Active Learning for Multi-relational Data Construction

Hiroshi Kajino*
The University of Tokyo
7-3-1, Hongo, Bunkyo
Tokyo, Japan
hiroshi_kajino@mist.i.u-tokyo.ac.jp

Akihiro Kishimoto
IBM Research - Ireland
Damastown Industrial Estate
Mulhuddart, Dublin 15, Ireland
AKIHIROK@ie.ibm.com

Adi Botea
IBM Research - Ireland
Damastown Industrial Estate
Mulhuddart, Dublin 15, Ireland
ADIBOTEA@ie.ibm.com

Elizabeth Daly
IBM Research - Ireland
Damastown Industrial Estate
Mulhuddart, Dublin 15, Ireland
elizabeth.daly@ie.ibm.com

Spyros Kotoulas
IBM Research - Ireland
Damastown Industrial Estate
Mulhuddart, Dublin 15, Ireland
Spyros.Kotoulas@ie.ibm.com

ABSTRACT

Knowledge on the Web relies heavily on multi-relational representations, such as RDF and Schema.org. Automatically extracting knowledge from documents and linking existing databases are common approaches to construct multi-relational data. Complementary to such approaches, there is still a strong demand for manually encoding human expert knowledge. For example, human annotation is necessary for constructing a common-sense knowledge base, which stores facts implicitly shared in a community, because such knowledge rarely appears in documents. As human annotation is both tedious and costly, an important research challenge is how to best use limited human resources, whiles maximizing the quality of the resulting dataset. In this paper, we formalize the problem of dataset construction as active learning problems and present the Active Multi-relational Data Construction (AMDC) method. AMDC repeatedly interleaves multi-relational learning and expert input acquisition, allowing us to acquire helpful labels for data construction. Experiments on real datasets demonstrate that our solution increases the number of positive triples by a factor of 2.28 to 17.0, and that the predictive performance of the multi-relational model in AMDC achieves the highest or comparable to the best performance throughout the data construction process.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Parameter learning*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Semantic networks*

*JSPS Research Fellow.

WWW 2015, May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3469-3/15/05.
http://dx.doi.org/10.1145/2736277.2741103.

Figure 1: Overview of the AMDC method. Given an initial model, (i) the model asks the oracle to provide labels on a set of selected triples that will be beneficial to improve the model and the dataset, (ii) the oracle returns the labels to the model, and (iii) the model updates its dataset and retrains the model parameters on the updated dataset.

Keywords

Multi-relational data; active learning; human annotation

1. INTRODUCTION

Multi-relational data are commonly used to represent multiple types of relations between entities. In the Semantic Web, the Resource Description Framework (RDF) represents relations between Web or physical entities as directed labeled multi-graphs using a subject-predicate-object notion called a *triple*. Knowledge bases [22, 4, 12] represent information in a variety of domains, providing semantic and lexical relations between words, or relations between well-known entities such as people and places. Common sense knowledge databases are used to support AI tasks such as natural language processing, Web search, and question answering. Databases in bioinformatics [2, 1] store interactions between proteins, and the involvement of a protein into mechanisms such as the cell cycle. Such databases aim to integrate the diverse body of experimental evidence on protein-protein interactions into a single dataset enabling the scientific community to extract information providing deeper insights into the protein interaction networks.

Multi-relational learning [24, 6] is essential to increase the usefulness of multi-relational datasets. Learning a predictive

model for a database allows knowledge inference not explicitly contained in it. For example, applying multi-relational learning to bioinformatics databases can reduce the need to perform experiments, predicting the results instead.

In this paper, we consider a fundamental problem in multi-relational learning: *How can we construct a dataset and a predictive model efficiently in a cold-start setting?* Some datasets can be automatically constructed or extracted from documents [3, 12, 10]. On the other hand, many multi-relational datasets depend crucially on human expert input, given that not all knowledge may be captured in documents. For example, the fact that "a banana is yellow" may not explicitly appear in documents. Similarly, undiscovered interactions between proteins cannot appear in scientific papers. However, employing human expertise to explicitly annotate such implicit knowledge is a time-consuming and costly process. In this light, there is strong demand in developing efficient human annotation methods to construct a multi-relational dataset.

We present the *Active Multi-relational Data Construction (AMDC) method*, a method to construct a multi-relational dataset and a predictive model efficiently. The main contribution is to make full use of human expertise by harnessing multi-relational learning as guidance, as shown in Fig. 1. We exploit a model of a multi-relational dataset to predict which triples need to be labeled by human experts (the *oracle*). AMDC repeats the following procedure. It trains the model using the current dataset and calculates a *query score* of each unlabeled triple based on the model. Feedback from a human expert about a triple with a low query score is helpful to enhance the model and the dataset. Then, the unlabeled triples with the lowest query scores are provided to the oracle for labeling. Finally, given the labels on the triples, it retrains the model with the updated dataset.

We apply an active learning algorithm to a multi-relational data model trained on the ROC-AUC (the Area Under the ROC Curve) loss function. The ROC-AUC loss function is commonly used to learn from an imbalanced dataset, in which the number of positive triples is much smaller than that of negative triples. As a multi-relational dataset is often imbalanced, many multi-relational learning techniques resort to it. Despite an attempt to apply active learning to another model trained on the ROC-AUC loss function [11], it is essentially difficult to apply active learning to a model trained on the ROC-AUC loss function. Active learning requires a decision boundary to discriminate positive triples from negative ones to compute query scores, but the ROC-AUC loss function does not provide the decision boundary. We resolve the issue by combining the ROC-AUC loss function with the classification error loss function, which provides a decision boundary. This enables us to perform active learning on an imbalanced dataset efficiently. Furthermore, we devise two modifications to improve the performance of an existing multi-relational learning algorithm. First, we extend the existing ROC-AUC loss function [21, 6] to make use of positive, negative, and unlabeled triples. The existing ROC-AUC loss function compares only scores of positive and non-positive triples, and it does not distinguish negative labels from unlabeled labels. We combine the existing ROC-AUC loss function with another ROC-AUC loss function that compares scores of negative and non-negative triples to explicitly handle negative triples. Second, we add two constraints to the RESCAL model [24] to increase the stability of learning. The two constraints decrease the model complexity, which helps to avoid overfitting.

We conduct two experiments with different scenarios to demonstrate the effectiveness of AMDC. The first scenario, called a *dataset construction problem*, aims to collect as many high-confidence positive triples as possible. The second scenario, called a *predictive model construction problem*, aims to learn a multi-relational model that has a high predictive ability on unlabeled triples. For each experimental setting, we use three real datasets to evaluate AMDC. Our experiments demonstrate the advantages of AMDC as a whole and the benefits of the separate components introduced in our work.

2. PRELIMINARIES

We first give our definition of multi-relational data and assumptions on the data. Then, we introduce the oracle that answers labeling queries. Finally, we define two problem settings that differ on the objectives of the dataset construction.

2.1 Multi-relational Data

We focus on directed binary relations between two entities, which is essentially the same as the RDF representation. Let $\mathcal{E}$ be a set of entities and $\mathcal{R}$ be a set of relations. A unit of the dataset is represented by (i, j, k) $(i, j \in \mathcal{E}, k \in \mathcal{R})$. We call (i, j, k) a *triple*. A triple (i, j, k) is positive if a directed relation k holds between entities i and j, and is negative if the relation does not hold. Let Δ be the set of all the possible triples, *i.e.*, $\Delta := \{(i, j, k) \mid i, j \in \mathcal{E}, k \in \mathcal{R}\}$. Then, a multi-relational dataset over Δ is defined as below, in Def 1.

DEFINITION 1 (MULTI-RELATIONAL DATASET). *A multi-relational dataset over Δ is defined as (Δ_p, Δ_n), in which Δ_p and Δ_n are sets of positive and negative triples, respectively, such that $\Delta_p \cup \Delta_n \subseteq \Delta$ and $\Delta_p \cap \Delta_n = \emptyset$.*

The number of positive triples in a multi-relational dataset is assumed to be much smaller than the total number of triples, *i.e.*, $|\Delta_p| \ll |\Delta|$. This holds true in many multi-relational datasets; some datasets (*e.g.*, the UMLS dataset we use in the experiment) have less than 1% positive triples.

2.2 Oracle

As an interactive environment, we have B (> 0) times access to the oracle, which provides the label of a triple. Letting positive and negative labels be 1 and 0, respectively, the oracle is defined as a function $\mathcal{O} : \mathcal{E} \times \mathcal{E} \times \mathcal{R} \to \{0, 1\}$. We call B a *budget*. We assume that the oracle is always correct with its labeling.

2.3 Problem Definitions

Finally, we define two problem settings with different objectives. The *dataset construction problem* (Problem 1) aims to collect as many correct positive triples as possible. As the set of positive triples is assumed to be a small fraction of the entire dataset, collecting as many positive triples as possible is essential to construct a useful dataset.

Problem 1. Given a set of entities $\mathcal{E}$, a set of relations $\mathcal{R}$, a budget B (> 0), and the oracle $\mathcal{O}$, the *dataset construction problem* is to construct a multi-relational dataset (Δ_p, Δ_n)

within the budget such that the number of positive triples is maximized.

The *predictive model construction problem* (Problem 2) aims to construct a model with high predictive performance. In order to construct a predictive model, not only positive triples, but also negative triples are informative. Therefore, the predictive model construction problem will lead to a different consequence from the dataset construction problem. We consider the model that outputs a predictive score $s_t \in \mathbb{R}$ given a triple $t \in \Delta$ such that a triple with a high score s_t is likely to be positive, and a triple with a low score s_t is likely to be negative. The predictive performance is measured by the Area Under the ROC Curve (ROC-AUC), which will be described in Section 4.2.

Problem 2. Given a set of entities $\mathcal{E}$, a set of relations $\mathcal{R}$, a budget B (> 0), and the oracle $\mathcal{O}$, the *predictive model construction problem* is to construct a predictive model $m : \mathcal{E} \times \mathcal{E} \times \mathcal{R} \to \mathbb{R}$ within the budget such that its ROC-AUC score on unlabeled triples is maximized.

3. ACTIVE MULTI-RELATIONAL DATA CONSTRUCTION

We present the Active Multi-relational Data Construction (AMDC) method to address Problems 1 and 2. Each problem is addressed with a slightly different instantiation of the generic AMDC method. An instantiation of AMDC implements a predictive model and a query score. The predictive model is a function $m : \mathcal{E} \times \mathcal{E} \times \mathcal{R} \to \mathbb{R}$ that assigns a predictive score s_t given a triple t. We use the same predictive model for both problem settings. The query score q_t ($t \in \Delta$) is used to select which triples should be given to the oracle for labeling. It is computed for each triple based on the predictive scores. We design two different query scores for the two problem settings, as discussed later.

AMDC first initializes its predictive model and then repeats the following three steps to create a multi-relational dataset and a predictive model. First, given the predictive model, it calculates query scores of unlabeled triples. Second, triples with the lowest query scores are provided to the oracle for labeling. Finally, it updates its dataset and retrains the predictive model.

In the following, we present the details of the predictive model and the query scores and the whole procedure of AMDC.

3.1 Predictive Model

We introduce a predictive model of multi-relational data based on the RESCAL model [24]. Our model assumes that each entity $i \in \mathcal{E}$ is modeled as a latent feature vector $\mathbf{a}_i \in \mathbb{R}^D$, and each relation $k \in \mathcal{R}$ is modeled as a linear operator in the latent feature space $R_k \in \mathbb{R}^{D \times D}$, where D (> 0) is the dimension of the latent feature space. We impose two constraints on these parameters. First, we fix the length of the latent feature vector to 1. Thus, we assume that for each $i \in \mathcal{E}$,

$$\|\mathbf{a}_i\|_2 = 1, \tag{1}$$

holds. Second, we introduce a new constraint, restricting the set of relation matrices to the set of rotation matrices. Thus, we assume that for each $k \in \mathcal{R}$,

$$R_k^\top R_k = I_D, \ |R_k| = 1, \tag{2}$$

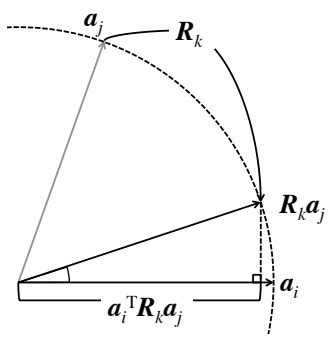

Figure 2: Geometrical illustration of our model. Positiveness of a triple $t = (i, j, k)$ is measured by the similarity between unit-length latent feature vectors $\mathbf{a}_i$ and $R_k\mathbf{a}_j$.

hold, where $I_D \in \mathbb{R}^{D \times D}$ is the D-dimensional identity matrix and $|\cdot|$ represents the determinant of a matrix. This reduces the degree of freedom of a relation matrix from $O(D^2)$ to $O(D)$. The second constraint ensures a relation matrix to be a closed operation on the ℓ_2 unit ball; for any $\mathbf{a}_j$ such that $\|\mathbf{a}_j\|_2 = 1$, $\|R_k\mathbf{a}_j\|_2 = 1$ holds. Given the latent representation, we further assume that a triple $t = (i, j, k)$ is positive if $\mathbf{a}_i = R_k\mathbf{a}_j$ holds, and the triple is negative if $\mathbf{a}_i = -R_k\mathbf{a}_j$ holds. Hence, a relation that the relation k does not hold is modeled as a relation matrix $-R_k$.

To measure the positiveness and negativeness of each triple, we define the predictive score s_t of each triple $t = (i, j, k)$ as

$$s_t = \mathbf{a}_i^\top R_k \mathbf{a}_j. \tag{3}$$

If the score is close to 1, the triple tends to be positive, and if the score is close to -1, the triple tends to be negative.

Letting $A = [\mathbf{a}_1 \ \ldots \ \mathbf{a}_{|\mathcal{E}|}]^\top \in \mathbb{R}^{|\mathcal{E}| \times D}$ be a set of latent feature vectors and $R = [R_k]_{k \in \mathcal{R}} \in \mathbb{R}^{D \times D \times |\mathcal{R}|}$ be a set of relation matrices, we call the set of model parameters (A, R), a *model*.

3.1.1 Optimization Problem for Learning

We describe our optimization problem to learn the model explained in Section 3.1. The model is learned by solving the following optimization problem:

$$\begin{aligned}\min_{A,R} &\frac{1}{|\Delta_p||\Delta \backslash \Delta_p|} \sum_{t_p \in \Delta_p} \sum_{\bar{t}_p \in \Delta \backslash \Delta_p} \left[\gamma - s_{t_p} + s_{\bar{t}_p}\right]_+ \\ &+ \frac{C_n}{|\Delta_n||\Delta \backslash \Delta_n|} \sum_{t_n \in \Delta_n} \sum_{\bar{t}_n \in \Delta \backslash \Delta_n} \left[\gamma - s_{\bar{t}_n} + s_{t_n}\right]_+ \\ &+ C_e \left(\frac{1}{|\Delta_p|} \sum_{t_p \in \Delta_p} \left[\gamma' - s_{t_p}\right]_+ + \frac{1}{|\Delta_n|} \sum_{t_n \in \Delta_n} \left[\gamma' + s_{t_n}\right]_+ \right),\end{aligned} \tag{4}$$

where $C_n, C_e, \gamma, \gamma'$ (> 0) are hyperparameters, and $[\cdot]_+ : \mathbb{R} \to \mathbb{R}$ is the hinge loss function such that

$$[x]_+ := \begin{cases} x & (x \geq 0), \\ 0 & (x < 0), \end{cases}$$

holds. Note that all the scores in the optimization problem (4) are functions of A and R as in Eq. (3).

The optimization problem comes from the following two assumptions on the scores. First, as relative relations between two scores, we assume that

$$s_{t_p} > s_{\bar{t}_p} \ (\forall t_p \in \Delta_p, \ \forall \bar{t}_p \in \Delta \backslash \Delta_p), \tag{5}$$

$$s_{t_n} < s_{\bar{t}_n} \ (\forall t_n \in \Delta_n, \ \forall \bar{t}_n \in \Delta \backslash \Delta_n), \tag{6}$$

hold. This induces the ROC-AUC loss functions, *i.e.*, the first and second terms in the objective function (4). Inequality (5) is a popular assumption when there are a small number of positive instances and a large number of unlabeled instances [26, 21, 6]. We newly introduce the relation between negative and non-negative triples as in Ineq. (6) in order to explicitly leverage negative triples. These two assumptions enable us to make full use of negative triples as well as positive triples. To the best of our knowledge, existing work does not take Ineq. (6) into account, and therefore, cannot effectively harness positive, negative, and unlabeled triples at the same time.

Second, as the sign of a score, we assume that

$$s_{t_p} > 0 \ (\forall t_p \in \Delta_p), \tag{7}$$

$$s_{t_n} < 0 \ (\forall t_n \in \Delta_n), \tag{8}$$

hold. This induces the classification error loss function, *i.e.*, the third term in the objective function (4). The second assumption enables us to discriminate positive triples from negative triples by threshold 0, while the first assumption does not provide such a decision boundary. As described later, a decision boundary is indispensable to define a query score called an uncertainty score. As far as we know, this is the first attempt to combine the ROC-AUC loss function and the classification error function to calibrate predictive scores to have threshold 0.

3.1.2 Learning Algorithm

We solve the optimization problem (4) by a stochastic gradient descent (SGD) algorithm in the same way as other related methods [26, 21, 6]. Rewriting the objective function (4) as

$$\begin{aligned} &\frac{1}{|\Delta_p||\Delta \backslash \Delta_p|} \sum_{t_p, \bar{t}_p} \left(\left[\gamma - s_{t_p} + s_{\bar{t}_p} \right]_+ + C_e \left[\gamma' - s_{t_p} \right]_+ \right) \\ &+ \frac{1}{|\Delta_n||\Delta \backslash \Delta_n|} \sum_{t_n, \bar{t}_n} \left(C_n \left[\gamma - s_{\bar{t}_n} + s_{t_n} \right]_+ + C_e \left[\gamma' + s_{t_n} \right]_+ \right), \end{aligned} \tag{9}$$

a SGD algorithm for Eq. (9) is derived as in Algorithm 1, where the Heaviside step function $H : \mathbb{R} \to \mathbb{R}$ is defined as

$$H(x) := \begin{cases} 0 & (x < 0), \\ 1 & (x \geq 0), \end{cases}$$

and for a matrix $R \in \mathbb{R}^{D \times D}$, $\mathrm{SVD}(R)$ outputs the singular value decomposition of R, *i.e.*, U, Σ, V such that $R = U \Sigma V^*$. In the SGD algorithm, we set the maximum number of updates T (> 0). We also set the learning rate at the t-th iteration as $\alpha_0 \cdot t^{-1/2}$, where α_0 (> 0) is a hyperparameter.

3.1.3 Hyperparameter Tuning

Our multi-relational learning algorithm involves several hyperparameters that have to be determined manually in order to achieve high performance. In particular in AMDC, the dataset is updated at each iteration, and therefore, the best set of hyperparameters at the beginning is not necessarily the best at the ending. We resort to validation to address this issue as shown in Algorithm 2. We prepare multiple sets of hyperparameters and a validation dataset. All the models are trained on the current training dataset, and ROC-AUC scores of all the models are calculated on the validation dataset using ValScore, which outputs ROC-AUC scores on the validation dataset. We choose the model with the best validation score to calculate the query score, which we call a *default model*.

3.2 Query Scores

We calculate a *query score* q_t for randomly sampled Q (> 0) unlabeled triple $t \in \Delta \backslash (\Delta_p \cup \Delta_n)$ and choose q ($< Q$) triples with the lowest scores for querying. We introduce two different query scores for the two problem settings.

1. Positiveness score
   Given the predictive scores $\{s_t\}_{t \in \Delta}$, the positiveness score of a triple $t \in \Delta$ is defined as
   $$q_t := -s_t.$$
   Choosing triples with the lowest query scores corresponds to choosing triples that the model believes to be positive.

2. Uncertainty score
   Given the predictive scores $\{s_t\}_{t \in \Delta}$, the uncertainty score of a triple $t \in \Delta$ is defined as
   $$q_t := |s_t| .$$
   This query score regards a triple whose predictive score close to 0 as uncertain. Such an interpretation is possible because we set a decision boundary between positive and negative triples on 0 as Eqs. (7) and (8). Querying the most uncertain samples is a standard heuristic to obtain a highly predictive model using less training data [28].

3.3 Active Learning Algorithm

We describe the entire procedure of the AMDC method in Algorithm 3. As inputs, AMDC is given a budget B (> 0), a budget for the validation set N_{val} (> 0), a small dataset $(\Delta_p^{(0)}, \Delta_n^{(0)})$, and multiple models with different hyperparameters.

The initialization consists of three parts. First, the validation set is created by randomly querying labels of N_{val} triples. Second, the models are trained on the initial dataset $(\Delta_p^{(0)}, \Delta_n^{(0)})$. Third, the validation scores of the models are calculated on the validation set, and the model with the best validation score is selected as the default model.

Then, the following steps are repeated until the budget B is exhausted. First, Q unlabeled triples are randomly sampled, the query scores are then calculated using the default model, and q triples with the lowest query scores are selected from the Q unlabeled triples. Second, the selected q unlabeled triples are provided to the oracle for labeling, and the dataset is updated using the labels given by the oracle. Third, all the models are retrained on the updated dataset, and the default model is re-selected using the validation step.

4. EXPERIMENTS

We conduct two experiments to show the effectiveness of AMDC in Problems 1 and 2. We investigate the effectiveness of our querying strategies, the constraints on the model

Table 1: Statistics of the datasets. $|\mathcal{E}|$ corresponds to the number of entities, $|\mathcal{R}|$ to that of relations, and $|\Delta_p|$ and $|\Delta_n|$ to the numbers of positive and negative triples in the original dataset.

	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\Delta_p\rvert$	$\lvert\Delta_n\rvert$
Kinships	104	26	10,790	270,426
UMLS	135	49	6,752	886,273
Nations	125	57	2,565	8,626

(Eqs. (1) and (2)), and the ROC-AUC loss function on negative and non-negative triples, which are part of the technical contributions implemented in AMDC. We first describe the datasets used in the experiments, the performance measures, evaluated methods, and the experimental setup. Then, we show experimental results on both problem settings.

4.1 Datasets

We use three real multi-relational datasets, which are also used by Bordes *et al.* [5] and Jenatton *et al.* [13]. Key statistics of the datasets are summarized in Table 1.

4.1.1 Kinships

The Kinships dataset describes kinship relations in Australian tribes that are famous for their complex kinship systems. The entities correspond to tribe members, and the relations correspond to kinship terms such as a brother, a son, and a wife (the Australian tribes often have more complex systems). A triple (i, j, k) is positive if member i calls member j by k. The dataset is fully-observed. This dataset was originally created by Denham [9], according to Kemp *et al.* [16]. We used the dataset distributed by Nickel *et al.* [24].

4.1.2 UMLS

The UMLS dataset is a biomedical semantic network constructed from the Unified Medical Language System, which was developed by McCray [20]. The entities correspond to high-level concepts, and the relations correspond to semantic relations. For example, a triple can be (Injury or Poisoning, Fully Formed Anatomical Structure, disrupts) and (Cell, Tissue, a part of) [20]. The dataset is fully-observed. We used the dataset distributed by Bordes *et al.* [6].

4.1.3 Nations

The Nations dataset contains both attributes of countries and relations between countries, which was originally curated by Rummel [27]. The entities correspond to countries, and the relations are, for example, the economical aid relation and the official visit relation. This dataset is partially-observed; there are positive, negative, and unlabeled triples in the original dataset. We only allow queries on the positive and negative triples in our experiments. We used the dataset distributed by Bordes *et al.* [6].

4.2 Performance Measures

The ROC-AUC is used to evaluate the performance of a predictive model. Given a test dataset $(\Delta_p^{(t)}, \Delta_n^{(t)})$, the ROC-AUC is calculated as

$$\frac{1}{\left|\Delta_p^{(t)}\right|\left|\Delta_n^{(t)}\right|}\sum_{t_p\in\Delta_p^{(t)}, t_n\in\Delta_n^{(t)}}\mathbb{I}[s_{t_p} > s_{t_n}],$$

where $\mathbb{I}[\text{condition}] = 1$ holds if the condition is true, and $\mathbb{I}[\text{condition}] = 0$ holds otherwise. In addition, the ratio of the positive triples the algorithm has collected so far to all the triples is used to evaluate the ability of the algorithm to collect positive triples. We call the latter performance measure a *completion rate*.

4.3 Methods Evaluated

Besides the fully-fledged AMDC, we evaluate three methods obtained by turning off, in each case, one feature of the full AMDC. AMDC_rand turns off our method of selecting the triples to be posed to the oracle, using a random querying strategy instead. AMDC_pos_only turns off the ROC-AUC loss function on negative and non-negative triples. AMDC_no_const ignores the additional constraints integrated into the model (Eqs. (1) and (2)). AMDC_rand can be derived by setting $Q = q$; we simply query labels on randomly sampled unlabeled triples. AMDC_pos_only corresponds to AMDC with $C_n = 0$. AMDC_no_const can be derived by skipping lines 13, 17, and 18 in Algorithm 1. However, without any constraints on the model parameters, the learning algorithm tends to be unstable. Therefore, we instead use regularized updates in Algorithm 1 by using regularization hyperparameters λ_A and λ_R for A and R, respectively. For example, the update rule in line 6 in Algorithm 1,

$$\mathbf{a}_{i_p} \leftarrow \mathbf{a}_{i_p} - \alpha\left[-(I_1 + I_2 C_e)\bar{R}_{k_p}\bar{\mathbf{a}}_{j_p}\right],$$

is modified into

$$\mathbf{a}_{i_p} \leftarrow \mathbf{a}_{i_p} - \alpha\left[-(I_1 + I_2 C_e)\bar{R}_{k_p}\bar{\mathbf{a}}_{j_p} + \lambda_A\bar{\mathbf{a}}_{i_p}\right].$$

We fix $\lambda_A = \lambda_R = 0.01$ in our experiments.

4.4 Protocol and Configuration

We design a protocol of an experiment as follows. Given the original dataset (Δ_p, Δ_n), we randomly divide the dataset into a test set $(\Delta_p^{(t)}, \Delta_n^{(t)})$ consisting of N_{test} triples, a validation set $(\Delta_p^{(v)}, \Delta_n^{(v)})$ consisting of N_{val} triples, the initial dataset $(\Delta_p^{(0)}, \Delta_n^{(0)})$ consisting of q triples, and the pool dataset $(\Delta_p^{(p)}, \Delta_n^{(p)})$ without overlaps between each other. Then, we run Algorithm 3 using the validation set, the initial dataset, and the pool dataset. Each algorithm is allowed to query the oracle, requesting labels for triples in the pool dataset. At each iteration of Algorithm 3, we calculate the performance measures introduced in Section 4.2, using the test set in case of the ROC-AUC score. We repeat this process 10 times and report the mean and the standard deviation of the scores at each iteration. For each dataset, we use the same experimental settings among the four tested methods, to evaluate the contribution of each individual feature turned on and off.

For the Kinships and UMLS datasets, we set $N_{\text{test}} = N_{\text{val}} = 1,000$, $B = 15,000$, $Q = 100,000$, and $q = 1,000$, and for the Nations dataset, we set $N_{\text{test}} = N_{\text{val}} = 100$, $B = 1,500$, $Q = 2,000$, and $q = 100$. This is because the number of labeled triples in the Nations dataset is much smaller than those in the Kinships and UMLS datasets. Other settings are the same across different datasets. The hyperparameter setting of the original AMDC method is $C = [5.0, 10.0, 30.0]$, $C_n = [1.0, 5.0, 30.0]$, $\gamma = 0.3$, $\gamma' = 0.9$, $D = [5, 10]$, $\alpha_0 = 0.1$, and $T = 50,000$. We use all the combinations of these subsets of hyperparameters; therefore the number of models L is 18. We override the hyperparameter setting when we use the other partial methods.

4.5 Experimental Results

We provide experimental results on the two problem settings in this section. We then discuss the properties of AMDC based on the experimental results.

4.5.1 Dataset Construction Problem

We provide experimental results as to the dataset construction problem (Problem 1). We run the four methods using the positiveness score to collect as many positive triples as possible. Figure 3 shows experimental results.

These charts emphasize four main conclusions. First, our querying strategy with the positiveness score enables us to collect many more positive triples than a random sampling strategy does. This strongly demonstrates the effectiveness of the positiveness score for the dataset construction problem. In particular, AMDC collected 6.18 times, 17.0 times, and 2.28 times more positive triples than AMDC_rand in the Kinships, UMLS, and Nations datasets, respectively.

Second, our querying strategy achieved the highest improvement on the UMLS dataset, the most imbalanced dataset. This shows that the AMDC method can successfully learn a model even from an imbalanced dataset.

Third, AMDC_pos_only had worse performance in the Kinships and UMLS datasets, but comparable performance with the other methods in the Nations dataset. This result can be explained by the property of the ROC-AUC loss function of negative triples. The contribution of the ROC-AUC loss function depends on the ratio between the number of unlabeled triples and positive triples. If the number of unlabeled triples is quite larger than that of positive triples, the ROC-AUC loss function has much influence on unlabeled triples. However, if the number of unlabeled triples decreases to be comparable to that of positive triples, it has less influence on unlabeled triples, and therefore, the effectiveness of the ROC-AUC loss function decreases. This can explain the fact that the increasing rate of the completion rate of AMDC_pos_only is comparable to that of other methods in the Kinships and UMLS datasets when the number of queries increases.

Fourth, the positiveness score is not beneficial to improve the predictive performance. AMDC achieved almost the same ROC-AUC scores as AMDC_rand on the UMLS and Nations datasets and achieved even worse ROC-AUC scores on the Kinships dataset. This supports our motivation to set two different problem settings.

4.5.2 Predictive Model Construction Problem

We then provide experimental results for the predictive model construction problem (Problem 2). We run the four methods using the uncertainty score to collect informative triples in order to improve the predictive performance. Figure 4 shows experimental results.

We present four conclusions drawn from these results. First, at every iteration, AMDC's ROC-AUC score is either the highest, or comparable to the highest, while the partial AMDCs are not always comparable to the highest one. In specific, AMDC performs the best on the Kinships and UMLS datasets, and is comparable to AMDC_rand on the Nations dataset. This demonstrates the benefits of the full AMDC method.

Second, AMDC_no_const resulted in the poorest score, and the standard deviations were the largest. This validates that the constraints we introduced helped to increase the stability of learning. By reducing the model complexity, we can successfully avoid overfitting.

Third, AMDC tends to outperform AMDC_rand in the middle of the active learning procedure. This illustrates the following active learning consequence. At the beginning, the predictive model shows insufficient active learning ability due to lack of labels. As the number of labels grows, the model is able to choose informative triples, and active learning becomes more effective. Towards the later iterations, the querying strategy benefits little when diminishing returns is observed in learning predictive models especially.

Fourth, the uncertainty score did not help improve the completion rate in the same way as the first experiment. This result, combined with the fourth finding in the previous section, shows that the dataset construction problem and the predictive model construction problem are essentially different problem areas. The predictive model construction problem needs both positive and negative labels, while the dataset construction problem aims to collect as many positive labels as possible.

5. RELATED WORK

The AMDC method is related to multi-relational data construction, multi-relational learning, and active learning in machine learning. We review existing studies and state the relationships with the AMDC method.

5.1 Multi-relational Data Construction

There are two main research directions in constructing a multi-relational dataset: manual construction and automatic construction by extracting data from documents.

ConceptNet [19], Cyc [18], and Wordnet [22] are constructed mainly by hand annotation. Von Ahn *et al.* [30] propose a game-with-a-purpose (GWAP) approach to use human resources efficiently. Unlike the AMDC method, these approaches do not make use of machine learning algorithms to facilitate manual dataset construction.

In bioinformatics, automatic extraction from scientific papers has been extensively studied. For example, Blaschke *et al.* [3] propose an automatic extraction method of protein-protein interactions from scientific abstracts. In the Semantic Web, knowledge bases are typically constructed from Web documents such as Wikipedia. DBpedia [17] and YAGO2 [12] have been built partly by extracting from Wikipedia. Dong *et al.* [10] combine automatic extraction methods and a machine learning method to construct a reliable knowledge base. The main difference between their method and ours is the objective of using machine learning. Dong *et al.* [10] utilize machine learning to improve the data reliability, while we apply machine learning to efficiently extract human knowledge that does not appear in documents.

5.1.1 Multi-relational Data Learning

Latent representation methods for multi-relational data learning have been broadly investigated because of their high predictive performance [7, 24, 25, 23, 5, 6, 31]. These approaches model entities as latent feature vectors and relations as operators on the vectors. In the RESCAL models [24, 25, 23], relations are modeled as asymmetric matrices that can be interpreted as operators on a latent feature vector of a tail of a triple. The models are learned by minimizing the squared loss function [24, 25] or the logistic loss function [23]. In the embedding models [7, 5, 6, 31], relations

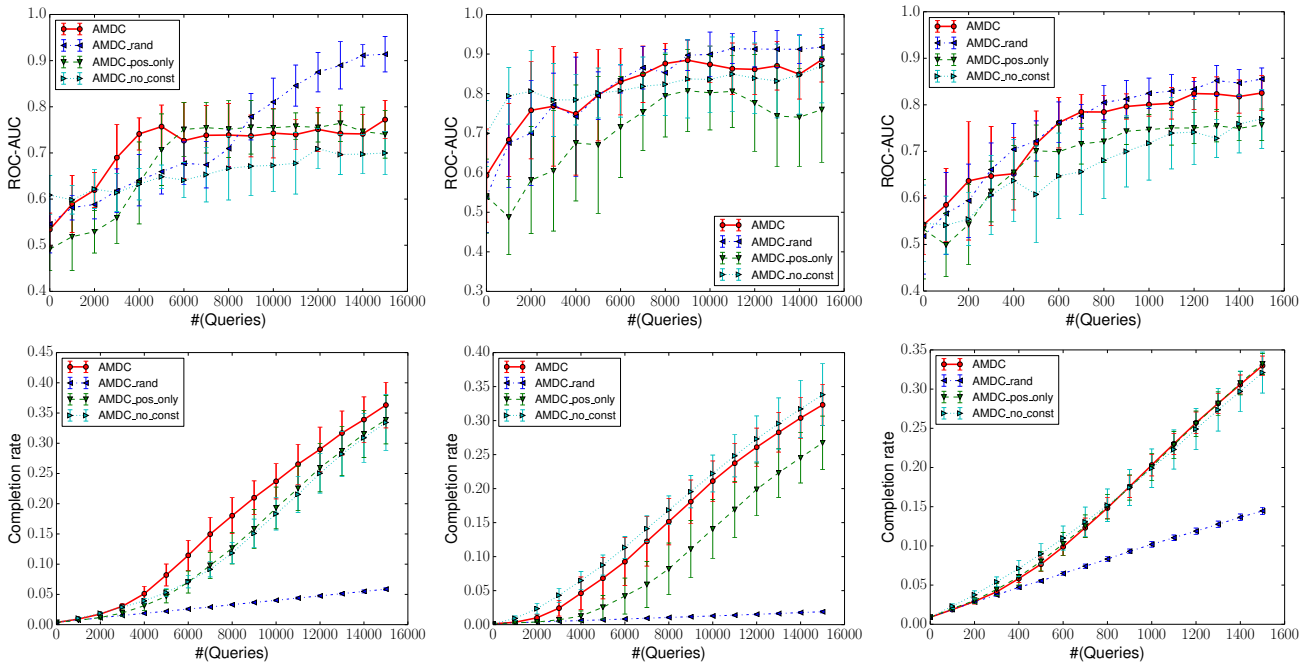

Figure 3: ROC-AUC scores and completion rates in the dataset construction problem (Problem 1). Upper-left, upper-middle, and upper-right figures correspond to the ROC-AUC scores on the Kinships, UMLS, and Nations datasets. Lower-left, lower-middle, and lower-right figures correspond to the completion rates on the Kinships, UMLS, and Nations datasets.

are modeled as matrices [7] or vectors [5, 6, 31]. The models are learned by minimizing the ROC-AUC loss function on positive and non-positive triples. We employ the RESCAL model in AMDC rather than the embedding models based on the initial experimental results.

The model complexity of the RESCAL models is high as compared to that of the embedding models. The RESCAL models assign $O(D^2)$ parameters to one relation while the embedding models [6, 31] assign $O(D)$ to one relation.

Our multi-relational data model has three crucial differences from the aforementioned existing models. First, by introducing additional constraints to the RESCAL model, we successfully develop a RESCAL-variant model with low model complexity. By fixing the length of latent feature vectors to 1 and restricting the set of relation matrices to the set of rotation matrices, we achieve the same model complexity with the embedding model [6]. Second, in order to make full use of negative triples, we combine the existing ROC-AUC loss function on positive and non-positive triples with another ROC-AUC loss function on negative and non-negative triples. Third, we further add the classification error loss function to the loss function in order to learn a decision boundary. The decision boundary plays a crucial role in active learning.

5.2 Active Learning

Active learning is a supervised learning scheme in which a learner is allowed to choose instances to be labeled. General techniques are summarized in the survey by Settles [28].

To the best of our knowledge, our approach is the first attempt to apply active learning to a multi-relational data model trained using the ROC-AUC loss function. Several research groups propose active learning algorithms for matrix factorization models [14, 15, 8, 29]. However, none of them employs the ROC-AUC loss function for learning, because these studies do not focus on imbalanced data. In order to achieve high predictive performance on an imbalanced multi-relational dataset, we develop an active learning algorithm for a model trained with the ROC-AUC loss function. Donmez *et al.* [11] propose an active learning algorithm on the learning-to-rank problem, in which a ranking model is learned using the ROC-AUC loss function. There are two main differences between Donmez *et al.*'s work and our work. The first difference is the model to be learned. Donmez *et al.* [11] employ RankSVM, while we develop a new multi-relational data model. The second difference is the method to estimate a decision boundary. Donmez *et al.* [11] assume that the decision boundary is between two scores whose difference is maximized. Our estimation method is believed to be more effective because it calibrates the predictive scores to satisfy the decision boundary 0 by incorporating the classification error loss function.

6. CONCLUSION

Constructing a multi-relational dataset can often benefit from human expert input, which can take the form of annotating triples whose positive/negative status is initially missing. While very valuable, human expert input is a limited resource, given that manually annotating a large amount of triples is a tedious and costly process. This research has focused on efficiently using human expert resources to construct a multi-relational dataset. We investigated two scenarios; a dataset construction problem and a predictive model construction problem.

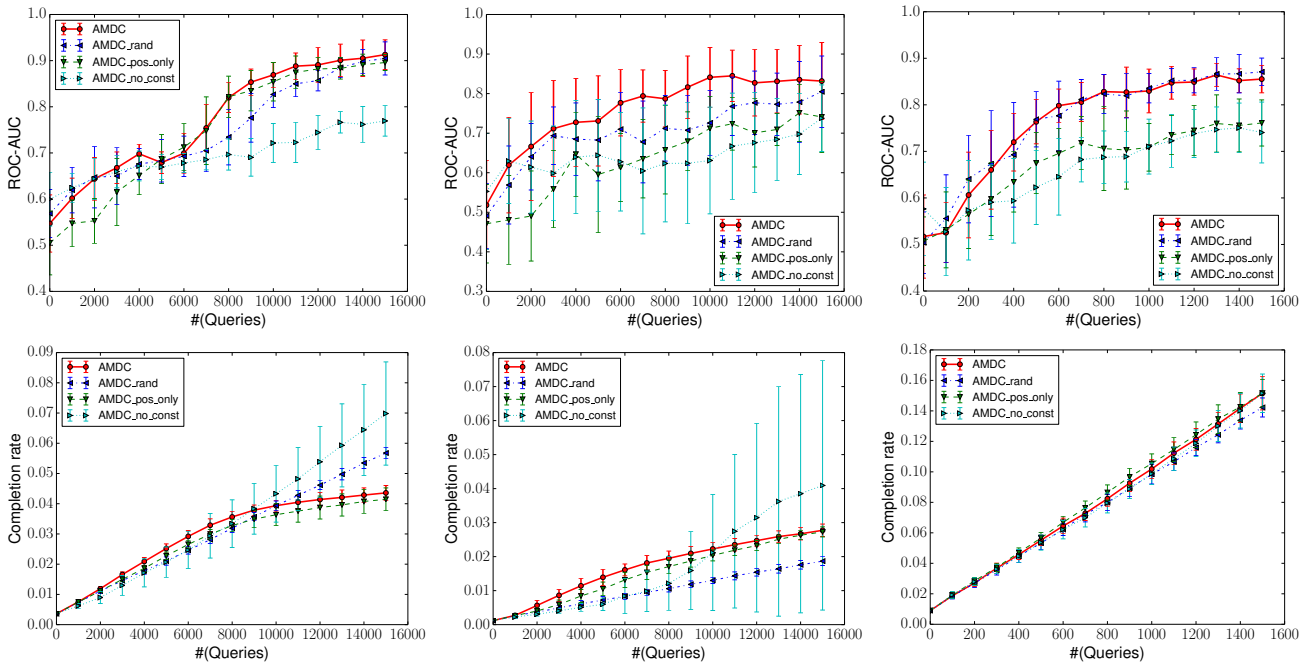

Figure 4: ROC-AUC scores and completion rates in the predictive model construction problem (Problem 2). Upper-left, upper-middle, and upper-right figures correspond to the ROC-AUC scores on the Kinships, UMLS, and Nations datasets. Lower-left, lower-middle, and lower-right figures correspond to the completion rates on the Kinships, UMLS, and Nations datasets.

We have introduced the AMDC method, our solution to this problem. AMDC performs active learning. It learns a model of a multi-relational dataset, and actively poses queries to an oracle, requesting labels to selected unlabeled triples. Our main technical contributions are threefold. First, we developed active learning algorithms using the ROC-AUC loss function. Active learning cannot directly be applied to a model learned on the ROC-AUC loss function, because such a model does not have a decision boundary, which is necessary to select uncertain triples for querying. We resolved this issue by using a classification error loss function, which enabled the model to learn the decision boundary. Second, we presented a new ROC-AUC loss function utilizing negative triples to learn efficiently from abundant negative triples. Existing learning methods employ the ROC-AUC loss function only on positive and non-positive triples, which do not distinguish negative triples from unlabeled ones. We combined a ROC-AUC loss function on negative and non-negative triples to make full use of negative triples. Third, we added two constraints to the RESCAL model to reduce the model complexity. We fixed the length of latent feature vectors and restricted the set of model parameters associated with relations.

We conducted experiments to validate the effectiveness of AMDC. We evaluated the impact of each of the three technical contributions, comparing the full AMDC method with three other variants where each of these features were turned off. This allowed to identify strengths and limitations of each individual contribution. We found that the full AMDC has consistently good performance at every iteration, showing that the technical contributions implemented in AMDC work well in conjunction with each other.

An interesting direction for future work involves using crowdsourcing for annotation purposes. Crowdsourcing provides easy access to abundant human resources at low cost. However, it has been pointed out that labels obtained with crowdsourcing are sometimes of low quality. We believe that the quality issue can be resolved by combining quality control techniques [32].

Another research direction is to incorporate automatic extraction methods. By using a large dataset extracted from documents as the initial dataset, AMDC will be able to select more appropriate triples that contribute to efficient data construction. The key challenge is to estimate the reliability of each triple. As the reliability of an automatically extracted dataset is not always guaranteed, it is necessary to estimate it with help of human annotation.

7. REFERENCES

[1] E. Antezana, W. Blondé, M. Egaña, A. Rutherford, R. Stevens, B. De Baets, V. Mironov, and M. Kuiper. BioGateway: a semantic systems biology tool for the life sciences. *BMC Bioinformatics*, 10, 2009.

[2] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette. Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *Journal of Biomedical Informatics*, 41(5):706–716, 2008.

[3] C. Blaschke, M. A. Andrade, C. Ouzounis, and A. Valencia. Automatic extraction of biological information from scientific text: protein-protein interactions. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 60–67, 1999.

[4] K. Bollacker, C. Evans, and P. Paritosh. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1249, 2008.

[5] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, 2013.

[6] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795, 2013.

[7] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, pages 301–306, 2009.

[8] S. Chakraborty, J. Zhou, V. Balasubramanian, S. Panchanathan, I. Davidson, and J. Ye. Active matrix completion. In *Proceedings of 2013 IEEE 13th International Conference on Data Mining*, pages 81–90, 2013.

[9] W. W. Denham. *The detection of patterns in Alyawarra nonverbal behavior*. PhD thesis, University of Washington, 1973.

[10] X. L. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. Knowledge Vault: a web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 601–610, 2014.

[11] P. Donmez and J. G. Carbonell. Active sampling for rank learning via optimizing the area under the ROC curve. In *Proceedings of the 31th European Conference on IR Research*, pages 78–89, 2009.

[12] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum. YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence*, 194:28–61, 2013.

[13] R. Jenatton, N. L. Roux, A. Bordes, and G. Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems 25*, pages 3167–3175, 2012.

[14] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme. Non-myopic active learning for recommender systems based on matrix factorization. In *Proceedings of 2011 IEEE International Conference on Information Reuse & Integration*, pages 299–303, 2011.

[15] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme. Exploiting the characteristics of matrix factorization for active learning in recommender systems. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, pages 317–320, 2012.

[16] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 381–388, 2006.

[17] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia – a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 2014.

[18] D. B. Lenat. CYC : a large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[19] H. Liu and P. Singh. ConceptNet – a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226, 2004.

[20] A. T. McCray. An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics*, 4(1):80–84, 2003.

[21] A. K. Menon and C. Elkan. Link prediction via matrix factorization. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2011*, pages 437–452, 2011.

[22] G. A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[23] M. Nickel and V. Tresp. Logistic tensor factorization for multi-relational data. In *Proceedings of ICML 2013 Workshop - Structured Learning: Inferring Graphs from Structured and Unstructured Inputs*, 2013.

[24] M. Nickel, V. Tresp, and H.-P. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning*, pages 809–816, 2011.

[25] M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*, pages 271–280, 2012.

[26] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461, 2009.

[27] R. J. Rummel. Dimensionality of nations project: attributes of nations and behavior of nation dyads, 1950-1965.

[28] B. Settles. Active learning literature survey. Technical report, University of Wisconsin–Madison, 2009.

[29] D. J. Sutherland, B. Póczos, and J. Schneider. Active learning and search on low-rank matrices. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 212–220, 2013.

[30] L. von Ahn, M. Kedia, and M. Blum. Verbosity: a game for collecting common-sense facts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 75–78, 2006.

[31] Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1112–1119, 2014.

[32] J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose vote should count more: optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043, 2009.

Algorithm 1 LEARN$((\Delta_p, \Delta_n), H)$

Input: multi-relational dataset (Δ_p, Δ_n) and hyperparameters $H = \{D, \gamma, \gamma', C_e, C_n, T, \alpha_0\}$.
Output: model (A, R).
Initialize: randomly initialize A, R to satisfy Eqs. (1) and (2), and set $\bar{A} = A, \bar{R} = R$, $t = 1$, $\alpha = \alpha_0$.

repeat
  Sample $v \sim$ Bernoulli(0.5)
  if $v = 1$ **then**
    Sample $t_p = (i_p, j_p, k_p)$ from Δ_p and $\bar{t}_p = (\bar{i}_p, \bar{j}_p, \bar{k}_p)$ from $\Delta \backslash \Delta_p$.
    Compute indicators as

$$I_1 \leftarrow H\left(\gamma - \mathbf{a}_{i_p}^\top R_{k_p} \mathbf{a}_{j_p} + \mathbf{a}_{\bar{i}_p}^\top R_{\bar{k}_p} \mathbf{a}_{\bar{j}_p}\right),$$
$$I_2 \leftarrow H\left(\gamma' - \mathbf{a}_{i_p}^\top R_{k_p} \mathbf{a}_{j_p}\right).$$

    Update model parameters as

$$\mathbf{a}_{i_p} \leftarrow \mathbf{a}_{i_p} - \alpha\left[-(I_1 + I_2 C_e)\bar{R}_{k_p} \bar{\mathbf{a}}_{j_p}\right],$$
$$\mathbf{a}_{j_p} \leftarrow \mathbf{a}_{j_p} - \alpha\left[-(I_1 + I_2 C_e)\bar{R}_{k_p}^\top \bar{\mathbf{a}}_{i_p}\right],$$
$$R_{k_p} \leftarrow R_{k_p} - \alpha\left[-(I_1 + I_2 C_e)\bar{\mathbf{a}}_{i_p} \bar{\mathbf{a}}_{j_p}^\top\right],$$
$$\mathbf{a}_{\bar{i}_p} \leftarrow \mathbf{a}_{\bar{i}_p} - \alpha\left[I_1 \bar{R}_{\bar{k}_p} \bar{\mathbf{a}}_{\bar{j}_p}\right],$$
$$\mathbf{a}_{\bar{j}_p} \leftarrow \mathbf{a}_{\bar{j}_p} - \alpha\left[I_1 \bar{R}_{\bar{k}_p}^\top \bar{\mathbf{a}}_{\bar{i}_p}\right],$$
$$R_{\bar{k}_p} \leftarrow R_{\bar{k}_p} - \alpha\left[I_1 \bar{\mathbf{a}}_{\bar{i}_p} \bar{\mathbf{a}}_{\bar{j}_p}^\top\right].$$

  else if $v = 0$ **then**
    Sample $t_n = (i_n, j_n, k_n)$ from Δ_n and $\bar{t}_n = (\bar{i}_n, \bar{j}_n, \bar{k}_n)$ from $\Delta \backslash \Delta_n$.
    Compute indicators as

$$I_3 \leftarrow H\left(\gamma - \mathbf{a}_{\bar{i}_n}^\top R_{\bar{k}_n} \mathbf{a}_{\bar{j}_n} + \mathbf{a}_{i_n}^\top R_{k_n} \mathbf{a}_{j_n}\right),$$
$$I_4 \leftarrow H\left(\gamma' + \mathbf{a}_{i_n}^\top R_{k_n} \mathbf{a}_{j_n}\right).$$

    Update model parameters as

$$\mathbf{a}_{i_n} \leftarrow \mathbf{a}_{i_n} - \alpha\left[(I_3 C_n + I_4 C_e)\bar{R}_{k_n} \bar{\mathbf{a}}_{j_n}\right],$$
$$\mathbf{a}_{j_n} \leftarrow \mathbf{a}_{j_n} - \alpha\left[(I_3 C_n + I_4 C_e)\bar{R}_{k_n}^\top \bar{\mathbf{a}}_{i_n}\right],$$
$$R_{k_n} \leftarrow R_{k_n} - \alpha\left[(I_3 C_n + I_4 C_e)\bar{\mathbf{a}}_{i_n} \bar{\mathbf{a}}_{j_n}^\top\right],$$
$$\mathbf{a}_{\bar{i}_n} \leftarrow \mathbf{a}_{\bar{i}_n} - \alpha\left[-I_3 C_n \bar{R}_{\bar{k}_n} \bar{\mathbf{a}}_{\bar{j}_n}\right],$$
$$\mathbf{a}_{\bar{j}_n} \leftarrow \mathbf{a}_{\bar{j}_n} - \alpha\left[-I_3 C_n \bar{R}_{\bar{k}_n}^\top \bar{\mathbf{a}}_{\bar{i}_n}\right],$$
$$R_{\bar{k}_n} \leftarrow R_{\bar{k}_n} - \alpha\left[-I_3 C_n \bar{\mathbf{a}}_{\bar{i}_n} \bar{\mathbf{a}}_{\bar{j}_n}^\top\right].$$

  end if
  for $i \in \mathcal{E}$ such that $\mathbf{a}_i \neq \bar{\mathbf{a}}_i$ **do**
    $\mathbf{a}_i \leftarrow \mathbf{a}_i / \|\mathbf{a}_i\|_2$.
    $\bar{\mathbf{a}}_i \leftarrow \mathbf{a}_i$.
  end for
  for $k \in \mathcal{R}$ such that $R_k \neq \bar{R}_k$ **do**
    $U_k, \Sigma_k, V_k \leftarrow \mathrm{SVD}(R_k)$.
    $R_k \leftarrow U_k V_k^*$.
    $\bar{R}_k \leftarrow R_k$.
  end for
  $\alpha \leftarrow \alpha_0 / \sqrt{t+1}$.
  $t \leftarrow t + 1$.
until $t \geq T$ holds.
return (A, R)

Algorithm 2 LEARNDEFAULT$((\Delta_p, \Delta_n), (\Delta_p^{(v)}, \Delta_n^{(v)}), \mathcal{H})$

Input: training dataset (Δ_p, Δ_n), validation dataset $(\Delta_p^{(v)}, \Delta_n^{(v)})$, and a set of models $\mathcal{H} = \{H^{(l)} \mid l \in \{1, \ldots, L\}\}$.
Output: default model (A, R).

for $l = 1, \ldots, L$ **do**
  $(A^{(l)}, R^{(l)}) \leftarrow$ LEARN$(\Delta_p, \Delta_n, H^{(l)})$.
  $v^{(l)} \leftarrow$ VALSCORE$(A^{(l)}, R^{(l)}, \Delta_p^{(v)}, \Delta_n^{(v)})$.
end for
$l^\star \leftarrow \arg\max_{l=1,\ldots,L} v^{(l)}$.
return $(A^{(l^\star)}, R^{(l^\star)})$.

Algorithm 3 Active Multi-relational Data Construction

Input: budget $B > 0$, budget for validation $N_{\mathrm{val}} > 0$, initial dataset $(\Delta_p^{(0)}, \Delta_n^{(0)})$, a set of hyperparameters $\mathcal{H} = \{H^{(l)} \mid l \in \{1, \ldots, L\}\}$, the number of queries q, the number of unlabeled triples to calculate a query score Q, and a query score function.
Output: dataset (Δ_p, Δ_n) and model $(A^\star, R^\star)$.
Initialization:

$(\Delta_p, \Delta_n) \leftarrow (\Delta_p^{(0)}, \Delta_n^{(0)})$.
Create a validation dataset of size N_{val}, $(\Delta_p^{(v)}, \Delta_n^{(v)})$.
$(A^\star, R^\star) \leftarrow$ LEARNDEFAULT$((\Delta_p, \Delta_n), (\Delta_p^{(v)}, \Delta_n^{(v)}), \mathcal{H})$.
$t \leftarrow 0$.

Main algorithm:

while $tq \leq B$ **do**
  Randomly sample Q unlabeled triples to construct Δ_u.
  for each triple $t \in \Delta_u$ **do**
    Compute the query score q_t using the query score function.
  end for
  Choose q triples with the lowest query scores to construct $\Delta_q^\star$.
  for each triple $t \in \Delta_q^\star$ **do**
    $l \leftarrow \mathcal{O}(t)$.
    if $l = 1$ **then**
      $\Delta_p \leftarrow \Delta_p \cup \{t\}$.
    else
      $\Delta_n \leftarrow \Delta_n \cup \{t\}$.
    end if
  end for
  $(A^\star, R^\star) \leftarrow$ LEARNDEFAULT$((\Delta_p, \Delta_n), (\Delta_p^{(v)}, \Delta_n^{(v)}), \mathcal{H})$.
  $t \leftarrow t + 1$.
end while
return (Δ_p, Δ_n) and $(A^\star, R^\star)$.