually expressed on the Web as subsets of their graphs. Thus, we can observe and use only $G'_{\preceq_k} \subseteq G_{\preceq_k}$ for learning entity orders. For example, a ranking of safe countries "1st: Iceland, 2nd: Denmark, and 3rd: Austria" implies $G'_{\preceq_k} = \{$("Denmark", "Iceland"), ("Austria", "Iceland"), ("Austria", "Denmark")$\}$ and $l_k =$ "safety".

Our principal purpose is to learn a linear function $f_k(\mathbf{e}_i) = \mathbf{w}_k^T \mathbf{e}_i$ based on a subset of a graph $G'_{\preceq_k}$ for each entity order $\preceq_k$, where $\mathbf{e}_i$ is an M-dimensional vector representing attributes of entity $e_i \in E$, and the d-th value of the vector represents a value of attribute a_d. We expect that the function f_k *preserves* the entity order $\preceq_k$: $e_i \preceq_k e_j \Rightarrow f_k(\mathbf{e}_i) \leq f_k(\mathbf{e}_j)$ for any $e_i, e_j \in E$, so that entities can be ranked by entity order $\preceq_k$ with learned function f_k. Moreover, attributes whose weights are non-zero are expected to explain the entity order well.

As we explained earlier, the key challenge of this problem is lack of training data: $|G'_{\preceq_k}|$ is typically small compared with the number of attributes M. For example, $M = 83$ for countries and $M = 137$ for cities in our experiments. Ranked lists of ten or fewer entities can provide only at most 45 entity pairs as training data, which are not considered as sufficiently large for learning. Moreover, M must be as large as possible for modeling a wide range of orders. Thus, some approaches are necessary for preventing the over-fitting problem caused by lack of training data.

[1] *Graph of a function*, a subset of the Cartesian product of two sets defining an order.

The key idea in our work is to use data other than $G'_{\preceq_k}$ for learning $\mathbf{w}_k$ effectively. One of the unique characteristics or assumptions in our problem is that textual representations for labeling criteria and attributes are available. Therefore, given a labeling criterion, it is possible to estimate a roughly appropriate weight for each attribute by leveraging the contexts regarding relations between the labeling criterion and attribute. This idea is instantiated as CGL, which is explained in the next subsection.

3.2 Context-Guided Learning

We introduce CGL, our proposed learning method that leverages contexts of labeling criteria and features. We begin with CGL for classification problems and then extend it to be used for ranking problems.

The input for a classification problem is $\mathcal{D} = \{D_k\}_{k=1}^K$, where $D_k = \{(\mathbf{x}_{k,i}, y_{k,i})\}_{i=1}^{N_k}$, $\mathbf{x}_{k,i} \in \mathbb{R}^M$, $y_{k,i} \in \{-1, +1\}$, K is the number of labeling criteria, and N_k is the number of examples for the k-th labeling criteria. Labeling criterion l_k is a textual representation to determine values for $y_{k,i}$. For example, if $\mathbf{x}_{k,i}$ represents a feature of a city and $y_{k,i} = +1$ if the city is a metropolitan city, the labeling criterion l_k could be "metropolitan city". Another example can be found in Fig. 1. The d-th value of a vector should correspond to a particular feature and have a name denoted by a_d. Example names include "population" and "GDP".

The requirements for CGL are summarized as follows: (1) A labeling criterion l_k is expressed in language, (2) Features $A = \{a_d\}_{d=1}^M$ are expressed in language, and (3) There is a corpus including contexts regarding relations between labeling criteria and feature names. It is not necessary that all the labeling criteria and feature are expressed in language. In contrast to multi-task learning, *CGL does not require that tasks (or labeling criteria in CGL) are similar.*

A classification problem can be formalized as learning function f_k for each labeling criterion $k = 1, \ldots, K$ such that $f_k(\mathbf{x}_{k,i}) \simeq y_{k,i}$. To solve this problem, we use a linear function $f_k(\mathbf{x}_{k,i}) = \mathbf{w}_k^T \mathbf{x}_{k,i}$. Letting $c_{k,d}$ represent contexts for labeling criterion l_k and feature $a_d \in A$, we can use the contexts for estimating $\mathbf{w}_k$ as follows:

$$w_{k,d} = \mathbf{u}^T \phi(c_{k,d}) + v_{k,d}, \tag{1}$$

where $w_{k,d}$ is the d-th value of $\mathbf{w}_k$, ϕ is a feature map function that transforms a context to a vector, and $\mathbf{u}$ is a weight vector that does not depend on labeling criteria. The equation above indicates that the weight for the labeling criterion l_k and feature a_d is estimated by their context $c_{k,d}$ and an intercept $v_{k,d}$. Equation 1 is generalization of $w_{k,d} = z_d + v_{k,d}$ in the regularized multi-task learning [15], where z_d is a weight common to multiple tasks. Equation 1 is reduced to their model if all the contexts are the same. If contexts for two labeling criteria are similar, or equivalently, labeling criteria are similar, $w_{k,d}$ tends to be similar for these labeling criteria. This property is similar to some multi-task learning methods [21,25].

Based on Eq. 1, $\mathbf{w}_k$ can be expressed as follows:

$$\mathbf{w}_k = \mathbf{g}_k + \mathbf{v}_k, \tag{2}$$

where $\mathbf{v}_k = (v_{k,1}, \ldots, v_{k,M})$, $\mathbf{g}_k = \varPhi_k^T \mathbf{u}$, and $\varPhi_k = (\phi(c_{k,1}), \ldots, \phi(c_{k,M}))$. This equation is illustrated in (C) of Fig. 1. We expect that the "rough" prediction $\mathbf{g}_k$ can be given by contexts of labeling criterion l_k, and ideal weights are close to $\mathbf{g}_k$; in other words, $\mathbf{v}_k$ is *not large*.

We propose to learn the linear function using a regularization approach similar to support vector machines (SVMs) and the regularized multi-task learning [15]. The optimization problem is shown below:

Problem 1

$$\min_{\mathbf{u}, \mathbf{v}_k, \xi_{k,i}} \|\mathbf{u}\|^2 + \frac{c}{K} \sum_{k=1}^{K} \|\mathbf{v}_k\|^2 + C \sum_{k=1}^{K} \sum_{i=1}^{N_k} \xi_{k,i}, \tag{3}$$

subject, for $k = 1, \ldots, K$ *and* $i = 1, \ldots, N_k$*, to the constraints that* $y_{k,i} f_k(\mathbf{x}_{k,i}) \geq 1 - \xi_{k,i},\ \xi_{k,i} \geq 0$*, where* c *and* C *are hyper parameters.*

Slack variables $\xi_{k,i}$ measure the error of the linear functions on the training data, while the other terms are regularization terms for the weights $\mathbf{u}$ and $\mathbf{v}_k$. Hyper parameters c and C can control the effect of the contexts on the model and the sensitivity for the error on the training data: a large value for c increases the effect of the contexts, while a large value for C tends to inhibit misclassification of the training data. We learn multiple functions f_k for $k = 1, \ldots, K$ with the single objective function so that we can learn the weight $\mathbf{u}$ based on the whole training data.

We show that Problem 1 can be solved in the same manner as would be used with the standard SVM. To this end, we first define a single function to be learned that summarizes functions f_k for $k = 1, \ldots, K$ as $F(\mathbf{x}, k) = f_k(\mathbf{x})$. This function, $F : \mathbb{R}^M \times \{1, \ldots, K\} \to \mathbb{R}$, can be written as a linear function:

$$F(\mathbf{x}, k) = \mathbf{w}^T \psi(\mathbf{x}, k), \tag{4}$$

by using the following settings:

$$\mathbf{w} = (\mathbf{u}^T, \sqrt{\frac{c}{K}} \mathbf{v}^T)^T,\ \psi(\mathbf{x}, k) = ((\varPhi_k \mathbf{x})^T, \underbrace{\mathbf{0}^T, \ldots, \mathbf{0}^T}_{k-1}, \sqrt{\frac{K}{c}} \mathbf{x}^T, \underbrace{\mathbf{0}^T, \ldots, \mathbf{0}^T}_{K-k})^T, \tag{5}$$

where ψ is a feature map function, and $\mathbf{0}$ is an M-dimensional vector whose values are all zeros.

Reassigning $\mathbf{x}_i$ to $\mathbf{x}_{k,i'}$, y_i to $y_{k,i'}$, and ξ_i to $\xi_{k,i'}$ ($i = \sum_{k'=1}^{k-1} N_{k'} + i'$), we can reduce Problem 1 to the standard SVM problem, as follows.

Theorem 1. *The optimization of Problem 1 is equivalent to solving the following problem:*

Problem 2. *Given* $D = \{((\mathbf{x}_i, k_i), y_i)\}_{i=1}^{N}$ *where* $N = \sum_{k=1}^{K} N_k$ *such that* $D = \bigcup_{k=1}^{K} \left\{((x_{k,i}, k), y_{k,i}) \middle| (x_{k,i}, y_{k,i}) \in D_k\right\}$,

$$\min_{\mathbf{w},\xi_i} \frac{1}{2}\|\mathbf{w}\|^2 + C' \sum_{i=1}^{N} \xi_i, \tag{6}$$

subject, for $i = 1, \ldots, N$, *to the constraints that* $y_i F(\mathbf{x}_i, k_i) \geq 1 - \xi_i,\ \xi_i \geq 0$, *where* $C' = C/2$ *and* ξ_i *is a slack variable for* $((x_i, k_i), y_i) \in D$.

Proof. The norm of $\mathbf{w}$ is $\|\mathbf{w}\|^2 = \|\mathbf{u}\|^2 + \frac{c}{K}\|\mathbf{v}\|^2$. Therefore, the objective function of Problem 2 is rewritten as:

$$\frac{1}{2}\left\{\|\mathbf{u}\|^2 + \frac{c}{K}\sum_{k=1}^{K}\|\mathbf{v}_k\|^2 + C\sum_{i=1}^{N}\xi_i\right\}, \tag{7}$$

which is equivalent to the objective function of Problem 1.

Since Problem 2 is the standard SVM problem, we can use the standard SVM dual problem for solving Problem 1. Furthermore, we can use an important characteristic of SVMs: *i.e.* non-linear functions can be used by means of kernels. While the linear function for classification (*i.e.* f_k) cannot be a non-linear function owing to the form of the model, we can use a non-linear function for estimating the weights based on contexts (see Eq. 1). The kernel method for CGL provides us with a wide range of choices for the representation of contexts. They can be represented as vectors, sets of vectors, trees, *etc.* as long as the kernel function is appropriately designed for two contexts.

3.3 Context-Guided Learning for Ranking

We extend CGL to the ranking problem and explain how it can be applied to the problem of ranking entities.

The input for the ranking problem is $\mathcal{D} = \{D_k\}_{k=1}^{K}$, where $D_k \subseteq \mathbb{R}^M \times \mathbb{R}^M$, and K is the number of labeling criteria. Labeling criterion l_k is a textual representation to determine the order for D_k: *i.e.* $(\mathbf{x}_{k,i}, \mathbf{x}_{k,j})$ in D_k indicates that $\mathbf{x}_{k,j}$ is higher than $\mathbf{x}_{k,i}$ in terms of the labeling criterion l_k. The d-th value of vectors in D_k must correspond to a particular feature and have a name denoted by a_d. The requirements are the same as those explained in regard to CGL for classification. A ranking problem can be formalized as a learning function f_k for each labeling criterion $k = 1, \ldots, K$ such that $f_k(\mathbf{x}_{k,j}) - f_k(\mathbf{x}_{k,i}) \simeq 1$ for $(\mathbf{x}_{k,i}, \mathbf{x}_{k,j})$ in D_k. As assumed in the classification problem, we use a linear function $f_k(\mathbf{x}_{k,i}) = \mathbf{w}_k^T \mathbf{x}_{k,i}$.

It is clear that the ranking problem can be reduced to the classification problem if we redefine D_k as follows: $D'_k = \left\{(\mathbf{x}_{k,j} - \mathbf{x}_{k,i}, 1) \middle| (\mathbf{x}_{k,i}, \mathbf{x}_{k,j}) \in D_k\right\}$, since $f_k(\mathbf{x}_{k,j} - \mathbf{x}_{k,i}) = f_k(\mathbf{x}_{k,j}) - f_k(\mathbf{x}_{k,i})$.

We can apply CGL for ranking to the problem in *Problem Definition* section by using vectors of entity pairs in $G'_{\preceq_k}$ as the training data, *i.e.* $D_k = \left\{(\mathbf{e}_i, \mathbf{e}_j) | (e_i, e_j) \in G'_{\preceq_k}\right\}$.

3.4 Context Models

Having described the learning method for the problem of ranking entities, we explain the context models used in the learning. Contexts can be a set of sentences or a set of documents regarding a labeling criterion and a feature. In this work, we describe methods of modeling contexts by using sentences retrieved from Web search results.

Given labeling criterion l_k and feature a_d, we create a query combining l_k and a_d with an AND operator, and use the query to retrieve the top $N^{(c)}$ search results using a particular Web search engine ($N^{(c)} = 500$ in our experiments). We then split snippets of the search results into sentences and find sentences including both the labeling criterion l_k and the feature a_d.

We use two basic methods for modeling sentences. One is a vector representation based on the TF-IDF weighting, and the other is a distributional representation of sentences [26]. The vector representation based on the TF-IDF weighting is sparse, and not sensitive to the order of words, but it can represent exact words appearing in the context. In contrast, the distributed representation of sentences is dense, and sensitive to the word order, but it might not retain the exact words appearing in the context.

4 Experiments

This section explains data used in the experiment, describes experimental settings, and shows the experimental results.

4.1 Data

Since there is no publicly available dataset for our task, we first explain our development of a dataset and its statistics.

Various kinds of entity orders in three datasets were mined from the Web and from magazines both automatically and manually. The three datasets include *City* (more specifically, Japanese prefectures), *Country*, and *Camera* entities, respectively. These classes were selected primarily for the following reasons: (1) availability of a wide range of entity orders, (2) availability of attributes, and (3) diversity of statistics. The language scope of our dataset was Japanese, as we used a Japanese crowd-sourcing service in the evaluation. Entity names and attribute names were Japanese and translated into English for this paper.

Entity orders were mined from Web pages for *City* and *Country* datasets, and from ten Japanese camera magazines for *Camera* dataset. The retrieved ranked lists were converted into a set of pairs for each entity orders. We excluded entity sets including less than five entities.

Attributes for *City* and *Country* datasets were mined from tables in Web documents. We chose *Web tables* as a resource for obtaining attributes because (1) the extraction method can be accurate and language-independent, and (2) standardization of numerical values was not necessary as units of numerical

Table 1. Statistics of the datasets and examples of entities, orders, and attributes.

	City	Country	Camera
# Entities	47	138	149
# Orders	64	40	54
# Entities/Order	13.3	17.7	14.4
# Attributes	137	83	16
Entity examples	Tokyo	Denmark	EOS 5DS
	Kyoto	Iceland	Nikon D3300
Attribute examples	Population	# Tourists	Resolution
	Crime rate	# Suicides	Weight
Order examples	Attractive	Livable	Portable
	Rich	Happy	Tough

Table 2. Accuracy in the three datasets (±SEM).

	Accuracy			
	City	Country	Camera	Total
RankNet [5]	0.482 (0.023)	0.478 (0.025)	0.530 (0.030)	0.497 (0.015)
RankBoost [16]	0.513 (0.028)	0.636 (0.024)	0.552 (0.036)	0.557 (0.018)
LinearFeature [29]	0.566 (0.019)	0.670 (0.024)	0.614 (0.034)	0.609 (0.015)
LambdaMART [31]	0.614 (0.021)	0.659 (0.019)	0.697 (0.024)	0.654 (0.013)
ListNet [7]	0.559 (0.020)	0.518 (0.022)	0.504 (0.031)	0.530 (0.014)
CGL (TF-IDF, Linear)	**0.661** (0.017)	0.716 (0.022)	**0.823** (0.019)	**0.730** (0.012)
CGL (TF-IDF, RBF)	**0.661** (0.019)	0.725 (0.021)	0.799 (0.019)	0.724 (0.012)
CGL (Distributed, Linear)	0.646 (0.020)	0.701 (0.023)	0.798 (0.021)	0.712 (0.013)
CGL (Distributed, RBF)	**0.661** (0.018)	**0.731** (0.022)	0.804 (0.021)	0.728 (0.013)

values are usually consistent within a table. Attributes for *Camera* dataset were scraped from Web pages of a Japanese Web site, Kakaku.com[2], which provides prices and specifications of products. All the numerical values for each attribute were normalized into $[0, 1]$.

Table 1 shows statistics and examples of entities, orders, and attributes. There are 158 entity orders in total. For most of the orders, we could not find all of the entities in a class in a ranking on the Web. There were many Web pages presenting the top three or ten entities for an order. Thus, the average number of entities per order is much less than the total number of entities.

4.2 Experimental Settings

We selected as baseline methods for this experiment some existing ranking methods that do not use contexts: (1) **RankNet** [5]: a pairwise ranking method that uses a neural network model and optimizes the cross entropy loss, (2) **RankBoost** [16]: application of AdaBoost [17] to pairwise preferences, (3) **LinearFeature** [29]: a linear feature-based model optimized by coordinate ascent, (4) **LambdaMART** [31]: a combination of the ranking model, LambdaRank [6], and the boosted tree model, MART [18], and (5) **ListNet** [7]: a listwise ranking method using a neural network model. We used these methods implemented in RankLib[3]. We used normalized discounted cumulative gain (nDCG@10) [23] as an evaluation metric to be optimized for some methods.

[2] http://kakaku.com/.
[3] https://www.lemurproject.org/ranklib.php.

We conducted experiments using the developed dataset in the following settings. For each set of ordered entities $G_{\preceq_k}$, we split entities in the set E into 50:50, E_{train} and E_{test}, and obtained training data $G_{\text{train}} = \{(e_i, e_j)|(e_i, e_j) \in G_{\preceq_k} \wedge e_i \in E_{\text{train}} \wedge e_j \in E_{\text{train}}\}$ and test data $G_{\text{test}} = G_{\preceq_k} - G_{\text{train}}$. Our task in this experiment is to learn a model based on G_{train}, and to predict the pairwise preference of e_i and e_j for $(e_i, e_j) \in G_{\text{test}}$. We measured the accuracy defined as the fraction of correctly predicted pairwise preferences. We used five-fold cross validation on entity orders *within* E_{train} *of the same dataset* to determine the best parameters for each method.

We configured CGL with the following settings. Two context models were used: **TF-IDF** and **Distributed** (distributed representation with 400 dimensional vectors). Parameters c and C were determined using the cross validation explained above. A linear kernel (**Linear**) and an RBF kernel (**RBF**) were used for the kernel in CGL.

4.3 Experimental Results

Table 2 shows the accuracy in the three datasets with the standard error of the mean (SEM). CGL in any settings were better than any of the baseline methods. Among the CGL-based methods, the best method was CGL (TF-IDF, Linear), followed by CGL (Distributed, RBF). The total improvement over the best baseline method, LambdaMart, was 11.6%. According to a randomized Tukey HSD test [8][4] ($\alpha = 0.01$), the differences between CGL (TF-IDF, Linear) and all the baseline methods were found to be statistically significant, while there was no statistically significant difference across methods based on CGL.

CGL (TF-IDF, Linear) achieved 8%, 11%, and 18% improvements over LambdaMART for *City*, *Country*, and *Camera*, respectively. We hypothesize that the quality and amount of contexts are the main factors that determine the effectiveness of CGL, based on the observation that the number of sentences used for modeling contexts per attribute was 36.0, 45.7, and 137 for *City*, *Country*, and *Camera*, respectively.

Table 3. Examples of linear functions learned by CGL, in which three attributes for the highest absolute weights are shown.

Class	Learned linear model		
City	Attractiveness = +0.035 Women's life expectancy	−0.032 # Accident fatalities	−0.031 Population/family
City	Avg. savings = −0.174 Highest temperature	+0.160 Healthy life-span	+0.148 # Country inns
Country	Reputation = +0.058 Happiness	−0.057 # Applicants for asylum	−0.045 # Suicides
Country	Peace = +0.170 Grain harvest	+0.166 GDP growth rate	−0.126 # Suicides
Camera	Operability = −0.240 Weight	−0.213 Height	+0.133 Max. shutter speed

We also conducted evaluation of the attributes used in the learned functions. Five attributes with the highest absolute weights for each entity order

[4] http://www.f.waseda.jp/tetsuya/tools.html.

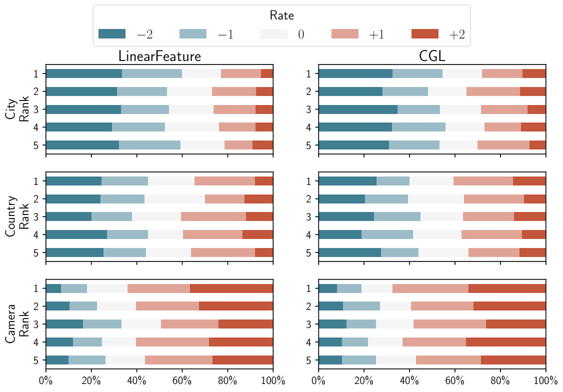


Fig. 2. Distribution of rates for five attributes with the highest absolute weights.

were pooled, and then presented to users in a Japanese crowd-sourcing service, Lancers[5]. In this evaluation, we aimed to understand to what extent the learned attributes could explain the orders. The instruction was as follows: "If you agree that there is a correlation between <labeling criterion> and <attribute>, please assign a score +2. If you disagree, please assign a score −2. If you cannot agree or disagree, please assign a score 0." Users could choose a rate from −2, −1, 0, +1, and +2. We assigned five users for each pair of a labeling criterion and an attribute. The best CGL method, CGL (TF-IDF, Linear), was selected for this evaluation. LinearFeature was used as a baseline method, since only this method used a linear function among the baseline methods.

Figure 2 shows the distribution of rates for five attributes with the highest absolute weights. The average rates of CGL were −0.455, −0.166, and +0.581, while those of LinearFeature were −0.560, −0.204, +0.516 for *City*, *Country*, and *Camera* datasets, respectively. These average rates show a high correlation with the accuracy of the models. Even though CGL could find more reasonable attributes in all of the classes than LinearFeature, their differences were small for those datasets. The average rates for *City* and *Country* datasets were negative indicating low explainability of the attributes. This is partially because some attributes only correlate to a particular labeling criterion, but were not considered as causes for increasing the criterion. Although CGL could learn a more accurate model than the baseline methods, it is still challenging to find *highly explanatory* attributes for a given label criterion.

Finally, we show some examples of linear functions learned by CGL in Table 3. Most of the attributes seem explainable and can possibly affect the entity order. While the others do not seem explanatory for the labeling criteria (*e.g.* "population/family" for "attractiveness" and "highest temperature" for "avg. savings"), they correlate well to the labeling criteria in our dataset, and are examples of attributes that were considered unreasonable in the subjective evaluation, but highly contributed to the prediction.

[5] http://www.lancers.jp/.

5 Conclusions

In this paper, we addressed the problem of learning orders of entities, by using partially observed orders as training data and attributes of entities as features. We proposed a learning method called context-guided learning (CGL) to avoid the over-fitting problem caused by lack of training data, and demonstrated the effectiveness of CGL for 158 orders in three datasets. Our future work includes theoretical analysis of CGL, application of CGL to the other problems (*e.g.* a fact verification task), exploration of better context models, and improvement of the efficiency of CGL for a large amount of data.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Numbers JP16H02906, JP17H00762, JP18H03243, and JP18H03244, and JST PRESTO Grant Number JPMJPR1853, Japan.

References

1. Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. Mach. Learn. **73**(3), 243–272 (2008)
2. Balog, K., Serdyukov, P., De Vries, A.P.: Overview of the TREC 2010 entity track. In: TREC (2010)
3. Balog, K., Serdyukov, P., De Vries, A.P.: Overview of the TREC 2011 entity track. In: TREC (2010)
4. Balog, K., De Vries, A.P., Serdyukov, P., Thomas, P., Westerveld, T.: Overview of the TREC 2009 entity track. In: TREC (2009)
5. Burges, C., et al.: Learning to rank using gradient descent. In: ICML, pp. 89–96 (2005)
6. Burges, C.J., Ragno, R., Le, Q.V.: Learning to rank with nonsmooth cost functions. In: NIPS, pp. 193–200 (2006)
7. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: from pairwise approach to listwise approach. In: ICML, pp. 129–136 (2007)
8. Carterette, B.A.: Multiple testing in statistical analysis of systems-based information retrieval experiments. ACM TOIS **30**(1), 4 (2012)
9. Caruana, R.: Multitask learning. Mach. Learn. **28**(1), 41–75 (1997)
10. Daumé III, H.: Bayesian multitask learning with latent hierarchies. In: UAI, pp. 135–142 (2009)
11. Davidov, D., Rappoport, A.: Extraction and approximation of numerical attributes from the web. In: ACL, pp. 1308–1317 (2010)
12. de Vries, A.P., Vercoustre, A.-M., Thom, J.A., Craswell, N., Lalmas, M.: Overview of the INEX 2007 entity ranking track. In: Fuhr, N., Kamps, J., Lalmas, M., Trotman, A. (eds.) INEX 2007. LNCS, vol. 4862, pp. 245–251. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85902-4_22
13. Demartini, G., Iofciu, T., de Vries, A.P.: Overview of the INEX 2009 entity ranking track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2009. LNCS, vol. 6203, pp. 254–264. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14556-8_26
14. Demartini, G., de Vries, A.P., Iofciu, T., Zhu, J.: Overview of the INEX 2008 entity ranking track. In: Geva, S., Kamps, J., Trotman, A. (eds.) INEX 2008. LNCS, vol. 5631, pp. 243–252. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03761-0_25

15. Evgeniou, T., Pontil, M.: Regularized multi-task learning. In: KDD, pp. 109–117 (2004)
16. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. J. Mach. Learn. Res. **4**, 933–969 (2003)
17. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **1**(55), 119–139 (1997)
18. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**(5), 1189–1232 (2001)
19. Guo, J., Xu, G., Cheng, X., Li, H.: Named entity recognition in query. In: SIGIR, pp. 267–274 (2009)
20. Iwanari, T., Yoshinaga, N., Kaji, N., Nishina, T., Toyoda, M., Kitsuregawa, M.: Ordering concepts based on common attribute intensity. In: IJCAI, pp. 3747–3753 (2016)
21. Jacob, L., Vert, J.p., Bach, F.R.: Clustered multi-task learning: a convex formulation. In: NIPS, pp. 745–752 (2009)
22. Jameel, S., Bouraoui, Z., Schockaert, S.: Member: Max-margin based embeddings for entity retrieval. In: SIGIR, pp. 783–792 (2017)
23. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM TOIS **20**(4), 422–446 (2002)
24. Kang, C., Yin, D., Zhang, R., Torzec, N., He, J., Chang, Y.: Learning to rank related entities in web search. Neurocomputing **166**, 309–318 (2015)
25. Kumar, A., Daumé III, H.: Learning task grouping and overlap in multi-task learning. In: ICML, pp. 1383–1390 (2012)
26. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: ICML, pp. 1188–1196 (2014)
27. Lee, S.I., Chatalbashev, V., Vickrey, D., Koller, D.: Learning a meta-level prior for feature relevance from multiple related tasks. In: ICML, pp. 489–496 (2007)
28. Madaan, A., Mittal, A., Mausam, G.R., Ramakrishnan, G., Sarawagi, S.: Numerical relation extraction with minimal supervision. In: AAAI, pp. 2764–2771 (2016)
29. Metzler, D., Croft, W.B.: Linear feature-based models for information retrieval. Inf. Retrieval **10**(3), 257–274 (2007)
30. Tran, T.A., Niederée, C., Kanhabua, N., Gadiraju, U., Anand, A.: Balancing novelty and salience: Adaptive learning to rank entities for timeline summarization of high-impact events. In: CIKM, pp. 1201–1210 (2015)
31. Wu, Q., Burges, C.J., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. Inf. Retrieval **13**(3), 254–270 (2010)
32. Yin, X., Shah, S.: Building taxonomy of Web search intents for name entity queries. In: WWW, pp. 1001–1010 (2010)
33. Yu, K., Tresp, V., Schwaighofer, A.: Learning gaussian processes from multiple tasks. In: ICML, pp. 1012–1019 (2005)
34. Zhou, M., Wang, H., Change, K.C.C.: Learning to rank from distant supervision: exploiting noisy redundancy for relational entity search. In: ICDE, pp. 829–840 (2013)