IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 18, 2020, accepted June 16, 2020, date of publication June 23, 2020, date of current version July 3, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3004448

Hamming Distance Encoding Multihop Relation Knowledge Graph Completion

PANFENG CHEN[1], YISONG WANG[1], QUAN YU[2], YI FAN[2,3], AND RENYAN FENG[1]

[1]Department of Computer Science and Technology, Guizhou University, Guiyang 550025, China
[2]School of Mathematics and Statistics, Qiannan Normal University for Nationalities, Duyun 558200, China
[3]Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

Corresponding author: Yisong Wang (yswang@gzu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61976065, and in part by the Guizhou Talents' Growth Foundation of Science and Technology under Grant KY[2019]201. The work of Yu Quan was supported by the Foundations under Grant [2019]QNSYXM05, Grant QNSY2018JS010, and Grant QianNan Science [2018]10.

ABSTRACT Knowledge graphs (KGs) play an important role in many real-world applications like information retrieval, question answering, relation extraction, etc. To reveal implicit knowledge from a knowledge graph (KG), viz. knowledge graph completion (KGC), is a crucial task for the downstream applications based on KG. For this purpose various embedding-based approaches have been proposed recently. This paper proposes a new approach named HRESCAL to KGC. It extends the well-known embedding-based approach RESCAL by introducing Hamming distance-based encoder to capture implicit multihop and partial inverse relation features in a KG. Experimental results on widely used KGC benchmarks show that the new approach achieves state-of-the-art or is competitive AUC performance.

INDEX TERMS Knowledge graph, knowledge graph completion, tensor factorization, matrix factorization, Hamming distance, multihop relations.

I. INTRODUCTION

Knowledge graphs are broadly applied in information retrieval [1], question answering [2], [3], natural language processing [4], machine reading [5], etc. They can supply prior knowledge or common sense, which enables many applications to be more intelligent. Consequently, KGs are attracting an increasing amount of research interests.

Nevertheless, many KGs suffer from incompleteness [6], [7] which harms downstream tasks. To be specific, some necessary relations among existing entities are missing, and thus applying such KGs can lead to incorrect results, which causes applications broken. For this reason, KGC has attracted considerable attention and effort. Recent years, knowledge graph embedding (KGE) becomes popular approach for this task. In other words, KGC is an important application of KGE.

Our motivation in this work is manifold. Firstly, as a canonical KGE model, the tensor factorization-based model is expressive with linear time complexity and the RESCAL [8] model is proved to be fully expressive [9] for KGC task. Secondly, inspired by Bernoulli embedding [10] we conjecture that Hamming distance should be potentially beneficial for multihop relational data mining due to Hamming distance is beneficial for multihop query [10] in relational data, though it is based on adjacency matrices.

For general tensor factorization, the most typical tools are CP [11], [12], DEDICOM [13], Tucker [14] etc. Unfortunately, they are too general and thus fail to work with specific KGC efficiently. As a result, RESCAL is developed based on DEDICOM. It follows the paradigm of ASALSAN [15] for efficient training. In contrast to DEDICOM, RESCAL relaxes [8], [16] some constraints [8] and is more suitable for KG data mining. After being proposed, it has rapidly become a typical tensor factorization [16] model for the factorization-based KGC task.

DistMult [17] extends RESCAL to explicit multihop KGC, which suffers from high time cost when the hop increases. Bernoulli embedding starts a new line to capture multihop relation features for better information retrieval performance by Hamming distance encoding. These works motivate us to capture multihop relation features from the adjacency tensor to support the factorization-based model for better KGC performance. Besides, IRNs [18] is also very outstanding model for implicit multihop relation mining. Nevertheless, it is a neural network model in different research line with high cost.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Zhang.

In this paper, a novel model named HRESCAL is proposed. As an extension to RESCAL, it is a factorization-based model as a decoder and Hamming distance-based approach as an encoder to incorporate the latent features of multihop and inverse relation hidden in data. This model can capture arbitrary hop relations between any pair of entities in KG with low time cost, implicitly. Furthermore, the feature of inverse relation is also proved to be captured in the process of encoding. Even though the performance is dataset-dependent, they are competitive or outperform state-of-the-art [19].

Hamming distance based approach is the key component for our model and used as an encoder, i.e. the adjacency tensor is no longer fed to the RESCAL model straightforwardly but rather encoding by Hamming distance approach to build a new tensor and then the result is fed to the decoder RESCAL. Combining the encoder and decoder forms a complete model which is referred to as HRESCAL[1] hereafter.

The HRESCAL family contains HRESCAL2 and HRESCAL3. The former models KG as an adjacency matrix initially, as a consequence, it is only suitable for single relation or type independent relation (which means the task is relation type irrelevant since the type feature of the relation is lost when the KG data is modeled by adjacency matrix) which is inspired from Bernoulli embedding more straightforward. The latter models KG as adjacency tensor which is proven to be fully expressive for KGC tasks due to the original presentation is extended from matrix to tensor as an extension of Bernoulli embedding.

A. OUR CONTRIBUTIONS

All contributions are based on the properties of the data and Hamming distance encoding. Firstly, entity vectors are bit strings, which is why Hamming distance can be used to capture the latent features. Secondly, the entity vectors in the adjacency tensor are double meanings, that is an entity vector also includes the relations of an entity pair from this one to any other ones. Compared with previous works, our contributions are of three folds: (1) Hamming distance based approach is introduced into the KGC model to encode latent features. (2) Implicit multihop relations and partial inverse relation features are mined from KG for its completion without complex computation and any assumptions about the data. Other models such as DistMult mainly focus on explicit multihop relations mining and suffer from high cost and IRNs is not a factorization based one and with a high cost. (3) For our HRESCAL2 model, a potential method for detecting the existence of a relation in an entity pair is provided, which achieves competitive performances by an order reduced model.

B. ARRANGEMENT

The remainder of this paper is organized as follows. Related works is introduced in Section II, also a brief review of the literature is provided. In Section III, the notations and preliminaries involved in the following paper are illustrated. In Section IV, the model of this study is constructed in detail, including representation, data manipulation, and optimization. In Section V, the core algorithm and some theoretical results are displayed. To show the performance of the novel model, some experimental results are presented in Section VI. Finally, conclusions and discussions will present in Section VII together with some future works.

[1]The source code https://github.com/gzupanda/HRESCAL

II. RELATED WORKS

In this section, embedding-based KGC and Hamming distance-related embedding are introduced.

A. EMBEDDING-BASED KGC

The linear time complexity and better performance of factorization-based models become more popular in this field. As a type of embedding-based model, the entire KG is usually represented as a tensor and learning the feature by known facts and then mining new facts from new data via score computed by Eq. 5. IRM [20], Triplerank [21], and BCTF [22] fall into this category. In other words, they are all tensor factorization-based models with different assumptions or theories.

IRM and BCTF are based on a nonparametric Bayesian approach. For the sake of computation and interpretability, a series of assumptions were made. Unfortunately, they can sometimes be unrealistic. Moreover, Triplerank is dedicated to relevance ranking of the semantic web and thus is not much suitable for KGC.

RESCAL is proposed for their improvement as a multi-relational data mining model based on previous works. It is also the most related to the model presented in this paper and is a typical embedding-based model for KGC tasks. A series of papers then followed this work such as DistMult, HolE [23], ANALOGY [24], and ComplEx [25]. Recently, the ComplEx model introduced the concept of complexity into the factorization-based model, starting a new line of research. By contrast, they either can capture one-hop relation feature in KG data only or capture multihop relation explicitly with high cost.

DistMult is an early extension to RESCAL for multihop relation KGC. It captures multihop relation features explicitly by the multiplication of relation matrices. To capture n-hop ($n \geq 2$) relation features, n times $N_e \times N_e$ matrices multiplication are required. The high complexity of time and space make it hard to be extended to more hops. To alleviate the problem of high cost, Hamming distance [10] based approach is introduced for implicit multihop relation features encoding.

Recently, there is an extension to RESCAL which is close to our work directly. Meanwhile, it requires a lot of tensor operations which increases the cost of computation greatly. In short, it is expressive but expensive. Moreover, they can only model one-hop relation latent features.

B. HAMMING DISTANCE-BASED EMBEDDING

The Bernoulli embedding model is devised to improve the latency of information retrieval. In spite of the KGC

performance concerned, Hamming distance used for capturing more semantic features is also rewarding. This inspires us that the row of adjacency tensor is also a bit string, which can be adopted to promote our work, too.

III. NOTATIONS AND PRELIMINARIES

The notations and preliminaries used in the following are demonstrated in this section.

A. NOTATIONS FOR HRESCAL

They are listed in TABLE 1. Note that **h**, **r**, and **t** denote the head, relation and tail **vector embedding** which follows a published paper [26] and it is termed **vector** for short. Thanks to the HRESCAL3 is fully expressive [27] for the KGC task and HRESCAL2 is a weaker model only for single relation or relation type independent KGC, all descriptions throughout this paper are about both HRESCAL2 and HRESCAL3 models unless it is mentioned explicitly.

TABLE 1. Summary of the notations in this paper.

Tensor		
Notation	Meaning	Domain
$\mathcal{Y}$	third-order tensor	$\mathbb{R}^{N_e \times N_e \times N_r}$
$\mathcal{R}$	factors tensor	$\mathbb{R}^{N_R \times N_R \times N_r}$
$\mathcal{Y}^a$	adjacency tensor	$\{0,1\}^{N_e \times N_e \times N_r}$
$\mathcal{Y}^h$	$\mathcal{Y}^a \xrightarrow{Eq.2} \mathcal{Y}^h$	$\mathbb{R}^{N_e \times N_e \times N_r}$
$\mathcal{Y}^d$	$\mathcal{Y}^h \xrightarrow{Eq.9} \mathcal{Y}^d$	$\mathbb{R}^{N_e \times N_e \times N_r}$
$\mathcal{Y}^p$	$\mathcal{Y}^d \xrightarrow{Eq.10} \mathcal{Y}^p$	$\mathbb{R}^{N_e \times N_e \times N_r}$
$\mathcal{Y}^q$	$\mathcal{Y}^p \xrightarrow{Eq.11} \mathcal{Y}^q$	$\{0,1\}^{N_e \times N_e \times N_r}$
Matrix		
Notation	Meaning	Demo
$\mathbf{R}_{::i}$	the ith slice of tensor $\mathcal{R}$	$\in \mathbb{R}^{N_e \times N_e}$
$\mathbf{H}$	the latent features of head entities	equal to $\mathbf{E}$
$\mathbf{T}$	the latent features of tail entities	equal to $\mathbf{E}^\top$
$\mathbf{E}$	the latent all features of entities	the factor matrix
$\mathbf{Y}_{::i}$	the ith slice in tensor $\mathcal{Y}$	$\in \mathbb{R}^{N_e \times N_e}$
Vector		
Notation	Meaning	Demo
$\mathbf{E}_{i:}$	the embedding of entities	the ith row in $\mathbf{E}$
$\mathbf{h}_i$	the ith embedding [26] of head entity	the ith row in $\mathbf{H}$
$\mathbf{t}_i$	the ith embedding of tail entity	the ith row in $\mathbf{T}$
$\mathbf{r}_i$	the ith embedding of relation	the ith row in $\mathbf{R}_{::k}$
$\mathbf{y}_{:jk}$	the jth column of $\mathbf{Y}_{::k}$	a fiber in tensor $\mathcal{Y}$
$\mathbf{y}_{i:k}$	the ith row of $\mathbf{Y}_{::k}$	a fiber in tensor $\mathcal{Y}$
Scalar		
Notation	Meaning	Domain
N_e	the number of entity	$\mathbb{R}$
N_r	the number of relation	$\mathbb{R}$
N_R	the number of rank	$\mathbb{R}$
e_i	the ith entity	$\mathbb{R}$
h_i	the ith head entity	$\mathbb{R}$
t_i	the ith tail entity	$\mathbb{R}$
r_i	the ith relation	$\mathbb{R}$
y_{ijk}	the entry in tensor $\mathcal{Y}$	$\mathbb{R}$
Other Notations		
Notation	Meaning	
$\mathbb{R}$	the real space	
(h, r, t)	any triple (fact) in KG	
$(h, r, t; \ell)$	ℓ is likelihood for the existence of fact (h, r, t).	
$f_{r_k}(h_i, t_j)$	the score of the kth relation between ith and jth entities.	
$r(h, t)$	the relation from entity h to t	
$r^{'}(t, h)$	the relation from entity t to h and it is the inverse of $r^{'}(h, r)$	

B. PRELIMINARY: HAMMING DISTANCE

Hamming distance is originally introduced for error detection and correction [28]. Recently, Misra and Bhatia *et al.* [10] proposed a model that introduces Hamming distance to deal with information retrieval from relational data which is referred to as Bernoulli embedding. Relations between any pair of entities are parameterized by the probabilities instead of independent Bernoulli random variables to avoid NP-hard bit embedding optimization tasks [10].

1) HAMMING DISTANCE ENCODING

Hamming distance was first proposed by Hamming [28]. It is defined as the distance between two bits x and y in the coordinates to show the difference between them. In the context of KGC, it can be understood as the distance among entity vectors or embeddings. As we know, Hamming distance is bit-wise exclusive or (XOR, the mathematical operator is $\oplus$). As a result, Hamming distance between the ith and jth entity vectors in kth slice is defined as in [29]

$$y^h_{ijk} = \texttt{count}(\mathbf{y}^a_{i:k} \oplus \mathbf{y}^a_{j:k}) \tag{1}$$

where y^h_{ijk} is the result of counting the different bits between the vectors (also bit strings) $\mathbf{y}^a_{i:k}$ and $\mathbf{y}^a_{j:k}$ bit-wisely, which is Hamming distance. Owning to computational convenience, the computation of Hamming distance in Bernoulli embedding model is also followed here.

$$y^h_{ijk} = (\mathbf{y}^a_{i:k})^\top(\mathbf{1} - \mathbf{y}^a_{j:k}) + (\mathbf{1} - \mathbf{y}^a_{i:k})^\top \mathbf{y}^a_{j:k} \tag{2}$$

C. PRELIMINARY: FACTORIZATION-BASED KGC

Tensor factorization model RESCAL is used as a decoder in our model.

1) ORIGINAL REPRESENTATION

The adjacency tensor is an instinctive and popular representation for a graph.

$$y^a_{ijk} = \begin{cases} 1, & \text{if the fact } (e_i, r_k, e_j) \text{ exists;} \\ 0, & \text{not existing or unknown.} \end{cases} \tag{3}$$

where $y^a_{ijk} \in \{0, 1\}$ is an entry of the adjacency tensor $\mathcal{Y}^a$, which indicates whether the kth relation between entity i and j exists. (e_i, r_k, e_j) is a fact composed of ith and jth entity with kth relation. i, j and k are the order number of entities and relations, respectively. In the adjacency tensor, y^a_{ijk} is a point of (i, j) in kth slice. It is also a straightforward representation of a KG. The entry of the adjacency tensor denotes the existence of relation by 0 and 1. Previously, for the sake of computing convenient or interpretability, 0 and 1 in adjacency tensor was usually assumed to follow the Bernoulli distribution [10]. Nevertheless, the entity vectors, which are also bit strings, was exploited only to a small extent

for long until [10] utilized them to capture latent relational feature by Hamming distance. For HRESCAL2, the original representation is an adjacency matrix and is similar to the Bernoulli embedding model where the relations are not distinguished. In other words, this model only cares about whether a relation exists and ignore its type.

2) TENSOR FACTORIZATION

Tensor factorization has been very widely applied in psychometrics and chemometrics for a long time [16]. Recently, these models have been used for relational learning which can be used for KGC tasks.

Typically, the KG modeled as an adjacency tensor, and get the factor matrices by factorization. Eq.4 shows how the factorization is conducted slice-wisely.

$$\mathbf{Y}^q_{::k} \approx \mathbf{E}\mathbf{R}_{::k}\mathbf{E}^\top = \mathbf{H}\mathbf{R}_{::k}\mathbf{T} \tag{4}$$

where $\mathcal{Y}^q$ is a new Boolean tensor similar to adjacency one, and its factor matrices are $\mathbf{E}$ and $\mathbf{R}_{::k}$.

After factorization slice-wisely, there are three factor matrices $\mathbf{H}$, $\mathbf{R}_{::k}$, and $\mathbf{T}$ which are including latent components. Every row of $\mathbf{H}$ and is considered to be the head entity embedding, $\mathbf{R}_{::k}$ is considered to be the kth relation embedding [26] for the whole KG and the row of $\mathbf{T}$ is the tail entity embedding. According to this information, the entry of reconstructed tensor can be computed by Eq. 5

$$f_{r_k}(\mathbf{h}_i, \mathbf{t}_j) = \mathbf{e}_i^\top \mathbf{R}_{::k}\mathbf{e}_j = \mathbf{h}_i^\top \mathbf{R}_{::k}\mathbf{t}_j \tag{5}$$

where $\mathbf{h}_i$ and $\mathbf{t}_j$ are the rows of $\mathbf{H}$ and $\mathbf{T}$, respectively. The scores are the entries of the reconstructed tensor, which means that $y^p_{ijk} = f_{r_k}(\mathbf{h}_i, \mathbf{t}_j)$. From the perspective of probability [30], it is can be considered as the likelihood of the relation to what extent existing in a fact.

3) TRAINING AND OPTIMIZATION

There is no closed-from solutions for Eq. 4 to obtain factorized matrices. The factor matrices $\mathbf{E}$, $\{\mathbf{R}_{::k}\}$ are computed by minimizing the following objective function.

$$\min_{\mathbf{E},\{\mathbf{R}_{::k}\}} \texttt{loss}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right) + \texttt{reg}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right) \tag{6}$$

where $\texttt{loss}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right)$ is the loss function which aims to seek for minimal loss between the output and input tensor.

$$\texttt{loss}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right) = \frac{1}{2}\left(\sum_{k=1}^{N_r} \left\| \mathbf{Y}^q_{::k} - \mathbf{E}\mathbf{R}_{::k}\mathbf{E}^\top \right\|_{L_2}\right) \tag{7}$$

To prevent the loss function from overfitting in the process of optimization, two regularization terms are added to form a function $\texttt{reg}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right)$.

$$\texttt{reg}\left(\mathbf{E}, \{\mathbf{R}_{::k}\}\right) = \frac{1}{2}\left(\lambda_E \|\mathbf{E}\|_{L_2} + \lambda_R \sum_{k=1}^{N_r} \|\mathbf{R}_{::k}\|_{L_2}\right) \tag{8}$$

where L_2 stands for the L_2 norm. Particularly, $\lambda_R \sum_{k=1}^{N_r} \|\mathbf{R}_{::k}\|_{L_2}$ is a sum of all relation slices if there are more than one relational matrices, and $k = N_r = 1$ for HRESCAL2. RESCAL is trained by 10 fold cross validation and optimized by the alternating least square (ALS) [8].[2]

D. PRELIMINARY: LINK PREDICTION

Link prediction is considered to be the key task of KGC [31]. Usually, the prediction of a new fact there are three forms like $(h, r, ?)$, $(h, ?, t)$ or $(h, r, t; \ell)$. In the context of KGC, there are two implementations for factorization-based models. The one is computing the score and ranking for the best ordered entities as candidates similar to TuckER [27] by Eq. 5, the other is obtaining the likelihoods of relations existence by Eq. 4. The former usually corresponding to the form of $(h, r, ?)$ and $(?, r, t)$. The later is trained by collective learning which is applied for RESCAL as well as our model HRESCAL and corresponding to the form of $(h, r, t; \ell)$. Additionally, the two different implementations usually report different metrics for evaluation.

IV. HRESCAL: MODEL CONSTRUCTION

The proposed model is composed of encoder and decoder as Fig. 1 shows.

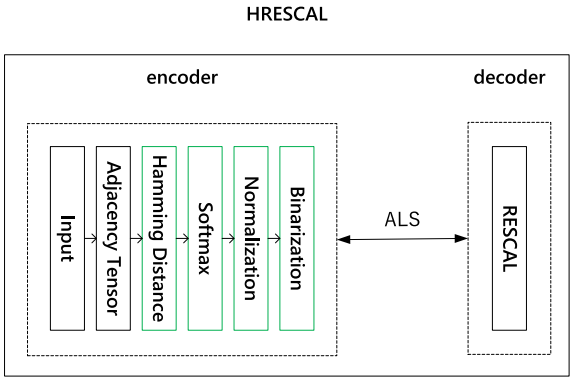

FIGURE 1. Simple illustration of the framework HRESCAL. It is composed of encoder and decoder overall. The green boxes are the work of this study.

A. ENCODING: REPRESENTATION AND HANDLING

In KGC, entities are encoded as nodes while edges are encoded as relations, and the aim is to find out latent edges in the graph. Also such a representation of KGs determines the succeeding steps to some extent. As for the original representation, the Eq. 3 is followed.

1) REPRESENTATION: FOR RESCAL2 AND RESCAL3

a: FOR HRESCAL3

Intuitively, representing Resource Description Framework (RDF)[3] triples as a third-order tensor is reasonable and straightforward. Every frontal slice of it denotes a relation with an $N_e \times N_e$ matrix and every slice keeps a type of relation

[2]When we construction loss function with regularized loss and optimize it by stochastic gradient descent (SGD).

[3]https://www.w3.org/RDF/

information. This means that the kth slice only captures the kth type of relations between all entity pairs, and the relation information is maintained in this way.

Meanwhile, it may not be necessary and also consumes a great amount of resources for certain kinds of tasks such as those involving KGs without relation type. This means that considerable redundant information may consume extra space and time. Given that this approach is full expressive [9], its cost may be too great. To model this type of KG for KGC task, a matrix is presumably sufficient in spite of it is not so expressive as a tensor.

b: FOR HRESCAL2

As mentioned above, this work is inspired by Bernoulli embedding. The relational data is modeled as a second-order tensor (adjacency matrix) for the purpose of information retrieval. Considering that it can be used for some kind of KGC tasks, the second-order version model HRESCAL2 also is discussed, briefly.

$$y^a_{ij} = \begin{cases} 1, & \text{if the fact } (e_i, r_{ij}, e_j) \text{ exists;} \\ 0, & \text{not exist or unknown.} \end{cases}$$

where y^a_{ij} is an entry of an adjacency matrix. This is the original representation of HRESCAL2. For subsequent decoding, it can be considered as an one slice third-order tensor and safely fed to decoder.

Furthermore, every slice corresponds to one-time singular value decomposition (SVD), whose time complexity is $O(N_e^3)$. This means that the time complexity is reduced from $O(N_e^3 N_r)$ to $O(N_e^3)$ after order reduction in theory just as Fig. 2 shows. Similar situations also occur as to the space complexity. Hence, in HRESCAL2, the adjacency tensor size is reduced from $N_e \times N_e \times N_r$ to $N_e \times N_e$.

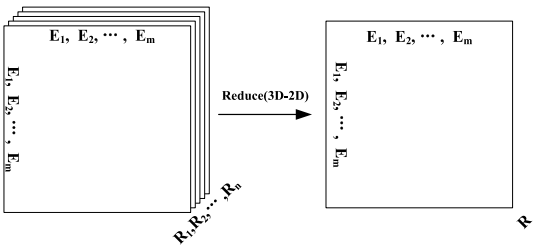

FIGURE 2. Simple illustration of order reduction. The left is third-order adjacency tensor and the right is a second-order one. This operation is for HRESCAL2 only.

The key difference between HRESCAL2 and HRESCAL3 is how they model the KG originally, i.e. they represent KG as adjacency matrix or tensor. It determines the potential capacity of model. Besides, one KG corresponds to one adjacency matrix in HRESCAL2, thus only one time of matrix factorization is required, and the training time is greatly reduced. After encoding by Hamming distance, the result tensor is projected by softmax, following the Skip-Gram [32] paradigm as used in Bernoulli embedding model. Therefore, it does not increase the time complexity because it is done before training.

2) MANIPULATION: HAMMING DISTANCE AND MAPPING

Hamming Distance Mapping: The result of Hamming distance encoding is an integer and usually neither 0 nor 1.

$$y^d_{ijk} = \text{softmax}_{j:k}\left(\lambda_h y^h_{ijk}\right) = \frac{e^{\lambda_h y^h_{ijk}}}{\sum_j^{N_e} e^{\lambda_h y^h_{ijk}}} \tag{9}$$

where $i:k$ is the ith row and any column in the kth slice, so it denotes the ith row in kth slice. In addition, all values of Hamming distance encoding by Eq. 2 are projected onto a real space by softmax row-wisely. The result is a real tensor $\mathcal{Y}^d$. Likewise, it is normalized as follows.

$$y^p_{ijk} = \text{normalize}\left(y^d_{ijk}\right) = \frac{y^d_{ijk} - \min(\mathbf{Y}^d_{::k})}{\max(\mathbf{Y}^d_{::k}) - \min(\mathbf{Y}^d_{::k})} \tag{10}$$

where $\mathcal{Y}^p$ is normalized from the $\mathcal{Y}^d$ and 0-1 normalization is used. Consequently, all values are in the same real space in every slice. Thus, the subsequent step is to convert them into Boolean values by **threshold** θ for the factorization step.

$$y^q_{ijk} = \text{binarization}(y^p_{ijk}) = \begin{cases} 1, & \text{if } y^p_{ijk} \geqslant \theta; \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

B. DECODING: RESCAL VS HRESCAL FAMILY

The HRESCAL family is composed of HRESCAL2 and HRESCAL3. After encoding, the latent features are captured from Hamming distance encoding working along with the whole process. Moreover, the networks of the score are illustrated with Fig. 3 which shows how the embeddings work for link prediction. They are obtained from the last step with information gathered from KGs in previous steps. HRESCAL is based on RESCAL. Moreover, HRESCAL2 needs to run the factorization algorithm only once, while RESCAL and HRESCAL3 run the operation N_r times. As for HRESCAL2, one distinguished feature is that the originally fed data are no longer rows of the adjacency tensor in some slice but rather the one that absorbs some features from an entity and its neighbors, which is denoted by the one blue circle around six circles. In other words, it contains more latent features captured from the raw data.

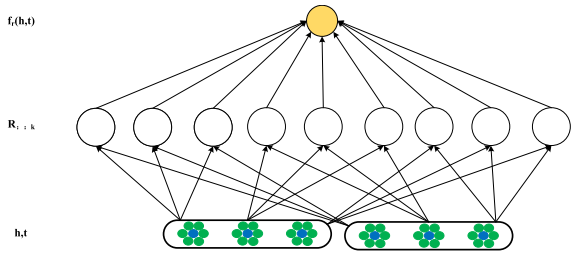

FIGURE 3. Simple illustration of HRESCAL as neural networks and it is adapted from [23].

V. ALGORITHM AND THEATRICAL RESULTS

A. ALGORITHM

Algorithm 1 here is to show the notion of the work in this study. It displays the algorithm of HRESCAL3 only and can be adapted to HRESCAL2 easily.

Algorithm 1 HRESCAL

Input: A knowledge graph $\mathcal{K}$
Output: Collective learned tensor $\mathcal{Y}$
$\mathcal{Y}^a \leftarrow$ computing the adjacency tensor of $\mathcal{K}$;
$\mathcal{Y}^h \leftarrow$ computing Hamming distance tensor of $\mathcal{Y}^a$ in terms of Eq. 2;
$\mathcal{Y}^d \leftarrow$ computing the softmax of $\mathcal{Y}^h$ in terms of Eq. 9;
$\mathcal{Y}^p \leftarrow$ computing the normalization of $\mathcal{Y}^d$ in terms of Eq. 10;
$\mathcal{Y}^q \leftarrow$ computing the binarization of $\mathcal{Y}^p$ in terms of Eq. 11;
return RESCAL($\mathcal{Y}^q$);

The adjacency tensor $\mathcal{Y}^a$ is obtained by modeling it as an ordinary graph. And then Hamming distance is computed on the basis of this tensor and obtain a new one $\mathcal{Y}^h$, and the succeeding result tensors are $\mathcal{Y}^d$, $\mathcal{Y}^p$ and $\mathcal{Y}^q$ after softmax projecting, normalizing and binarizing, respectively. In doing so, the latent semantic component of triples are captured and collective learned data are more interpretative. Finally, the RESCAL model runs, and the prediction result is obtained in the form of tensor $\mathcal{Y}$ which gets close to the factorized tensor $\mathcal{Y}^q$.

B. THEATRICAL BASIS

A KG is treated as a graph and its adjacency tensor is computed in our model, therefore, whether it follows a Bernoulli distribution or not is no longer concerned with the model running. As a result, the rows are bit strings is sufficient for the proposed model. Owning to the works on Bernoulli embedding also indicate that just mapping them by softmax is hard to optimize [10]. Fortunately, the introduction of Hamming distance encoding, which is beneficial for measuring the semantic similarity. This founding is promising and has been transferred from bit string to image searching [29] and relational data learning [10]. On the contrary, it is not so instinctive for semantic web mining. Furthermore, the work on the Bernoulli embedding model proves that it is indeed beneficial for relational data mining, which is close to our HRESCAL2 model. Hamming distance related operations are favorable to KGC tasks, and this point is strengthened by our experiments in Section VI. Additionally, there are some properties of adjacency tensor and Hamming distance encoder that serve as foundations of the framework presented in the context of KGC.

Proposition 1: Every entity vector from adjacency tensors is entity-relation double meanings.

Proof 1: It is easy to understand this property by the Eq. 4. That means every row in a frontal slice of adjacency tensor can be factorized into the product among entity and relation factors (vector or matrix). That is

$$\mathbf{y}^a_{i:k} \approx \mathbf{E}_{i:}(\mathbf{R}_{::k}\mathbf{E}^\top)$$

where $\mathbf{y}^a_{i:k}$ is the ith row of reconstructed tensor of $\mathcal{Y}^a$ in kth frontal slice. This equation holds according to RESCAL. It shows that the ith row in kth slice can be factorized into product of entity embedding $\mathbf{E}_{i:}$, $\mathbf{E}$ and relation matrix $\mathbf{R}_{::k}$. Hence, the row contains the features of entity and relation at the same time.

Proposition 2: The encoder can express anti-reflexive relation.

Proof 2: Let $\mathbf{y}^a_{i:k} = \mathbf{y}^a_{j:k}$ in kth slice, then $\mathbf{y}^{a\top}_{i:k}(\mathbf{1}-\mathbf{y}^a_{j:k}) = \mathbf{y}^{a\top}_{j:k}(\mathbf{1}-\mathbf{y}^a_{j:k}) = \mathbf{0}$ holds. This means that Eq. 2 takes out the information from the node vector itself.

Proposition 3: The encoder can express reversible relation, partially.

Proof 3: Let $r_k(e_i, e_j)$ denotes the kth relation between ith entity and jth neighbour, and $\neg r_k(e_i, e_j)$ is its inverse one which can be also denoted as $r_k(e_j, e_i)$. Because it is a Boolean value, equation $1 - r_k(e_i, e_j) = \neg r_k(e_i, e_j) = r_k(e_j, e_i)$ holds in this sense. Consequently, the Eq. 2 can be rewritten as

$$y^h_{ijk} = \mathbf{y}^{a\top}_{i:k}(\neg\mathbf{y}^a_{j:k}) + (\neg\mathbf{y}^a_{ijk})^\top\mathbf{y}_{j:k}$$

where y^h_{ijk} is Hamming distance between ith and jth entities in kth slice of tensor $\mathcal{Y}^h$. Eq. 3 illustrates that Hamming distance encoding capturing the inverse relation features from head and tail entities.

In translation-based models [31], [33], reversible relation samples in KG dataset are usually considered to be negative ones and thus filtered out to improve performances. On the contrary, in the model HRESCAL, they are taken to provide more information for better performances. Further explanations of how and why they manage this are as follows.

Admittedly, these features absorbed only from a part of inverse relations, not all. TABLE 2 is used to illustrate this point. It shows the four cases of observable relation, observable inversion relations, both observable and latent relations as well as the relations can be encoded by Hamming distance encoder, they are denoted as $r_k(e_i, e_j)$, $r_k(e_j, e_i)$, $r'_k(e_j, e_i)$ and y^d_{ijk}, respectively. The *State* column record the state of whether the inverse relations are captured or not. ✓ denotes the features have captured in the model. Case 1 and 4 in the table demonstrate that the inverse relations either missing or

TABLE 2. Which kinds of inverse relations can be captured.

Case	$r_k(e_i, e_j)$	$r_k(e_j, e_i)$	$r'_k(e_i, e_j)$	y^h_{ijk}	*State*
1	Exist	Not Exist	Exist	Not Exist	Missing
2	Exist	Not Exist	**Not Exist**	**Not Exist**	✓
3	Not Exist	Exist	**Exist**	**Exist**	✓
4	Not Exist	Exist	Not Exist	Exist	Redundant

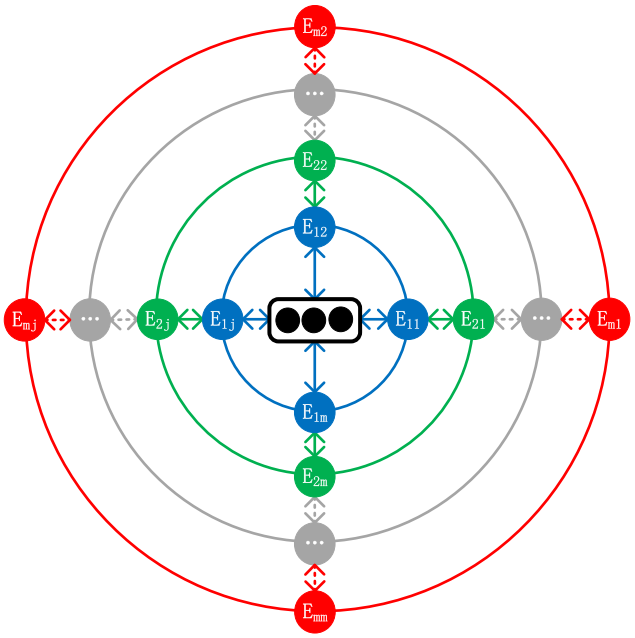

FIGURE 4. Simple illustration of how an entity-relation layer absorbs the features from all its reachable neighbors by Hamming distance encoder.

get false relation features. This is the reason why the proposed model can't get 100% of link prediction on dataset `Kinship` like other models. Case 2 and 3 are the newfound inverse relations since Hamming distance encoding results y^h_{ijk} are consistent with the inverse relation $r_k(e_j, e_i)$ in real KG.

Proposition 4: The encoder can capture multihop relation features.

This property is desired but its proof remains open. This point is explained as follows. To better understand this point, the row (or column) of any frontal slice of adjacency tensor is termed as an **entity-relation layer**. When computing Hamming distances, the entity vector will have to multiply all rows except itself. This situation is similar to the information spread from one layer to another. As a result, an entity vector absorbs the features of all their reachable neighbors in the end. In a multihop relation path, the nodes capture the features of all their reachable neighbors which they pass by and skip themselves along with it. This is why it is considered to capture multihop relation features. Fig. 5 illustrates the meaning of the entity-relation layer and how it captures multihop relation features.

Fig. 5 demonstrates that the vectors from any frontal slice absorb the information by Hamming distance encoding. Eq. 2 is beneficial for better understanding this point. The entry of the new tensor is computed by Eq. 2 from which can be found the entity vector in every slice of the tensor absorbs features of all its reachable neighbors. In other words, the observed component $\mathbf{y}_{i:k}$ is a row of the kth slice in adjacency tensor and is also the relation between all its reachable neighbors, and $(\mathbf{1} - \mathbf{y}_{j:k})$ is the set of inverse relation between all of its reachable neighbors. And then, they interconnect by multiplying. Consequently, the ith head entity vector connects all

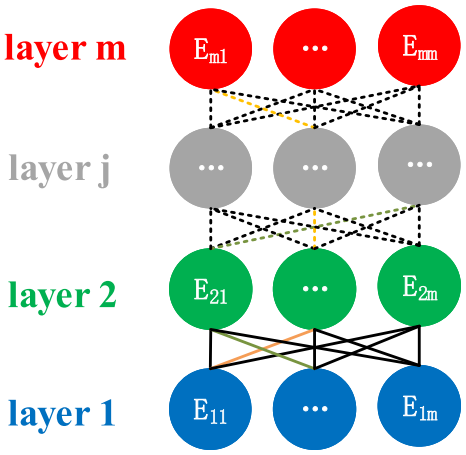

FIGURE 5. Simple illustration of multihop features capturing layer by layer.

other reachable neighbors in this way, and the jth entity vector acts in the same manner. Finally, every entry from 1 to N_e absorbs features of all their reachable neighbors in this way. This process is similar to a fully connected network, where every node interacts with all of the other nodes in the whole KG. It ensures that every entry of $(\mathbf{1} - \mathbf{y}_{j:k})$ captures more features from all other $(N_e - 1)$ nodes around it except for itself. This means that to obtain all entities of KG, $O(N_e^2)$ vector dot product operations are required. Consequently, a vector of Hamming distance involves all vectors of entities and their reachable neighbors. To explain how Hamming distance encoder captures the features, a figure is drawn to assist understanding, as Fig. 5 shows.

In HRESCAL, every entity vector captures the information among the respective entity and all of its reachable neighbors. As a result, an entry in tensor produced by Hamming distance encoding absorbs all nodes between their reachable neighbors. In this sense, any entity in this tensor is potentially reachable to anyone in the whole KG. Hence, it captures multihop relation features as a vector if they exist. All of them are absorbed by multiplying among vectors. If every entity absorbs features from their reachable neighbors, thus it may capture the feature of multihop relations in the end. Every entity vector can be considered as a layer and the information propagates layer by layer. And then, the last related (not reaching the last layer) entity vector (represented as a vector layer) is reached, forming a multihop relation path. In doing so, a multihop relation is encoded, and the features absorbed. It can be found that every entity vector is fully connected to its neighbors or the next layer and that the relations hidden in any entity pair is captured. For a multihop relation path, there are two types: m-length and j-length. For instance (refer to Fig. 5), there may exist path as follows: $E_{11} \rightarrow E_{22} \rightarrow \cdots \rightarrow E_{m1}(m = N_e)$. In most cases, the length of the path is not as long as $(N_e - 1)$. Here is another example. The path shown with green color in Fig. 5 is $E_{12} \rightarrow E_{21} \rightarrow \cdots \rightarrow E_{j.}(1 < j < (N_e - 1))$. This path is shorter than $(N_e - 1)$ but is more than one. Hence, they are all multihop relations.

TABLE 3. The datasets used in the experiments.

Dataset	N_{Entity}	$N_{Relation}$	N_{Train}	N_{Valid}	N_{Test}
Kinship	104	26	8,544	1,068	1,074
Nations	14	56	1,592	199	201
UMLS	135	49	5,216	652	661
Countries	224	23	1,111	24	24
FB15K	14,951	1,345	483,142	50,000	59,071
FB15K237	14,505	237	272,115	17,535	20,466

Notably, it is have to highlight that this approach does not explicitly implement multihop relation reasoning because it is highly coupled but just captures the information of the multihop in a single entity vector layer. It works with Hamming distance encoding as a whole, and it is unable to determine which part is contributed by which piece of information and from which node. As a result, Hamming distance mainly encodes the similarity of two strings of a signal or picture bit-wisely while in the context of KGC, it is given a new meaning. Every bit is the likelihood of a relation. As a matter of fact, it no longer refers to similarity but to what extend there exists a relation of the pair of a node to any of another entity from the KG. The information from reachable neighbors is coupled in Hamming distance as a vector causing the problem of interpretability.

VI. EXPERIMENTS

To empirically evaluate the model proposed for the KGC task, link prediction experiments on the widely used datasets are conducted.

A. DATASETS

To evaluate our model, six datasets are introduced. All datasets can be divided into three groups. The first group includes `Kinship`, `Nations`, and `UMLS`. These are the original datasets used for the baseline models. `Countries` is a relational dataset with rule constraints. `FB15K` and `FB15K237` are large-scale datasets used for evaluating the scalability of models.

1) KINSHIP

This dataset was created by Denham [20]. It includes anthropological data of relations from the Central Australia tribe called Alyawarra.

2) NATIONS

This dataset was produced during the Cold War [34]. The triples reflect the relations between socialist and capitalist camps.

3) UMLS

This dataset was gathered by McCray [35]. It is from a special medical ontology called the Unified Medical Language System.[4] Therefore, this dataset is a subset of it.

[4]https://www.nlm.nih.gov/research/umls/index.html

4) COUNTRIES

This dataset was gathered by Bouchard *et al.* [36]. It is used for testing the model capacity for inference. There are 5 regions and 23 subregions, and they contain 224 countries. It includes 1158 facts. **S1**, **S2**, and **S3** are conditions that are increasingly more strict. More details can be found in [36]. In [37], there are newer results.

5) FB15K

This dataset was gathered by Bordes *et al.* [31] to evaluate their model TransE. It is a subset of the real KG of Freebase[5] and has become one of the benchmark standard datasets for evaluating a new model. It is usually used for evaluating the link prediction capacity of a KGC model.

6) FB15K237

This dataset is a subset of `FB15K` that is more of a challenge. Toutanova and Chen [38] believe that `FB15K` is test leaking for some models which get negative samples by swapping head and tail entities. But this problem is not existing in non-negative factorization model such as RESCAL. It is usually considered to be more challenging than `FB15K`.

All datasets are initialized to matrix or tensors in `.mat` format. Namely, the first group are $104\times104\times26$, $135\times135\times49$ and $14\times14\times56$ and the second group are the same size of $271\times271\times2$. Before running, they have to be converted into $104\times104\times1$, $135\times135\times1$, $14\times14\times1$ and $271\times271\times2$. Furthermore, the last two datasets are considerably large: they are $14951\times14951\times1345$ and $14505\times14505\times237$ and it is run in HRESCAL2 only due to their size. The results are based on these datasets. All datasets are prepossessed into MATLAB format tensor for further steps.

B. EXPERIMENTAL SETUP

1) RUNNING ENVIRONMENT AND EXPERIMENTAL SETUP

The algorithm HRESCAL is implemented by Python 3.6 and runs on a computer with an Intel Core i7-7700 CPU that has 8 cores with a 3.60 GHz main frequency and 8 GB RAM. The tensor factorization component is developed by Maximilian *et al.*.[6] All other codes are also tested in Python 3.6.

2) HYPERPARAMETERS

There are five hyperparameters in HRESCAL model. They are λ_h, N_R, λ_E, λ_R and θ, respectively. λ_h is the coefficient of Hamming distance for adjusting it to be appropriate scale to operate with other data together which can be negative or positive. N_R is the rank of recovered matrices. The larger the number, the more time it takes. λ_E and λ_R are the coefficients of regular for adjusting of them. They are used to prevent the objective function overfitting when optimization. They can be different, but they are set the same for simplicity in the

[5]http://www.freebase.be/

[6]RESCAL with the originally implemented Python source code is available at: http://www.cip.ifi.lmu.de/ nickel

TABLE 4. The best hyperparameters in the experiments.

Dataset	HRESCAL	Hyperparameters			
		λ_h	N_R	λ_E	λ_R
Kinship	3-order	-0.02	10	20	20
	2-order	0.005	2	1	1
Nations	3-order	-0.0005	3	1	1
	2-order	0.003	3	1	1
UMLS	3-order	-0.0005	50	0.5	0.5
	2-order	-0.0005	50	0.5	0.5
Counties S1	3-order	-0.00005	2	0.02	0.02
	2-order	-0.00005	2	1	1
Counties S2	3-order	-0.0005	3	0.5	0.5
	2-order	-0.0005	2	1	1
Counties S3	3-order	-0.0005	30	0.1	0.1
	2-order	-0.0005	100	2	2
FB15k	2-order	0.005	10000	10	10
FB15k237	2-order	0.005	10000	10	10

experiment. $\theta \in (0, 1)$ is a threshold which determines the relation score is 1 or 0. There are no standard to determine this hyperparameter, it is set 0.5 by default.

When training, the four parameter values that have to be set as TABLE 4 shows. These values are tuned manually. $\lambda_h \in \{-1, -0.5, -0.3, -0.2, -0.1, -0.05, -0.005, -0.0005, -0.00005, 0.00005, 0.0005, 0.005, 0.05, 0.1, 0.2, 0.3, 0.5, 1\}$, $N_R \in \{1, 2, 3, 5, 10, 20, 30, 50, 100, 1000, 10000\}$, $\lambda_E(\lambda_R) \in \{0.05, 0.1, 0.5, 1, 2, 3, 5, 10, 20, 50, 100\}$, respectively. Note that, the dataset `Nations` only with 14 dimensions, therefore, the N_R is less than 14. For datasets `FB15k` and `FB15k237`, they are time and space exhausting for our equipment. As a consequence, only HRESCAL2 is evaluated in this experiment.

3) EVALUATION PROTOCOL

For better comparison, the metric of the area under the precision-recall curve (AUC) is reported here the same as decoder of the model RESCAL. There are some reasons for this point. Firstly, the decoder is RESCAL which implements the collective learning algorithm [8] by using ASALSAN [15]. The likelihoods of relation existence is produced by tensor operations and get the results as a whole. AUC is a reasonable metric [30], even though it can also be implemented by ranking entity score and selecting candidates for link prediction just like SimplE and TuckER doing. Secondly, Wang *et al.* [39] argue that the entity rank metrics is tend to overestimate the model performance and this point is also taken into account. AUC is a metric of general machine learning and used in numbers of KGC model evaluation. Finally, as a method of static relational learning, AUC is also a suitable measurement for it and the baseline published papers also report it, which is convenient for our comparison.

C. RESULTS AND ANALYSIS

1) EXPERIMENTAL RESULTS

To verify the improvement of HRESCAL2 and HRESCAL3 relative to baseline models and state-of-the-art. There are mainly four aspects data in four tables, respectively. All data are from the published papers except the new proposed models HRESCAL2 and HRESCAL3.

TABLE 5. AUC of link prediction of our model. The data is taken from the paper [8].

Model	Kinship	Nation	UMLS
CP [8]	0.9400	0.8300	0.9500
IRM [8]	0.6600	0.7500	0.7000
BCTF [8]	0.9000	N/A	0.9800
RESCAL [8]	0.9500	0.8400	0.9800
Linear+Reg [19]	0.9399	-	0.8822
Quad+Reg [19]	0.9389	-	0.8811
Linear+Constraint [19]	0.9287	-	0.8018
Quad+Constraint [19]	0.9384	-	0.9107
HRESCAL2(ours)	**0.9991**	**0.9782**	**0.9923**
HRESCAL3(ours)	**0.9986**	**0.9604**	**0.9997**
Improved2	5.25%	16.45%	1.26%
Improved3	5.12%	14.33%	2.01%

TABLE 6. AUC of link prediction and improvement for the datasets with different logic rule constraints.

Model	Countries		
	S1	S2	S3
HolE [23]	0.9970	0.7720	0.6970
ComplEx [25]	0.9977	0.9075	0.5474
NTPλ [37]	**1.0000**	0.9304	0.7726
MINERVA [40]	**1.0000**	0.9304	0.7726
NeuralLP [41]	**1.0000**	0.7510	0.9220
GNTP [42]	**1.0000**	0.9348	0.9127
HRESCAL2(ours)	**0.9990**	**0.9987**	**0.9982**
HRESCAL3(ours)	0.9963	**0.9940**	**0.9909**
Improved2	-0.10%	7.34%	8.57%
Improved3	-0.37%	6.84%	8.57%

In TABLE 5, the AUC metric is reported and they are compared with RESCAL and a newly published model extended from RESCAL. The results of model CP, IRM, BCTF and RESCAL are from the paper [8], Linear + Reg, Quad + Reg, Linear + Constraint and Quad + Constraint are from the recent paper [19].

In TABLE 6, the table records the results of the models HolE, ComplEx, NTPλ, MINERVA, NeuralLP and GNTP are from the original papers.

In TABLE 7, the results of RuleN and GNN are from their original paper. And the left records are from the paper [19] except the proposed model. In this TABLE, the results of HRESCAL2 is reported only due to the expensive computational cost. Besides, there is no paper reporting the metric of AUC on dataset `FB15k` found.

In TABLE 8, time cost is record comparison with RESCAL VS HRESCAL2. This report is for HRESCAL2 only, not HRESCAL3.

2) ANALYSIS OF PERFORMANCE

a: PERFORMANCE OF AUC

The TABLE 5, TABLE 6 and TABLE 7 report the three groups datasets in the metric of AUC, respectively.

TABLE 7. AUC of link prediction and improvement for the large-scale and challenging datasets.

Model	FB15K	FB15K237
RuleN [43]	-	0.9225
GNN [44]	-	0.9337
RESCAL [19]	-	**0.9761**
Non Neg RESCAL [19]	-	**0.9781**
TransE [19]	-	0.5084
DistMult [19]	-	0.7028
ComplEx [19]	-	0.6764
Linear+Reg [19]	-	0.9649
Quad+Reg [19]	-	0.9720
Linear+Constraint [19]	-	0.8000
Quad+Constraint [19]	-	0.9459
HRESCAL2(ours)	**0.9907**	0.9722
Improved2	-	-0.61%

In TABLE 5, the performance of the proposed models are not only better than CP, IRM, BCTF and RESCAL on all three datasets, but also a new reported result on all datasets except `Nations`. The new model achieve state-of-the-art performance. For `Nations`, there are 14 country groups as entities and 56 binary relations among them, achieving a 14.33% improvement and up to 96.04%. As for single relation KGC task, HRESCAL2 is up to 99.91% on dataset `Kinship` and 99.23% on `UMLS`, the improvement of the latter is just 1.26%. Despite the AUC of it is up to 99.23%, it is still worse than that of HRESCAL3 on this dataset. Surprisingly, it is different from the other two datasets.

In TABLE 6, for `Countries`, the proposed model is comparable with `Countries S1` and outperforming all other two with large margin. Especially, our model better than state-of-the-art with margin of 8.57% by both HRESCAL2 and HRESCAL3. In the process of Hamming distance encoding, only party of inverse relations to be handled correctly. That's why the proposed model can't get the AUC of 100% on the dataset `Countries S1`.

In TABLE 7, these are two large scale datasets and the result of HRESCAL2 is reported only because of the encoding results are out of the capacity of our equipment. For `FB15K`, the AUC score is up to 99.07% which is promising in machine learning, though there are no report on metric of AUC. For `FB15K237`, in this experiment, the AUC is also up to 97.22%, it is on-par the newly published paper.

b: FURTHER ANALYSIS

The improvements of the proposed model are due to the following factors. The first one is that HRESCAL can capture multihop relations latent features. This means that the multi-relations contained in the KG can also be mined and lead to a better performance. The adjacency tensor is only concerned with its one-hop neighbors and the multihop ones are ignored. Consequently, even if it is hard to find out how this information works to support the improvement of the performance, after being encoded by Hamming distance encoder, the vectors are shown to actually capture all entities from their reachable neighbors, implicitly. The second one is that HRESCAL captures partial inverse relation latent features. The expression of Hamming distance not only contains the relations in the KG but also captures the features of their inverse relations. This is another important reason for the improvement of performance relative to the baseline RESCAL. They are partly included in the entry after being encoded by Hamming distance. It is easy to understand this point from Eq. 2. To promote the persuasiveness, an experiment is conducted on the dataset `Countries`.[7]

c: PERFORMANCE OF TIME

In TABLE 8, the time cost is reduced between one tenth to one fifth. This result due to HRESCAL2 reduces the order from third to second. The factorization operation is significantly reduced. As a consequence, the time complexity is reduced from $O(N_e^3 N_r)$ to $O(N_e^3)$. This means that the time complexity is reduced N_r times to 1. From the perspective of the data in TABLE 8, the time consumed is greatly reduced.

TABLE 8. The time consumption results of HRESCAL2 model.

Dataset	Model	Running Time(unit:s)		
		Rank		
		10	20	40
Kinship	RESCAL	10.76	11.06	12.08
	HRESCAL2	2.38	2.45	2.51
Nations	RESCAL	27.39	27.33	27.83
	HRESCAL2	3.46	3.63	3.77
UMLS	RESCAL	28.40	29.15	29.96
	HRESCAL2	4.84	5.08	5.24

The results in TABLE 8 show how much time is reduced compared with the baseline model for different ranks of factor matrices. Note that this operation keep relation type features only when there does not exist a multi-relation and the relation type is of no concern. For example, in the `Nations` dataset, there are many multi-relations. Between Brazil and the USA, there are up to 26 relations, i.e. $Brazil \xrightarrow{treaties} USA$, $Brazil \xrightarrow{reltreaties} USA$, $\cdots$, $Brazil \xrightarrow{commonbloc2} USA$. According to our algorithm of the adjacency matrix, only the last relation $Brazil \xrightarrow{commonbloc2} USA$ is maintained, and the other 25 relations are lost. The information loss is more serious than that in the other two datasets. This is why the AUC is at most 97.82%, though it is much better than that of the baseline model. This means that whether order reduction should be performed is dataset-dependent.

VII. CONCLUSIONS AND FUTURE WORKS

In this section, some conclusions can be drawn from the experimental results in this section and research works worth doing are listed.

[7]The data of HolE are from the paper. The performance of ComplEx and NTPλ [37] are from the paper [37]. The performance of HRESCAL2 and HRESCAL3 are from the experiments.

A. CONCLUSIONS AND DISCUSSION

During constructing and evaluating the HRESCAL model, some basic conclusions is drawn on the basis of the experimental results in the context of KGC and a proof is presented in this paper. (1) Hamming distance encoder does capture partial inverse relation features of the relational data. (2) Hamming distance encoder can capture implicit multihop relation feature from KG. It is an open problem concerning the proof to the point, but a detailed explanation is given to insight into this point. Furthermore, it is built on the basis of the double meaning of entity-relation vector in the adjacency tensor (matrix). (3) Hamming distance based encoder captured latent features are beneficial for KGC tasks. Although it is hard to distinguish, Hamming distance indeed contribute to the multihop relation mining because the inverse relations are ticked out from the dataset `FB15k237` while the AUC is still improved with large margin. This is the evidence supporting (2) and (3) in the conclusions.

There are some meaningful problems worth discussing. (1) DistMult captures multihop by matrix multiplication, it is easy to be understood but hard to be extended to more hops relation inference. The proposed model performs multihop feature gathering by Hamming distance encoding, which is conducted implicitly. Nevertheless, it suffer from the limitation of interpretability. (2) When it captures the partial inverse relation features, it introduces some unavoidable noise. Hence, the model can never get the AUC up to 100%, the record of TABLE 7 illustrates this point. How to get rid of the noise remains a problem. (3) Although the Hamming distance based encoding is beneficial for the KGC tasks and handled before decoding, it is time-consuming to compute. Distributed computation may be is one of a reasonable resolutions. (4) Besides, the decoder can be one of any other factorization-based models such as TuckER, SimplE, etc. (5) Although our model HRESCAL is discussed for the KGC task only, it can also be applied to information retrieval, question answer, relation prediction, etc.

B. FUTURE WORKS

This work reveals the fact that Hamming distance is outstanding for the task of relational data mining and related tasks. Further research should be conducted to investigate the following problems. (1) Owing to a symmetric matrix obtained after Hamming distance encoding, the process of computation can be optimized according to this property. (2) Hamming distance encoding is memory exhausting, and a feasible solution may compute it slice-wisely to alleviate the stress of memory. It is also well coordinate with the collective learning of decoder. (3) As we know, HRESCAL2 and HRESCAL3 are not only used for KGC, but also other related applications such as information retrieval, question answer, etc. These potential applications are worth exploring. (4) For the implicit multihop relation mining by Hamming distance encoding, the mathematical basis is relatively weak, though it has a reasonable and detailed explanation. A tenable proof of this point may start a new line of KGC which remain open. It is promising and significant for KGC and related tasks. (5) The relations and attributes in the KG are not distinguished, the desired property that an attribute usually has no further relation with other nodes. That means the out-degree of the attribute node is 0, and this information can be used for further determining the likelihood of a relation existence right or wrong to further improve the accuracy of link prediction. (6) Now that every fact is an instance of some axioms, there are strong ties between logic rules and relation inference is a reasonable hypotheses. How to use the rule based information of it to improve the performance of KGC model is also a promising research direction.

REFERENCES

[1] Q. Song, Y. Wu, P. Lin, L. X. Dong, and H. Sun, "Mining summaries for knowledge graph search," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 10, pp. 1887–1900, Oct. 2018.
[2] S. Hu, L. Zou, J. X. Yu, H. Wang, and D. Zhao, "Answering natural language questions by subgraph matching over knowledge graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 5, pp. 824–837, May 2018.
[3] W. Zheng, H. Cheng, J. X. Yu, L. Zou, and K. Zhao, "Interactive natural language question answering over knowledge graphs," *Inf. Sci.*, vol. 481, pp. 141–159, May 2019.
[4] G. Wu, Y. He, and X. Hu, "Entity linking: An issue to extract corresponding entity with knowledge base," *IEEE Access*, vol. 6, pp. 6220–6231, 2018.
[5] B. Yang and M. T. Mitchell, "Leveraging knowledge bases in LSTMs for improving machine reading," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, R. Barzilay and M.-Y. Kan, Eds. Vancouver, BC, Canada: Association Computational Linguistics, Jul./Aug. 2017, pp. 1436–1446.
[6] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, "Knowledge base completion via search-based question answering," in *Proc. 23rd Int. Conf. World wide Web (WWW)*, Seoul, South Korea, Apr. 2014, pp. 515–526.
[7] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," in *Proc. 14th Int. Semantic Web Conf.*, Bethlehem, PA, USA, Oct. 2015, pp. 640–655.
[8] M. Nickel, V. Tresp, and H. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. 28th Int. Conf. Mach. Learn.*, L. Getoor and T. Scheffer, Eds. Bellevue, WA, USA: Omnipress, Jun./Jul. 2011, pp. 809–816.
[9] S. M. Kazemi and D. Poole, "Simple embedding for link prediction in knowledge graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4284–4295.
[10] V. Misra and S. Bhatia, "Bernoulli embeddings for graphs," in *Proc. 32nd AAAI Conf. Artif. Intell. 30th Innov. Appl. Artif. Intell. (IAAI), 8th AAAI Symp. Educ. Adv. Artif. Intell. (EAAI)*, S. A. McIlraith and K. Q. Weinberger, Eds. New Orleans, LA, USA: AAAI Press, Feb. 2018, pp. 3812–3819.
[11] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of 'Eckart-Young' Decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, Sep. 1970.
[12] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multi-modal factor analysis," in *Proc. UCLA Work. Papers Phonetics*, vol. 16, 1970, pp. 1–84. [Online]. Available: http://publish.uwo.ca/~harshman/wpppfac0.pdf
[13] R. A. Harshman, "Models for analysis of asymmetrical relationships among N objects or stimuli," in *Proc. 1st Joint Meeting Psychometric Soc. Soc. Math. Psychol.* Hamilton, ON, Canada: McMaster Univ., 1978. [Online]. Available: http://publish.uwo.ca/~harshman/asym1978.pdf
[14] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, Sep. 1966.
[15] B. Bader, R. A. Harshman, and T. G. Kolda, "Temporal analysis of semantic graphs using ASALSAN," in *Proc. 7th IEEE Int. Conf. Data Mining (ICDM)*. Omaha, NE, USA: IEEE Computer Society, Oct. 2007, pp. 33–42.

[16] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, Aug. 2009.

[17] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. 3rd Int. Conf. Learn. Represent.*, Y. Bengio and Y. LeCun, Eds. San Diego, CA, USA, May 2015.

[18] Y. Shen, P. Huang, M. Chang, and J. Gao, "Modeling large-scale structured relationships with shared memory for knowledge base completion," in *Proc. 2nd Workshop Represent. Learn. NLP (Rep4NLP ACL)*, P. Blunsom, A. Bordes, K. Cho, S. B. Cohen, C. Dyer, E. Grefenstette, K. M. Hermann, L. Rimell, J. Weston, and S. Yih, Eds. Vancouver, BC, Canada: Association Computational Linguistics, Aug. 2017, pp. 57–68.

[19] A. Padia, K. Kalpakis, F. Ferraro, and T. Finin, "Knowledge graph fact prediction via knowledge-enriched tensor factorization," *J. Web Semantics*, vol. 59, Dec. 2019, Art. no. 100497.

[20] C. Kemp, B. Joshua Tenenbaum, L. Thomas Griffiths, T. Yamada, and N. Ueda, "Learning systems of concepts with an infinite relational model," in *Proc. 21st Nat. Conf. Artif. Intell. 18th Innov. Appl. Artif. Intell. Conf.* Boston, MA, USA: AAAI Press, Jul. 2006, pp. 381–388.

[21] T. Franz, A. Schultz, S. Sizov, and S. Staab, "Triplerank: Ranking semantic Web data by tensor decomposition," in *Proc. Semantic Web, 8th Int. Semantic Web Conf.*, in Lecture Notes in Computer Science, A. Bernstein, D. R. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, Eds. Chantilly, VA, USA: Springer, Oct. 2009, pp. 213–228.

[22] I. Sutskever, R. Salakhutdinov, and B. Joshua Tenenbaum, "Modelling relational data using Bayesian clustered tensor factorization," in *Proc. Adv. Neural Inf. Process. Syst*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Vancouver, BC, Canada: Curran Associates, Dec. 2009, pp. 1821–1828.

[23] M. Nickel, L. Rosasco, and A. T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. 13th AAAI Conf. Artif. Intell.*, Phoenix, AZ, USA, Feb. 2016, pp. 1955–1961.

[24] H. Liu, Y. Wu, and Y. Yang, "Analogical inference for multi-relational embeddings," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, D. Precup and Y. W. Teh, Eds. Sydney, NSW, Australia: PMLR, Aug. 2017, pp. 2168–2178.

[25] T. Trouillon, C. R. Dance, É. Gaussier, J. Welbl, S. Riedel, and G. Bouchard, "Knowledge graph completion via complex tensor factorization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 4735–4772, 2017.

[26] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[27] I. Balazevic, C. Allen, and M. Timothy Hospedales, "Tucker: Tensor factorization for knowledge graph completion," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, K. Inui, J. Jiang, V. Ng, and X. Wan, Eds. Hong Kong: Association Computational Linguistics, Nov. 2019, pp. 5184–5193.

[28] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, Apr. 1950.

[29] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Computer Vision—ECCV 2008* (Lecture Notes in Computer Science), D. A. Forsyth, P. H. S. Torr, A. Zisserman, Eds. Marseille, France: Springer, Oct. 2008, pp. 304–317.

[30] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs," *Proc. IEEE*, vol. 104, no. 1, pp. 11–33, Jan. 2016.

[31] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds. Lake Tahoe, NV, USA: MIT Press, Dec. 2013, pp. 2787–2795.

[32] T. Mikolov, I. Sutskever, K. Chen, S. Gregory Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, Eds. Lake Tahoe, NV, USA: MIT Press, Dec. 2013, pp. 3111–3119.

[33] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, C. E. Brodley and P. Stone, Eds. Jul. 2014, Québec City, QC, Canada: AAAI Press, pp. 1112–1119.

[34] J. R. Rummel, "Dimensionality of nations project: Dyadic foreign conflict variables," *Int. J. Genomics*, pp. 1950–1965, Feb. 1992, doi: 10.3886/ICPSR05408.v1.

[35] A. T. McCray, "An upper-level ontology for the biomedical domain," *Int. J. Genomics*, vol. 4, no. 1, pp. 80–84, 2003.

[36] G. Bouchard, S. Singh, and T. Trouillon, "On approximate reasoning capabilities of low-rank vector spaces," in *Proc. AAAI Spring Symposia*. Palo Alto, CA, USA: AAAI Press, Mar. 2015.

[37] T. Rocktäschel and S. Riedel, "End-to-end differentiable proving," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds. Long Beach, CA, USA: MIT Press, Dec. 2017, pp. 3788–3800.

[38] K. Toutanova and D. Chen, "Observed versus latent features for knowledge base and text inference," in *Proc. 3rd Workshop Continuous Vector Space Models Compositionality*, 2015, pp. 57–66.

[39] Y. Wang, D. Ruffinelli, R. Gemulla, S. Broscheit, and C. Meilicke, "On evaluating embedding models for knowledge base completion," in *Proc. 4th Workshop Represent. Learn. NLP (RepL4NLP ACL)*, I. Augenstein, S. Gella, S. Ruder, K. Kann, B. Can, J. Welbl, A. Conneau, X. Ren, and M. Rei, Eds. Florence, Italy: Association Computational Linguistics, Aug. 2019, pp. 104–112.

[40] R. Das, S. Dhuliawala, M. Zaheer, L. Vilnis, I. Durugkar, A. Smola, A. Krishnamurthy, and A. McCallum, "Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning," in *Proc. 6th Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, Apr./May 2018.

[41] F. Yang, Z. Yang, and W. William Cohen, "Differentiable learning of logical rules for knowledge base reasoning," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, Eds. Long Beach, CA, USA: MIT Press, Dec. 2017, pp. 2319–2328.

[42] P. Minervini, M. Bošnjak, T. Rocktäschel, S. Riedel, and E. Grefenstette, "Differentiable reasoning on large knowledge bases and natural language," 2019, *arXiv:1912.10824*. [Online]. Available: http://arxiv.org/abs/1912.10824

[43] C. Meilicke, M. Fink, Y. Wang, D. Ruffinelli, R. Gemulla, and H. Stuckenschmidt, "Fine-grained evaluation of rule-and embedding-based systems for knowledge graph completion," in *The Semantic Web—ISWC 2018* (Lecture Notes in Computer Science), vol. 11136. Monterey, CA, USA: Springer, Oct. 2018, pp. 3–20.

[44] K. K. Teru, E. Denis, and W. L. Hamilton, "Inductive relation prediction by subgraph reasoning," 2019, *arXiv:1911.06962*. [Online]. Available: http://arxiv.org/abs/1911.06962

PANFENG CHEN received the B.S. degree in science from the China Three Gorges University, in 2007, and the M.S. degree in science from Central China Normal University, in 2011. He is currently pursuing the Ph.D. degree with Guizhou University, Guiyang, Guizhou, China. His main research interests include knowledge presentation and reasoning, especially knowledge graph related problems.

YISONG WANG received the B.S., M.S., and Ph.D. degrees from Guizhou University, in 1998, 2004, and 2007, respectively. He is currently a Professor with Guizhou University. His main research interests include knowledge representation and reasoning, nonmonotonic reasoning, and answer set programming in particular, inductive logic programming and their applications. He has been serving as an Associate Editor for the *Annals of Mathematics and Artificial Intelligence*, since 2018.

QUAN YU received the Ph.D. degree from Sun YAT-SEN University, in 2015. He is currently a Professor with Qiannan Normal University for Nationalities. His main research interests include knowledge representation and reasoning, and modal logic.

YI FAN received the Ph.D. degree from Griffith University, in 2017. He is currently an Associate Professor with Qiannan Normal University for Nationalities. His main research interests include knowledge representation, heuristic search, and image processing.

RENYAN FENG received the B.S. degree from the North China University of Technology, in July 2015, and the M.S. degree from Guizhou University, in July 2018, where she is currently pursuing the Ph.D. degree. Her main research interests include knowledge representation and reasoning, planning, and answer set programming.