Active Ensemble Learning for Knowledge Graph Error Detection

Junnan Dong
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
hanson.dong@connect.polyu.hk

Qinggang Zhang
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
qinggangg.zhang@connect.polyu.hk

Xiao Huang
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
xiaohuang@comp.polyu.edu.hk

Qiaoyu Tan
Texas A&M University
College Station, TX, USA
qytan@tamu.edu

Daochen Zha
Rice University
Houston, TX, USA
daochen.zha@rice.edu

Zihao Zhao
The Hong Kong Polytechnic University
Hung Hom, Hong Kong SAR
zi-hao.zhao@connect.polyu.hk

ABSTRACT

Knowledge graphs (KGs) could effectively integrate a large number of real-world assertions, and improve the performance of various applications, such as recommendation and search. KG error detection has been intensively studied since real-world KGs inevitably contain erroneous triples. While existing studies focus on developing a novel algorithm dedicated to one or a few data characteristics, we explore advancing KG error detection by assembling a set of state-of-the-art (SOTA) KG error detectors. However, it is nontrivial to develop a practical ensemble learning framework for KG error detection. Existing ensemble learning models heavily rely on labels, while it is expensive to acquire labeled errors in KGs. Also, KG error detection itself is challenging since triples contain rich semantic information and might be false because of various reasons. To this end, we propose to leverage active learning to minimize human efforts. Our proposed framework - KAEL, could effectively assemble a set of off-the-shelf error detection algorithms, by actively using a limited number of manual annotations. It adaptively updates the ensemble learning policy in each iteration based on active queries, i.e., the answers from experts. After all annotation budget is used, KAEL utilizes the trained policy to identify remaining suspicious triples. Experiments on real-world KGs demonstrate that we can achieve significant improvement when applying KAEL to assemble SOTA error detectors. KAEL also outperforms SOTA ensemble learning baselines significantly.

CCS CONCEPTS

• **Information systems** → **Data cleaning**; • **Computing methodologies** → **Ensemble methods**; **Active learning settings**.

KEYWORDS

knowledge graphs, ensemble learning, knowledge graph refinement

WSDM '23, February 27-March 3, 2023, Singapore, Singapore

ACM ISBN 978-1-4503-9407-9/23/02…$15.00
https://doi.org/10.1145/3539597.3570368

ACM Reference Format:
Junnan Dong, Qinggang Zhang, Xiao Huang, Qiaoyu Tan, Daochen Zha, and Zihao Zhao. 2023. Active Ensemble Learning for Knowledge Graph Error Detection. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27-March 3, 2023,Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570368

1 INTRODUCTION

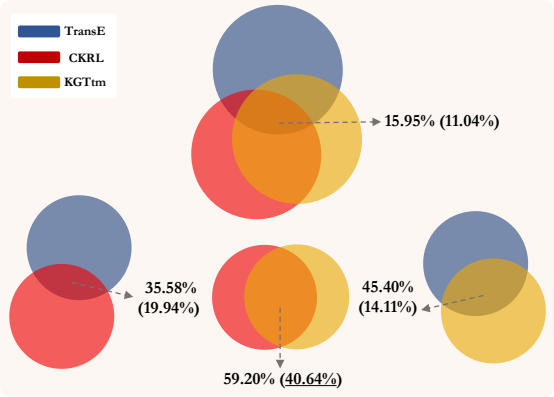

Figure 1: Overlaps(%) among three detectors on UMLS.

Error detection is essential to knowledge graphs (KGs), since real-world KGs inevitably contain considerable false triples [1, 2]. KGs are an efficient data structure that could integrate numerous real-world assertions as a network [3]. By embedding KGs [4], we could incorporate them into various intelligent systems and enhance the prediction performance, such as recommender systems and conversational agents [5, 6]. Real-world KGs are too large to be manually constructed. Instead, heuristic algorithms were employed to extract triples from unstructured information sources, such as crowd-sourcing websites. During this process, considerable errors were inevitably introduced. For example, (*Fred_Goodwin*, *ceoof*, *Scotland*) is an erroneous triple extracted from the sentence "*Former Royal Bank of Scotland boss Fred Goodwin has had his knighthood removed*" [7]. To rectify these errors, intensive KG error detection algorithms have been proposed [8, 9]. Early studies mainly focus on defining golden rules that correct triples should not violate [10, 11].

Recent efforts have been devoted to embedding KGs and learning regulations that correct triples tend to match [12, 13]. A few studies explore to train suitable classifiers based on randomly generated negative samples [14]. However, their empirical performance still can not meet the demands in practice.

While existing knowledge graph error detectors focus on exploiting a novel algorithm to model one or a few types of errors, according to particular data characteristics [1, 9], we explore applying ensemble learning to integrate state-of-the-art (SOTA) KG error detectors. It is motivated by the successful applications of ensemble learning in several learning tasks, such as classification [15], regression [16] and anomaly detection [17]. Figure 1 illustrates an interesting observation demonstrating that ensemble learning could potentially advance KG error detection. We apply three detectors, i.e., TransE [12], CKRL [18], and KGTtm [1] on a medical KG named UMLS [19] to detect 326 errors. The percentages of overlap between the suspicious triples identified by different models are shown in Figure 1, where only a portion of suspicious triples are real errors. The percentage of real anomalies to 326 is shown in '( )'. TransE is based on KG embedding, and its overlap with CKRL only contain 19.94% real errors among 326, while there are 40.64% errors in the overlap between CKRL and KGTtm since they are both path-based. In total, TransE, CKRL, and KGTtm find 92, 169, and 174 real errors, respectively, while there are only 36 overlaps among them. Thus, different detectors focus on different groups of errors. This motivates us to assemble different off-the-shelf algorithms to combine their strengths for effective error detection.

However, it is challenging to develop an effective ensemble learning framework for knowledge graph error detection. (i) KG error detection itself is difficult, given that triples have rich semantic meanings. A deviant triple could be caused by various factors [20]. For example, we could have erroneous relations, e,g., (*Will_Smith*, *actor_of*, *The_Karate_ Kid*), or incorrect tail/head entities. (ii) Most off-the-shelf ensemble learning models are supervised [21], while it is expensive to obtain labeled errors in KGs. Labels are essential for the evaluation and integration in ensemble learning [22]. Real-world KGs often contain millions of triples [23]. It is impossible to manually check the entire KG. Thus, we propose to perform active ensemble learning on KGs. Given a specific triple, an expert could determine its correctness on the grounds of their understanding and references. A small number of active annotations might facilitate error detection, but also bring additional challenges to ensemble learning. (iii) It requires the ensemble learning model to conduct assembling actively and iteratively. The model needs to balance the exploration and exploitation. (iv) The search space of active queries is large in KGs. In each active learning iteration, only one triple is selected for experts to annotate [24]. The annotation budget is considerably less than potential false triples, while the latter is further significantly less than correct triples.

To this end, we investigate the active ensemble learning for knowledge graph error detection, aiming to minimize human efforts. We propose an effective framework - Knowledge graph Active Ensemble Learning (KAEL). In particular, we aim to answer two research questions as follows. (i) Given a set of SOTA KG error detection algorithms, how to perform effective ensemble learning and integrate them to detect more erroneous triples? (ii) How to utilize the opportunity of actively querying experts/oracle to optimize the ensemble learning policy? We summarize our major contributions.

Notation	Description
(h, r, t)	a triple, i.e., (head, relation, tail)
$(\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t) \in \mathbb{R}^{3d}$	embedding representation of (h, r, t)
$b \in [1, 2, ..., m]$	index of base KG error detector
$\mathbf{x} \in \mathbb{R}^{3d}$	embedding of a triple
$f_b(\mathbf{x})$	suspicious score of a triple in b
$\mathbf{X}^{(b)(k)} \in \mathbb{R}^{u \times 3d}$	embedding matrix of detector b in k steps
$r^{(i,b)}$	a reward for triple i from base detector b
$\mathbf{r}^{(b)(k)} \in \mathbb{R}^{u}$	reward vector for detector b in k steps
Q	query budget
p	triple expectation of being false
$\mathbf{V}$	detected triple embedding matrix

Table 1: Major notations and definitions.

- We formally define the problem of active ensemble learning for knowledge graph error detection.
- We propose an effective framework - KAEL. It takes multiple sets of top suspicious triples returned by multiple SOTA detectors as input, and integrates them by using a novel and active ensemble learning model.
- We design a tailored active learning algorithm to adaptively update the ensemble learning policy of KAEL in each iteration, based on the feedback from the oracle. It uses a dedicated multi-armed bandit algorithm to take full advantage of the limited active annotations to optimize the ensemble learning.
- We conduct experiments on real-world KGs and empirically demonstrate that KAEL outperforms SOTA KG error detectors and SOTA ensemble learning models. KAEL achieves better performance when more or better base detectors are integrated.

2 PROBLEM STATEMENT

Notations: We denote scalars as lowercase alphabets (e.g., x), vectors as boldface lowercase alphabets (e.g., $\mathbf{x}$) and matrices as boldface uppercase alphabets (e.g., $\mathbf{V}$). We list main notations in Table 1. Let $\mathcal{G}$ be a knowledge graph with n triples. Each triple (h, r, t) consists of a head entity h, a relation r, and a tail entity t. We involve human annotations as oracle and formally define the problem of active ensemble learning for KG error detection as follows.

> Given a KG $\mathcal{G}$ and m off-the-shelf error detection algorithms, we aim to apply ensemble learning to identify top K suspicious deviant triples. Meanwhile, it is allowed to query the oracle about the correctness of any one triple for Q times ($Q < K$). The performance is evaluated based on the number of real errors in the K suspicious triples identified by the proposed framework.

3 ACTIVE ENSEMBLE LEARNING - KAEL

This section elaborates on the KAEL framework, which leverages active learning for an ensemble learning based KG error detection. Figure 2 illustrates the core idea with a toy example. There are totally ten triples in the inputted KG, labeled from 1 to 10. We aim to detect five potential anomalies within query budget $Q = 3$. Starting from sub-figure (a), i.e., the initialization stage, we assemble

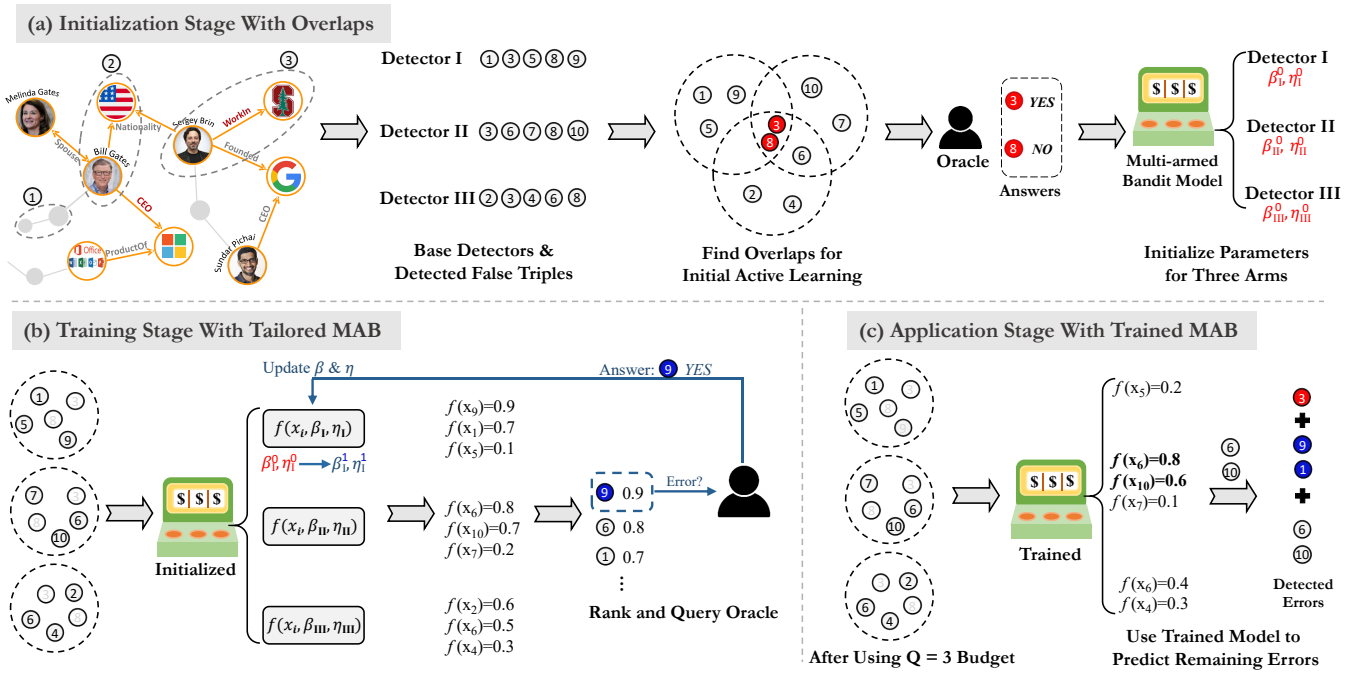

Figure 2: By assembling detectors b =[I, II, III], we reduce the search space from entire n triples to three sets of suspicious ones. A tailored multi-armed bandit (MAB) model is then designed to perform active ensemble learning. We adaptively update it based on the answers from oracle in each iteration.

three detectors b ={I, II, III} with three sets of top five suspicious triples they detected respectively, i.e., $\{1, 3, 5, 8, 9\}$, $\{3, 6, 7, 8, 10\}$, and $\{2, 3, 4, 6, 8\}$. We use two of the budget to query the oracle with all overlaps among these candidates, i.e., $\{3, 8\}$, and initialize the multi-armed bandit (MAB) with answers. In sub-figure (b), we further train the policy with the remaining lists, i.e., $\{1, 5, 9\}$, $\{6, 7, 10\}$, and $\{2, 4, 6\}$, and expend the remaining one budget on the highest suspicious triple $\{9\}$ detected by MAB. While in sub-figure (c) with no oracle involved, we directly apply a trained MAB on the rest of three sets of triples and output two triples with the highest scores, i.e., $\{6, 10\}$ to supplement the error list. We first introduce how we conduct ensemble learning with SOTA base detectors in a unified and concentrated space (**Section 3.1**). We then present an active ensemble policy learning algorithm based on the tailored MAB (**Section 3.2**). Finally, we propose a three-stage active ensemble scheme by combining absolute majority voting and MAB to maximally exploit the human feedback for ensemble policy learning (**Section 3.3**).

3.1 Ensemble Learning for KG Error Detection

The core idea is to integrate multiple SOTA unsupervised detectors to obtain multiple initially ranked lists of top K suspicious triples. We consider these prioritized triples as valuable candidates for errors. Through concentrating on this significantly shrunk training space, we boost the ensemble learning on KGs. However, integrating various base detectors is nontrivial due to two reasons. First, different detectors extract features in various means. For example, [12] learns the embedding in a real space while [25] represents triples on the complex space with both real and imaginary parts. Recent efforts trend to dedicate more on non-euclidean spaces, such as a hyperbolic space [26]. Second, the compiling environments of detectors are also different, i.e., [18] experimentalizes based on C++. To tackle these challenges, we resort to representing the returned triples uniformly in the same embedding space and compiling environment before the ensemble policy training. Specifically, we adopt the popular translation-based embedding model TransE [12] to encode triples as follows.

$$\mathbf{e}_h + \mathbf{e}_r \approx \mathbf{e}_t, \tag{1}$$

where $\mathbf{e}_h \in \mathbb{R}^d$ (or $\mathbf{e}_t \in \mathbb{R}^d$) denotes the head (or tail) entity embedding, and $\mathbf{e}_r \in \mathbb{R}^d$ is the embedding of relation. After TransE is well-trained, we generate the triple representation for triple (h, r, t) by concatenating its three embedding vectors, i.e., $\mathbf{x} \in \mathbb{R}^{3d} = [\mathbf{e}_h, \mathbf{e}_r, \mathbf{e}_t]$.

These triple representations will be used to perform active ensemble learning in the following section. Following this principle, we can not only unify the outputs of different types of base detectors, but also pay our best attention to those informative suspicious triples.

3.2 Active Ensemble Policy Learning with MAB

To facilitate the training of ensembles, we design an active query strategy to label the most expected candidate in each iteration. Given that the query budget is often limited in practice as human annotation is laborious and expensive, the proposed active ensemble learning framework suffers from a dilemma during the learning

process. On one hand, we want to make full use of prior knowledge by greedily selecting the most suspicious triple in each iteration, i.e., *exploitation*. On the other hand, we also want to collect feedback from the candidate sets that are under-explored at the moment, i.e., *exploration*. To tackle this challenge, we develop a novel active ensemble learning algorithm based on a tailored multi-armed bandit to trade-off for valuable selections.

Specifically, we formulate our active ensemble learning objective as a reward maximization problem. Let $r^{(i,k)} \in \{0, 1\}$ denotes the reward determined by the feedback from the oracle, where i is a triple from $m \times K$. In the current iteration k, if the selected query, i.e., triple i, is a true error, we will get a reward, denoted as $r^{(i,k)} = 1$, otherwise, 0. Our learning objective is to maximize the cumulative rewards in K iterations as follows:

$$\max \sum_{k=1}^{K} r^{(i,k)}. \tag{2}$$

The query function and error detection function is coherently combined with an expectation function $E(\cdot)$ below to capture the linear correlation between triple embedding $\mathbf{x}_i^{(b)}$ and $r^{(i,b)}$, measuring the suspicion of a triple i from detector b being anomalous and indicating the expectation of obtaining a reward $r^{(i,b)}$.

$$E(r^{(i,b)}|\mathbf{x}_i^{(b)}) = \mathbf{x}_i^{(b)} \times \boldsymbol{\beta}^{(b)} + \eta^{(b)}, \tag{3}$$

where $\boldsymbol{\beta}^{(b)}$ is a non-negative parameter vector for b learned in former $k-1$ steps and $\eta^{(b)}$ is a trade-off noise satisfying the Gaussian distribution $\mathcal{N}(0, (\sigma^{(b)})^2)$. Since maximizing the $\mathbf{x}^{(b)(k)} \times \boldsymbol{\beta}^{(b)(k)}$ term can encourage exploitation [27], we could effectively update $\boldsymbol{\beta}^b$ by modeling the correlations among the anchor triple $\mathbf{x}_i^{(b)} \in \mathbb{R}^{3d}$ and all selected triples $\mathbf{X}^{(b)(k)} \in \mathbb{R}^{u \times 3d}$ for base detector b in k steps, as well as all the historically obtained rewards $\mathbf{r}^{(b)(k)} \in \mathbb{R}^u$ by choosing the current detector b. Notably, $u \leq k$ indicates the number of triples that the model has selected from the current detector b. To be specific, we update $\boldsymbol{\beta}^b$ as below.

$$\begin{aligned} J(\mathbf{X}^{(b)(k)}, \mathbf{r}^{(b)(k)}) &= \sum_{k=1}^{K} (\mathbf{r}^{(i,b)} - \mathbf{X}^{(b)(k)} \boldsymbol{\beta}^{(b)})^2 \\ \rightarrow \boldsymbol{\beta}^{(b)} &= ((\mathbf{X}^{(b)(k)})^\top \mathbf{X}^{(b)(k)})^{-1} (\mathbf{X}^{(b)(k)})^\top \mathbf{r}^{(b)(k)}, \end{aligned} \tag{4}$$

where J is the ordinary least square based loss function. While the current strategy works generally, it may suffer from low bias and high variance issues in practice. To avoid over-fitting and keep the reversibility of $\mathbf{X}^{(b)(k)}$, we introduce a penalty term based on L2 norm in a ridge regression as

$$\begin{aligned} J(\mathbf{X}^{(b)(k)}, \mathbf{r}^{(b)(k)}) = &\sum_{k=1}^{K} (\mathbf{r}^{(b)(k)} - \mathbf{X}^{(b)(k)} \boldsymbol{\beta}^{(b)})^2 \\ &+ \lambda^b \parallel \boldsymbol{\beta}^{(b)} \parallel_2^2 . \end{aligned} \tag{5}$$

λ is a hyperparameter, we solve the over-fitting problem by adopting a suitable λ. By differentiating the above formula and making the derivative of the cost function zero, we solve for the parameter as follows,

$$\begin{aligned} J(\boldsymbol{\beta}^{(b)}) = &(\mathbf{r}^{(b)(k)} - \mathbf{X}^{(b)(k)} (\boldsymbol{\beta}^{(b)})^\top (\mathbf{r}^{(b)(k)} - \mathbf{X}^{(b)(k)} \boldsymbol{\beta}^{(b)}) \\ &+ \lambda^b (\boldsymbol{\beta}^{(b)})^\top \boldsymbol{\beta}^{(b)}. \end{aligned} \tag{6}$$

By setting it to zero, we can update $\boldsymbol{\beta}^{(b)}$ for $k+1$ iteration as follows,

$$\boldsymbol{\beta}^{(b)} = \left((\mathbf{X}^{(b)(k)})^\top \mathbf{X}^{(b)(k)} + \lambda^b \mathbf{I} \right)^{-1} (\mathbf{X}^{(b)(k)})^\top \mathbf{r}^{(b)(k)}, \tag{7}$$

where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is an identity matrix.

Since exploitation is fulfilled by maximizing the expectation with a learned $\boldsymbol{\beta}^b$, to facilitate exploration on less explored detectors, we adopt an upper confidence bound for exploration-exploitation trade-off by introducing $\eta^{(b)}$. Inspired by prevailing studies [28], for any $\delta > 0$ with the probability at lease $(1 - \delta)$, the reward expectation $E(r^{(i,b)}|\mathbf{x}_i^{(b)})$ is bounded by a confidence interval:

$$\mathbf{x}_i^{(b)} \boldsymbol{\beta}^{(b)} - \gamma \times f_\gamma(\mathbf{X}^{(b)(k)}) \leq E \leq \mathbf{x}_i^{(b)} \boldsymbol{\beta}^{(b)} + \gamma \times f_\gamma(\mathbf{X}^{(b)(k)}), \tag{8}$$

where γ is a constant value, i.e., $\gamma = 1 + \sqrt{ln(2/\delta)/2}$. Also we can derive the correlation term $f_\gamma(\mathbf{X}^{(b)(k)})$ hereunder.

$$f_\gamma(\mathbf{X}^{(b)(k)}) \triangleq \sqrt{\mathbf{x}_i^{(b)} \left((\mathbf{X}^{(b)(k)})^\top \mathbf{X}^{(b)(k)} + \lambda^b \mathbf{I} \right)^{-1} (\mathbf{x}_i^{(b)})^\top}. \tag{9}$$

Hence, through learning from the anchor triple $\mathbf{x}_i^{(b)} \in \mathbb{R}^{3d}$ and all historically selected triples $\mathbf{X}^{(b)(k)} \in \mathbb{R}^{u \times 3d}$ for base detector b in k steps, we can simply derive the equation to appropriately update the trade-off noise term $\eta^{(b)}$ for the next iteration.

$$\begin{aligned} \eta^{(b)} = \\ \gamma \times \sqrt{\mathbf{x}_i^{(b)} \left((\mathbf{X}^{(b)(k)})^\top \mathbf{X}^{(b)(k)} + \lambda^b \mathbf{I} \right)^{-1} (\mathbf{x}_i^{(b)})^\top}. \end{aligned} \tag{10}$$

When a triple with larger $\mathbf{x}_i^{(b)} \boldsymbol{\beta}^{(b)}$ is selected, this reflects an exploitation process. Similarly, when the model chooses a triple with larger $\eta^{(b)}$, this variance shows an exploration process since the model performs few selections of the current detector. Thus, jointly maximizing $(\mathbf{x}_i^{(b)} \times \boldsymbol{\beta}^{(b)} + \eta^{(b)})$ helps us get rid of the dilemma.

In consequence, active ensemble learning for KG error detection is an effective human-in-the-loop approach, which can timely leverage the feedback from domain experts to optimize the ensemble policy. Guided by the objective of maximizing cumulative rewards s, we train the model to learn from limited annotations and balance the exploitation and exploration by a tailored MAB algorithm.

3.3 Three-stage Active Ensemble Scheme

In this section, we introduce a three-stage scheme designed for active ensemble KG error detection. By integrating majority voting and a tailored MAB, we fully utilize oracle's feedback with limited queries in each stage. The whole process is illustrated in Figure 2. Given a default Q budget to query the oracle for labels, we divide it into Q_1 and Q_2 for both initialization and training stages, respectively, where $Q = Q_1 + Q_2$.

Pre-training of base detectors. Empirically, both the performance and the quantity of base detectors lay decisive influences, we carefully choose base detectors from two aspects: (i) embedding-based methods: **TransE** [12] and **ComplEx** [25] that expresses a triple in the complex space and learns representation by dividing the entities

Algorithm 1: Training stage of tailored MAB - KAEL

Input: Candidate triples, Initialized $\boldsymbol{\beta}^{(b)} \in \mathbb{R}^{1\times d}$ and $\eta^{(b)}$;
Output: Learned $\boldsymbol{\beta}^{(b)}$ and $\eta^{(b)}$, Detected error list $\mathbf{V}$, Total rewards s;
Initialize an $\mathbf{X}$={ } and $\mathbf{r}$={ } to store selected triples and the rewards, and remaining budget to query ($Q2$);
for *i=1:*($Q2$) **do**
  for *j=1:m* **do**
    Sort triples with expectation p from current base detector with $\boldsymbol{\beta}^{(j)}$ and $\eta^{(j)}$ and pick the best;
  Rank all picked triples, pick the best of the best triple$^{(j)}$ and query for reward $r^{(j)}$;
  if *triple*$^{(j)}$ *is a real anomaly* **then**
    Obtain reward $r_i^{(j)}$ 1;
    Append triple$_i^{(j)}$ to $\mathbf{V}$, s plus 1;
  else
    Obtain reward $r_i^{(j)}$ 0;
  Append triple$^{(j)}$ and reward $r^{(j)}$ to $\mathbf{X}^{(\mathrm{i})(\mathrm{j})}$ and $\mathbf{r}^{(i)(j)}$.
  Update $\boldsymbol{\beta}^{(j)}$ and $\eta^{(j)}$ based on $\mathbf{X}^{(i)(j)}$ and $\mathbf{r}^{(i)(j)}$.
Return $\boldsymbol{\beta}^{(b)}$, $\eta^{(b)}$, $\mathbf{V}$ and s;

and relations into real parts; (ii) KG error detectors: **CKRL** [18], which singles out noises based on a confidence-aware framework leveraging both local and global information to jointly optimize the training process and **KGTtm** [1] that learns natural semantics of triples, together with the global structural information to measure trustworthiness of each individual triple.

3.3.1 Initialization Stage with Majority Voting. Intuitively, if all the base detectors vote with agreement on a suspicious triple, then this triple stands in most probability to be erroneous indeed. This is a common strategy termed *absolute majority vote* as a traditional ensemble learning strategy. In the initialization stage, we realize this intuition by prioritizing the overlapping triples among all sets of candidates.

Definition 3.1 (Overlapping triples). We refer overlapping triples as the duplicated triples among all m base detectors, $\mathbf{A} = [\mathbf{X}^{(1)} \cap \mathbf{X}^{(2)} \cap ... \cap \mathbf{X}^{(m)}]$.

The first Q_1 querying opportunity is spent on all overlapping triples from $\mathbf{A}$ to initialize the MAB policy with as many real error samples as possible. Let c_1 be the number of real anomalies among the overlaps, $c_1 \leq Q_1$. All overlaps are appended into $\mathbf{V} \in \mathbb{R}^{Q_1\times d}$. This absolute majority voting strategy could effectively alleviate the *cold-start* problem with valuable labels and improve the efficiency of the use of the limited budget.

3.3.2 Training Stage of Active Ensemble Policy. Given an initialized policy, the training stage will make full use of the remaining Q_2 budgets to incrementally update the selection and detection. The main training procedure is summarized in Algorithm 1. First, we input the remaining sets of candidate triple embedding and the initialized $\boldsymbol{\beta}^{(b)} \in \mathbb{R}^d$ and $\eta^{(b)}$. Then in k steps, we calculate triple expectation $E(r^{i,b}|\mathbf{x}_i^{(b)})$ iteratively, $k \leq Q_2$, walking through each

Algorithm 2: Application stage with trained MAB - KAEL

Input: Remaining Sets of candidate triples, learned $\boldsymbol{\beta}^{(b)} \in \mathbb{R}^{1\times d}$ and $\eta^{(b)}$;
Output: Detected error list $\mathbf{V}$;
Initialize remaining budget to query ($K - Q_1 - c_2$);
for *i=1:*($K - Q_1 - c_2$) **do**
  for *j=1:m* **do**
    Pick the triple with highest expectation p from current base detector with learned $\boldsymbol{\beta}^{(j)}$ and $\eta^{(j)}$;
  Rank all picked triples, choose the best of the best triple$^{(j)}$;
  Append triple$^{(j)}$ to $\mathbf{V}$;
Return $\mathbf{V}$;

detector with corresponding $\boldsymbol{\beta}^{(b)}$, $\eta^{(b)}$ and select top one triple with the highest p. After that, among all m selected triples in current step, we pick the best of the best to query the oracle for a final reward $r^{i,k}$ for the current iteration and append the queried triple$_i^b$ into $\mathbf{X}^{(b)k}$. Finally, we update $\boldsymbol{\beta}^{(b)}$ and $\eta^{(b)}$ to learn from the previous selections to differentiate particular error types in the best detector. Among all queried triples in this stage, let c_2 be the number of queries receiving $r = 1$. After spending $Q2$, we append this batch of c_2 real anomalies into the list $\mathbf{V} \in \mathbb{R}^{(Q_1+c_2)\times d}$, $c_2 \leq Q_2$.

3.3.3 Application Stage of the Trained Policy. In the final stage, after running out of the budget, we apply the trained policy on remaining triples to replenish the error list $\mathbf{V}$ until totally Top@K triples are detected. The overall procedure is summarized in Algorithm 2. Since there is no oracle involved in this phase, we do not further update the policy with newly detected triples. Alternatively, we directly compute p for all the remaining candidates from m detectors with previously learned $\boldsymbol{\beta}^{(b)}$ and $\eta^{(b)}$ and rank them for top $(K-Q_1-c_2)$ triples. This aims to find triples best fit the implicit patterns we excavated from historically selected triples $\mathbf{X}^{(b)(k)} \in \mathbb{R}^{u\times 3d}$ and obtained rewards $\mathbf{r}^{(b)(k)} \in \mathbb{R}^u$ in the current base detector b, where $u \leq (Q_1+c_2)$. Afterwards, we verify all triples that are screened out in this stage for performance evaluation, and let c_3 be the number of real anomalies, $c_3 \leq (K - Q_1 - c_2)$.

4 EXPERIMENTS

In this section, we evaluate KAEL upon three real-world KGs. Specifically, we aim to answer following research questions. **RQ1**: How effective is KAEL compared the state-of-the-art KG error detection methods? **RQ2**: Can KAEL outperform other ensemble learning and active ensemble learning based strategies? **RQ3**: How to set an appropriate active querying budget for KAEL over different datasets? **RQ4**: What are the impacts of different combinations of base detectors on KAEL? **RQ5**: How does KAEL make queries among different base detectors through active ensemble learning?

4.1 Experimental Settings

4.1.1 Datasets. We consider two benchmark real-world KGs. **WN-18RR**: it describes the characteristics of associations among English words with 46,943 entities, 11 relations and 93,003 triples. **FB15K-237**: it is a subset of Freebase [29], containing 14,541 entities, 237

Datasets		UMLS			WN18RR			FB15k-237		
Top@K		1%	3%	5%	1%	3%	5%	1%	3%	5%
SOTA Detectors	TransE	0.615	0.385	0.282	0.589	0.337	0.272	0.955	0.853	0.777
	ComplEx	0.646	0.595	0.571	0.428	0.281	0.241	0.677	0.557	0.454
	CKRL	0.800	0.605	0.518	0.683	0.464	0.353	0.895	0.679	0.534
	KGTtm	0.661	0.620	0.534	0.647	0.471	0.393	0.916	0.711	0.551
Ensemble Learning	RS	0.633	0.584	0.446	0.624	0.403	0.368	0.765	0.630	0.501
	RMV	0.791	0.663	0.580	0.701	0.512	0.426	0.923	0.742	0.544
Active Ensemble	OIS	0.775	0.649	0.562	0.688	0.497	0.395	0.894	0.701	0.526
	AAD-IFOR	0.477	0.520	0.454	0.329	0.246	0.182	0.560	0.511	0.405
	AAD-HST	0.615	0.515	0.426	0.299	0.224	0.176	0.583	0.497	0.423
	KAEL	**0.847**	**0.716**	**0.669**	**0.862**	**0.612**	**0.479**	**0.964**	**0.903**	**0.815**

Table 2: Error detection performance of baselines and KAEL on three real-world datasets with an anomaly ratio of 5%.

relations and 310,116 triples. For all the datasets, we synthesize anomalies by replacing the head or tail entity of a triple randomly, with an anomaly ratio of 5%. We summarize the dataset statistics of in Table 3. Additionally, to evaluate the generalization ability of KAEL, we also include a specific medical domain KG. **UMLS**: Unified Medical Language System, a collection of documents for health along with biomedical vocabularies. It includes 135 entities, 46 relations and 6,529 pairs in total.

4.1.2 Evaluation Metric. We first formally adopt True Negative Rate (TNR) [30] for anomaly detection tasks to quantify the fraction of real negative triples that are correctly identified in top K triples, which is defined as

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \tag{11}$$

where TN and FP denote the number of true negatives and false negatives, respectively. A TNR of 1 suggests perfect detection with all the top K triples being anomalies, while 0 indicates all the top K triples are not anomalies. The overall performance is proceeded by the triples and real anomalies detected from each of the three stages, which could be equivalently transformed to

$$\text{TNR} = \frac{c_1 + c_2 + c_3}{\text{K}}. \tag{12}$$

4.1.3 Hyperparameter Tuning. In this subsection, we introduce how we tune the hyperparameter, λ. This is the coefficient of squared canonical term in L2 used to equilibrate variance and bias in ridge regression. Specifically, as λ increases, the result of $\left((\mathbf{X}^{(b)(k)})^\top \mathbf{X}^{(b)(k)} + \lambda^b \mathbf{I}\right)$ will increase accordingly, which leads to the decrease of the inverse and increase of the bias. Thus, through tuning λ appropriately, we can achieve better exploitation when $\boldsymbol{\beta}$ tends to be stable. After pre-training, we utilize the returned m sets of candidate triples, and adopt a 10-fold cross validation to find the best $\lambda^{(b)}$ with the smallest variances for each detector b. Moreover, tuning of λ is model-agnostic since we take it as a prerequisite which is independent of KAEL. This allows our model to be transferable and could be easily enhanced by integrating more suitable and diverse off-the-shelf KG error detection algorithms without further parameter tuning.

Datasets	Triples	Entities	Relations	Anomalies
UMLS	6,529	135	46	326
WN18RR	93,003	40,943	11	4,650
FB15k-237	310,116	14,541	237	16281

Table 3: Statistics of datasets with an anomaly ratio of 5%.

4.2 Experimental Results

4.2.1 Comparison with KG Error Detectors. To answer **RQ1**, we compare KAEL with the base detectors, i.e., TransE, ComplEx, CKRL and KGTtm, to evaluate whether the proposed active ensemble learning strategy is effective. Results are summarized in the row of 'SOTA Detectors' in Table 2. We observe that KAEL significantly outperforms the base detectors across all KGs. For instance, KAEL achieves significant improvement of 11.5%, 8.6% and 3.8% over UMLS, WN18RR and FB15k-237 with Top@K = 5% when compared with the best baselines, and 22.7%, 43.4% and 28.7% over the lowest baseline performance on three datasets with Top@K = 1%. The performance diversity is attributed to their different designs of conventions. In such a way, they may either perform good detection performance on particular datasets or be confined by unknown error types. For instance, TransE performs the worst on UMLS but the best on FB15k-237. Consistent outperforming results of KAEL demonstrate the effectiveness of our proposed active ensemble learning strategy with a tailored MAB.

4.2.2 Empirical Analysis on Ensemble Methods. To investigate **RQ2**, we compare KAEL with the following heuristic variants of the absolute majority voting ensemble for KG error detection. (i) *Random Selection*: Except for overlaps, we complete the list to Top@K with random selection in the remaining sets of triples, in abbreviation **RS**; (ii) *Relative Majority Voting*: Obeying the superiority of majority over minority, we take all overlaps in pair level among all m candidate sets, and supplement the list with top-ranked triples until the total number of identified triples satisfies Top@K, in shorthand, **RMV**. Their results are averaged over 5 times and summarized in the row of 'Ensemble Learning' of Table 2. We get the following interesting observations. First, majority voting appears to be an effective way to improve the performance. Specifically, even

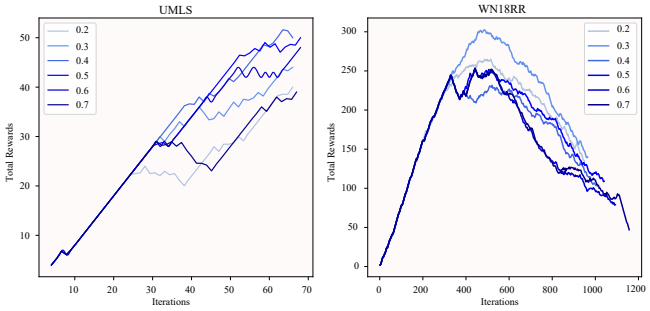

Figure 3: By varying budget ratios ∈ [0.2, 0.3, 0.4, 0.5, 0.6, 0.7], we study the impact of budgets with the smoothness of slope changes on UMLS and WN18RR with Top@K = 1%. Real errors are rewarded '1', while wrong predictions receive '-1'.

a heuristic ensemble method that combines majority voting and random selection achieves modest improvements over at least one base model on all three datasets. **RMV** exceeds at least three base detectors, particularly taking the first places on UMLS at Top@K = 3% and 5% and WN18RR. Second, we observe that KAEL consistently outperforms these two baselines by around 10% across all the datasets, which demonstrates that our MAB-based active ensemble learning strategy is effective. These results shed light on our motivation to design multi-armed bandit based ensemble policy for KG error detection.

4.2.3 Comparison with Active Ensemble Learning Strategies. To further answer **RQ2**, we consider three active-learning-enhanced ensemble learning algorithms: (i) *Oracle Involved Selection*: By replacing the random part in RMV, we query oracle with top-ranked triples wihttps://www.overleaf.com/project/615e385f2d31de116068ff38thin same budget Q as KAEL, and directly replenish the list with remaining highly-ranked ones, denoted as **OIS**. This is semi-active without learning from the answers. (ii) We adopt two typical active ensemble learning methods from AAD [31]. By employing a greedy query strategy, **AAD-IFOR** integrates a traditional ensemble learning model, Isolation Forests, and **AAD-HST** uses HS Trees. We categorize these baselines in a group of 'Active Ensemble' and the main statistics is shown in Table 2. From the disparate results, we could easily tell the significance of majority voting on prioritized suspicious triples in **OIS** compared with tree-based ensembles, as well as the superior performance of KAEL.

4.3 Budget Study for Active Querying

To answer **RQ3**, we further investigate the impact of budget Q. Intuitively, more opportunities for interaction with oracle lead to higher performance. However, this premise is not practically held in KGs since human annotations are expensive and limited. Realizing when to stop is an important feature for applying active learning to KGs. To observe when the accuracy reaches the plateau, we vary the percentage of Q in Top@K through controlled experiments. Main results are shown in Figure 3 with four base detectors combined and the anomaly ratio of 1%. We change the budget ratios ∈ [0.2, 0.3, 0.4, 0.5, 0.6, 0.7]. For evaluation, the model is rewarded '1' if the detected triple is a real anomaly, while a '−1' for a mistakenly identified normal triple. We paint the tendency lines from shallow to deep, i.e., light blue to dark blue, when the anomaly ratio increases.

The smoothness of the slope changes reflect the changing ability of detecting real anomalies at the certain point. Our observation reveals an essential homophily of choosing a proper budget for two different datasets, where the slopes commonly trend to be more stable with the ratio of budget increasing from 0.2 to 0.5, while they turn to be steeper sharply when the budget ratio rises from 0.5 to 0.7. An intuitive assumption arises from the fact that, considering the dispersion among both normal and erroneous triples, there exists a critical point of budget Q where the model can be fed to exactly recognize erroneous patterns and avoid both under-fitting and over-fitting. Otherwise, either less or more budgets will render it harder to identify errors from the remaining ranking list, as well as a waste of opportunities to query less valuable triples [32]. Thus, we uniformly set the budget ratio at 0.5 for both OIS and KAEL on three datasets in Table 2.

	3			4			5
TransE	0.282	-	-	0.282	0.282	-	0.282
ComplEx	-	0.571	-	0.571	-	0.571	0.571
DistMult	-	-	0.534	-	0.534	0.534	0.534
CKRL	0.518	0.518	0.518	0.518	0.518	0.518	0.518
KGTtm	0.534	0.534	0.534	0.534	0.534	0.534	0.534
KAEL	**0.615**	**0.645**	**0.638**	**0.669**	**0.658**	**0.680**	**0.707**

Table 4: Comparisons of different combinations with 3/4/5 base detectors on UMLS, where each column is a combination and '-' means not included in the current combination.

4.4 Analysis of Ensemble Construction

For **RQ4**, we study how different combinations of the base detectors impact the performance. Our proposed ensemble is built up with two types of base detectors in parallel, i.e., embedding-based methods and KG error detectors based on a tailored MAB algorithm, which ensures both harmony and diversity. Moreover, for embedding-based methods, TransE is the translational distance model, while ComplEx is a typical semantic matching model. Additionally, we introduce DistMult [33], a bi-linear embedding-based model, which results in five base models in the candidate pool. We consider the combinations of '3', '4', '5' base detectors to study (i) how the number of detectors and (ii) different combinations of detectors impact the performance.

The results on UMLS with Top@K = 5% are presented in Table 4, where each column represents one combination. For example, in the first column, only TransE, CKRL, and KGTtm have numbers. This indicates that we only combine these three detectors without the ones with '-' (i.e., ComplEx, DistMult). We make the following observations. First, combining more base detectors always achieve better performance. KAEL performs the best when combining all five detectors, and combinations of '4' consistently outperforms '3'. Second, integrating stronger base detectors leads to better ensemble results. Specifically, for all combination '3', combination with ComplEx outperforms both with either TransE or DistMult. In consequence, the hypothesis is developed that the more or stronger base detectors we combine, the better performance KAEL will achieve.

(mins)	UMLS	WN18RR	FB15k-237
TransE	3.77	10.53	36.46
ComplEx	4.08	12.47	38.07
DistMult	4.52	15.03	39.97
CKRL	12.40	40.10	72.55
KGTtm	16.22	52.27	77.28
KAEL	**0.51**	**7.53**	**32.68**

Table 5: Efficiency evaluation with the average running time.

4.5 Efficiency Evaluation

As both existing active learning and ensemble learning policies suffer from the efficiency problem on large datasets, to evaluate KAEL, we compare the running time with state-of-the-art detectors and summarize the results in Table 5. The computation time of KAEL on UMLS, WN18RR and FB15K-237 at Top@K = 5% is around a half minute, 10 minutes and 30 minutes, respectively, which is more efficient than the baselines; The superiority mainly benefits from two design principles. First, KAEL is dataset-agnostic since it builds upon top K triples prioritized by m base detectors. In other words, KAEL only takes $m \times K$ highly suspicious triples as input to alleviate the costs on the whole large KG, where $m \times K$ is far smaller than the original sample size n. Second, we treat base detectors as prior knowledge and exclude them to be trained by our framework. The primary time consumption is spent on the policy optimization based on important triples. KAEL proves to be adaptive and promising on larger real-world KGs.

4.6 Case Study with Visualization of Selections

To provide insights for **RQ5**, we visualize the distribution of selection times of each base detector in Figure 4 on UMLS and WN18RR with Top@K = 5% and 1%, respectively. After learning from overlaps, KAEL realizes differentiation by equilibrating the actions of exploration and exploitation in the training stage. This is interpretable as both actions can be distinctly observed in the figures, which performs as choosing the current best detector and exploring the potential one. We observe how KAEL achieves exploitation from the results of KGTtm in UMLS. Specifically, the color changes from shallow to deep, which suggests KAEL is exploiting the best detector, consequently increasing the number of selections from the candidate lists of best detectors. Similarly, the process of exploration can be sufficiently observed in WN18RR. KAEL queries more triples from the detectors that are not well-explored with the color changing from deep to shallow.

5 RELATED WORK

Knowledge Graph Error Detection. Many KG error detection methods have been proposed to distinguish noise from normal triples. The majoity of them can be categorized into two groups. (i) Embedding-based methods [34]. Except for TransE, ComplEx and DistMult [12, 25, 33], TransH [35] models a relation hyperplane for solving 1-N and N-1 problems in TransE; NTN [36] and ConvE [4] evaluate the probability between h and t w.r.t. r by applying neural networks; (ii) Path-based methods. PTransE [37] integrates the translational assumption in TransE with a multi-hop path-aware embedding model, which is followed by CKRL [18] and KGTtm [1]. Other heuristic algorithms like PaTyBRED [14] and PRGE[9] learn

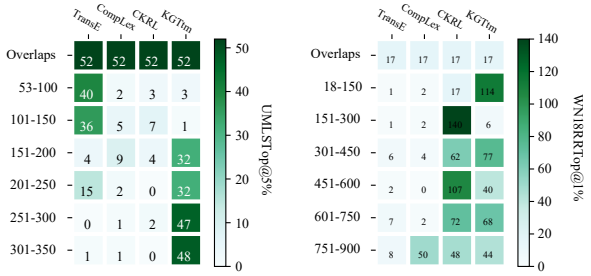

Figure 4: Visualization of selection times of base detectors.

from additional type information and propose advanced path ranking models for relation errors based on PRA [38]. As they dedicate to identifying particular KG error characteristics, in this paper, we effectively leverage strengths of both two types of detectors with a novel ensemble learning framework.

Ensemble learning for Anomaly Detection. Ensemble learning contributes prominently for traditional anomaly detection on i.i.d datasets [39–41]. It improves the generalization capability by integrating homogeneous [42, 43] or heterogeneous detectors [44] with an effective combination strategy [21, 45]. As labels are expensive to be obtained, a few attempts have been made for unsupervised training to avoid label costs [46, 47]. Most of them apply predicted labels aggregated from an estimator to train ensemble classifiers [48, 49]. However, they perform unsteadily with pseudo labels [22]. Since ensemble learning has not been explored for KG error detection, we tackle the challenges by learning from a limited number of active queries with a tailored MAB.

Active Anomaly Detection. Human-in-the-loop methods have been explored for anomaly detection for label efficiency under weak supervision [50]. Prevailing stream-based efforts search the whole space to query sequentially [24], and train the policies with different optimization objectives [31, 51–53]. Considering the intrinsic semantics and huge sizes of KGs, we focus on a concentrated space with valuable candidates, which best facilitate the query strategy.

6 CONCLUSIONS

In this paper, we present a novel active ensemble learning framework, KAEL, for KG error detection. We design a tailored multi-armed bandit algorithm to integrate multiple base detectors. A three-stage active ensemble scheme is employed to fully utilize the oracle's feedback. Experimental results demonstrate the significant superiority of KAEL over all the baselines. We also visualize the process of selection to illustrate the interpretability of our policy when solving the exploitation and exploration dilemma. For future works, we will explore further improving KAEL with more progressive combination principles for ensembles.

ACKNOWLEDGMENTS

The authors gratefully acknowledge receipt of the following financial support for the research, authorship, and/or publication of this article. This work was supported in full by the Hong Kong Polytechnic University, Start-up Fund (project number: P0033934).

REFERENCES

[1] Shengbin Jia, Yang Xiang, Xiaojun Chen, and Kun Wang. Triple trustworthiness measurement for knowledge graph. In *WWW*, 2019.

[2] Congcong Ge, Yunjun Gao, Honghui Weng, Chong Zhang, Xiaoye Miao, and Baihua Zheng. Kgclean: An embedding powered knowledge graph cleaning framework. *arXiv*, 2020.

[3] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 2017.

[4] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, 2018.

[5] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge graph embedding based question answering. In *WSDM*, pages 105–113, 2019.

[6] Daochen Zha, Louis Feng, Qiaoyu Tan, Zirui Liu, Kwei-Herng Lai, Bhargav Bhushanam, Yuandong Tian, Arun Kejariwal, and Xia Hu. Dreamshard: Generalizable embedding table placement for recommender systems. *arXiv preprint arXiv:2210.02023*, 2022.

[7] Yaqing Wang, Fenglong Ma, and Jing Gao. Efficient knowledge graph validation via cross-graph representation learning. In *CIKM*, 2020.

[8] Farhad Abedini, Mohammad Reza Keyvanpour, and Mohammad Bagher Menhaj. Correction tower: A general embedding method of the error recognition for the knowledge graph correction. *IJPRAI*, 34(10):2059034, 2020.

[9] Konstantinos Bougiatiotis, Romanos Fasoulis, Fotis Aisopos, Anastasios Nentidis, and Georgios Paliouras. Guiding graph embeddings using path-ranking methods for error detection in noisy knowledge graphs. *arXiv*, 2020.

[10] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. Amie: association rule mining under incomplete evidence in ontological knowledge bases. In *WWW*, 2013.

[11] Yanfang Ma, Huan Gao, Tianxing Wu, and Guilin Qi. Learning disjointness axioms with association rule mining and its application to inconsistency detection of linked data. In *CSWS*, 2014.

[12] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. *NeurIPS*, 26, 2013.

[13] Aibo Guo, Zhen Tan, and Xiang Zhao. Measuring triplet trustworthiness in knowledge graphs via expanded relation detection. In *KSEM*. Springer, 2020.

[14] André Melo and Heiko Paulheim. Detection of relation assertion errors in knowledge graphs. In *KCAP*, 2017.

[15] A Krishna. Ensemble learning for classification-a survey. 2020.

[16] Joao Mendes-Moreira, Carlos Soares, Alípio Mário Jorge, and Jorge Freire De Sousa. Ensemble approaches for regression: A survey. *Acm computing surveys (csur)*, 2012.

[17] Mohammad Kazim Hooshmand and Ibrahim Gad. Feature selection approach using ensemble learning for network anomaly detection. *CAAI*, 2020.

[18] Ruobing Xie, Zhiyuan Liu, Fen Lin, and Leyu Lin. Does william shakespeare REALLY write hamlet? knowledge representation learning with confidence. *AAAI*, 2018.

[19] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 2004.

[20] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 2017.

[21] Xibin Dong, Zhiwen Yu, Wenming Cao, Yifan Shi, and Qianli Ma. A survey on ensemble learning. *Frontiers of Computer Science*, 08 2019.

[22] Shebuti Rayana and Leman Akoglu. Less is more: Building selective anomaly ensembles. *TKDD*, 2016.

[23] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. YAGO3: A knowledge base from multilingual wikipedias. *CIDR*, 2015.

[24] Tivadar Danka and Peter Horvath. modal: A modular active learning framework for python. *arXiv*, 2018.

[25] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*. PMLR, 2016.

[26] Ivana Balazevic, Carl Allen, and Timothy Hospedales. Multi-relational poincaré graph embeddings. *NeurIPS*, 2019.

[27] Yifang Chen, Chung-Wei Lee, Haipeng Luo, and Chen-Yu Wei. A new algorithm for non-stationary contextual bandits: Efficient, optimal and parameter-free. In *COLT*. PMLR, 2019.

[28] Thomas J Walsh, István Szita, Carlos Diuk, and Michael L Littman. Exploring compact reinforcement-learning representations with linear regression. *arXiv*, 2012.

[29] Kurt Bollacker, Robert Cook, and Patrick Tufts. Freebase: A shared database of structured general human knowledge. In *AAAI*, 2007.

[30] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *IJDKP*, 2015.

[31] Shubhomoy Das, Md Rakibul Islam, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Active anomaly detection via ensembles: Insights, algorithms, and interpretability. *arXiv preprint arXiv:1901.08930*, 2019.

[32] Sudhanshu Mittal, Maxim Tatarchenko, Özgün Çiçek, and Thomas Brox. Parting with illusions about deep active learning. *arXiv*, 2019.

[33] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. 2014.

[34] Qinggang Zhang, Junnan Dong, Keyu Duan, Xiao Huang, Yezi Liu, and Linchuan Xu. Contrastive knowledge graph error detection. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2590–2599, 2022.

[35] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. *AAAI*, 2014.

[36] Richard Socher, Danqi Chen, Christopher D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. *NeurIPS*, 2013.

[37] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv*, 2015.

[38] Ni Lao and William W Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 2010.

[39] Abdulla Amin Aburomman and Mamun Bin Ibne Reaz. A survey of intrusion detection systems based on ensemble and hybrid classifiers. *Computers & Security*, 65:135–152, 2017.

[40] Juan Vanerio and Pedro Casas. Ensemble-learning approaches for network security and anomaly detection. In *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, pages 1–6, 2017.

[41] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *DMKD*, 2018.

[42] Vladimir Bukhtoyarov and Vadim Zhukov. Ensemble-distributed approach in classification problem solution for intrusion detection systems. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 255–265. Springer, 2014.

[43] Saman Masarat, Hassan Taheri, and Saeed Sharifian. A novel framework, based on fuzzy ensemble of classifiers for intrusion detection systems. In *2014 4th international conference on computer and knowledge engineering (ICCKE)*, pages 165–170, 2014.

[44] Ying Zhong, Wenqi Chen, Zhiliang Wang, Yifan Chen, Kai Wang, Yahui Li, Xia Yin, Xingang Shi, Jiahai Yang, and Keqin Li. Helad: A novel network anomaly detection model based on heterogeneous ensemble learning. *Computer Networks*, 169:107049, 2020.

[45] Yongquan Yang, Haijun Lv, and Ning Chen. A survey on ensemble learning under the era of deep learning. *arXiv*, 2021.

[46] Ramazan Ünlü and Petros Xanthopoulos. A weighted framework for unsupervised ensemble learning based on internal quality measures. *Annals of Operations Research*, 2019.

[47] Jia Zhang, Zhiyong Li, Ke Nai, Yu Gu, and Ahmed Sallam. Delr: A double-level ensemble learning method for unsupervised anomaly detection. *Knowledge-Based Systems*, 2019.

[48] Xupeng Zou, Zhongnan Zhang, Zhen He, and Liang Shi. Unsupervised ensemble learning with noisy label correction. In *SIGIR*, 2021.

[49] Shuang Zhou, Xiao Huang, Ninghao Liu, Qiaoyu Tan, and Fu-Lai Chung. Unseen anomaly detection on networks via multi-hypersphere learning. In *SDM*, pages 262–270. SIAM, 2022.

[50] Yao-Yuan Yang, Shao-Chuan Lee, Yu-An Chung, Tung-En Wu, Si-An Chen, and Hsuan-Tien Lin. libact: Pool-based active learning in python. *arXiv*, 2017.

[51] Daochen Zha, Kwei-Herng Lai, Mingyang Wan, and Xia Hu. Meta-aad: Active anomaly detection with deep reinforcement learning. In *ICDM*, 2020.

[52] Felix Neutatz, Mohammad Mahdavi, and Ziawasch Abedjan. Ed2: A case for active learning in error detection. *CIKM*, 2019.

[53] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *ICDM*, 2016.