Link Prediction on N-ary Relational Data

Saiping Guan
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
guansaiping@ict.ac.cn

Xiaolong Jin*
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
jinxiaolong@ict.ac.cn

Yuanzhuo Wang
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
wangyuanzhuo@ict.ac.cn

Xueqi Cheng
CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences; School of Computer and Control Engineering, University of Chinese Academy of Sciences
cxq@ict.ac.cn

ABSTRACT

With the overwhelming popularity of Knowledge Graphs (KGs), researchers have poured attention to link prediction to complete KGs for a long time. However, they mainly focus on promoting the performance on binary relational data, where facts are usually represented as triples in the form of (*head entity*, *relation*, *tail entity*). In practice, n-ary relational facts are also ubiquitous. When encountering such facts, existing studies usually decompose them into triples by introducing a multitude of auxiliary virtual entities and additional triples. These conversions result in the complexity of carrying out link prediction concerning more than two arities. It has even proven that they may cause loss of structural information. To overcome these problems, in this paper, without decomposition, we represent each n-ary relational fact as a set of its role-value pairs. We further propose a method to conduct Link Prediction on N-ary relational data, thus called NaLP, which explicitly models the relatedness of all the role-value pairs in the same n-ary relational fact. Experimental results validate the effectiveness and merits of the proposed NaLP method.

CCS CONCEPTS

• **Computing methodologies → Reasoning about belief and knowledge**.

KEYWORDS

Link prediction; n-ary relational fact; knowledge graph; neural network

ACM Reference Format:
Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2019. Link Prediction on N-ary Relational Data. In *Proceedings of the 2019 World Wide Web Conference (WWW '19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3308558.3313414

*Corresponding author: Xiaolong Jin (jinxiaolong@ict.ac.cn)

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313414

1 INTRODUCTION

Since Google announced Knowledge Graph (KG) in 2012[1], KG has been increasingly popular in these years. And link prediction on KG has caught much attention for a long time, to complete KG and further promote KG based applications. Usually, a KG is represented as a set of binary relational triples in the form of (*head entity*, *relation*, *tail entity*), where n-ary relational facts are decomposed into multiple triples via introducing virtual entities, such as the Compound Value Type (CVT) entities in Freebase [1]. Actually, n-ary relational facts are not in the minority, as an example, in Freebase, more than $\frac{1}{3}$ of its entities are involved in n-ary relational facts [31]. Based on these public KGs of binary form, researchers have developed numerous methods for link prediction [21, 29], which includes the tasks of head entity prediction, relation prediction and tail entity prediction. The popular tensor/matrix based methods [4, 17, 20, 23, 27, 28] and translation based ones [2, 11–13, 15, 16, 30, 32] have been developed and improved for so long. They view triples in a KG as entries in a tensor/matrix, or relational translations between head entities and tail entities. Relatively newly developed neural network based methods [3, 5, 7, 19, 25, 26] use neural network to learn the information propagation of the entities across the edges (relations) of a KG and integrate existing effective methods, or use neural network to model triples directly. Regardless of these various methods, link prediction is essentially reduced to scoring triples by designing different score functions and casted into a ranking/classification task, with valid triples preceding invalid ones in terms of their scores or having different labels from invalid ones.

However, manipulating n-ary relational facts into triples has some deficiencies. Firstly, it makes link prediction involving more than two arities need to consider many triples. Since existing link prediction methods on triples, like the above methods, usually model triples one by one individually and correspondingly conduct link prediction on only a single triple each time, it is intricate to

[1] https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

carry out link prediction involving more than one triple. Secondly, there is a loss of structural information in some of the conversions from n-ary relational facts to triples [31], which may lead to inaccurate link prediction. Thirdly, the added virtual entities along with the corresponding triples bring in more parameters to be learned.

With these considerations in mind, in this paper, each n-ary relational fact is represented as a set of its role-value pairs instead of being converted into multiple triples. Hence, no additional entities and data are introduced, and all the structural information is retained. Then, we propose a method to handle link prediction on n-ary relational data. Specifically, in this paper, link prediction on n-ary relational data is to predict the missing value or role of a relational fact.

Recently, there are also a few studies for link prediction on n-ary relational data directly. In m-TransH [31], a relation (binary or n-ary relation) is defined by the mappings from a sequence of roles corresponding to this type of relation, to their values. Each specific mapping is an instance of this relation. Then, a translation based method is proposed to model these instances. However, m-TransH does not take the relatedness of the components in the same n-ary relational fact into consideration. RAE [34] further improves m-TransH by complementally modeling the relatedness of values, i.e., the likelihood that two values co-participate in a common instance. Although RAE achieves favorable performance, it does not consider the roles explicitly when evaluating the above likelihood. Actually, under different sequences of roles (corresponding to different relations), the relatedness of two values is greatly different. For example, under the role sequence (*person*, *award*, *point in time*, *together with*) (in this paper, the examples including the role names are all derived from the raw data in Wikidata[2]), *Marie Curie* and *Henri Becquerel* are more related than under the role sequence (*person*, *spouse*, *start time*, *end time*, *place of marriage*), since they won *Nobel Prize in Physics* in *1903* together. To address these problems, our method explicitly models the relatedness of the role-value pairs involved in the same n-ary relational fact. The above example also elucidates its necessity.

In general, the main contributions of this paper can be summarized as follows:

- We advocate a representation form for n-ary relational facts, which represents each n-ary relational fact as a set of its role-value pairs.
- We propose a link prediction method NaLP on n-ary relational data, which captures the relatedness of the role-value pairs in the same n-ary relational fact explicitly.
- Experimental results and further analyses testify the effectiveness and superiority of the proposed NaLP method.

In addition, as publicly available n-ary relational datasets are limited, we build a practical one WikiPeople based on the raw data in Wikidata, which is available on github[3] for further research.

2 RELATED WORKS

According to the fact type that link prediction focuses on, link prediction and its methods can be divided into two categories, i.e., link prediction on binary relational data and link prediction on n-ary relational data.

2.1 Link Prediction on Binary Relational Data

The rapid development of KG has triggered the spring up of a great many methods, including tensor/matrix based methods, translation based methods, and neural network based ones, for link prediction on binary relational data.

Among tensor/matrix based methods [4, 17, 20, 23, 27, 28], the well-known one, RESCAL [23], views a KG as a three-way tensor, where each way corresponds to head entities, relations and tail entities, respectively. And the entry corresponds to the score indicating the validity of the triple formed by the values of the thee ways. If a triple (h, r, t) is observed in the KG, the entry is set to 1, otherwise, set to 0. Through minimizing the reconstruction error of the tensor, the embeddings of entities and relations are learned. Then, the reconstructed tensor from the learned embeddings is used to conduct link prediction, with triples corresponding to entries of high score as valid ones. Similarly, a matrix factorization based method, ComplEx [28], constructs a matrix for each relation, which is decomposed and optimized via minimizing the reconstruction error like RESCAL to learn the embeddings. Differently, complex values are used to define the embeddings, so as to deal with antisymmetric relations effectively.

Translation based methods are derived from the simple but effective method TransE [2]. Its popular translation assumption advocates that valid triples can be viewed as relational translation operations from head entities to tail entities. Based on this assumption, different types of translations and accordingly different score functions measuring the similarity between the relational translation results of head entities and the target tail entities are defined [2, 11–13, 15, 16, 30, 32]. The embeddings of entities and relations are then learned via minimizing score function based loss, which encourages valid triples to have much larger scores than invalid ones. Among these methods, the seminal one, TransE, performs translation operations in a shared space of entities and relations. Further, TransH [30] thinks translation operations should be conducted in relation-specific hyperplanes. Thus, it relates each relation with a hyperplane. Then, entities are projected into the hyperplanes of relations before translations.

Inspired by the excellent performance of neural network in various applications, it has been introduced into link prediction on binary relational data and achieved extremely promising results [3, 5, 7, 19, 25, 26]. Among them, the R-GCN method [25] introduces relational graph convolutional networks as its encoder to get the embeddings of entities, and uses the existing method [33] as its decoder to compute the scores of triples with the embeddings of entities from the encoder and the embeddings of relations randomly initialized. Differently, ConvE [3] models entity prediction directly, which reshapes and concatenates the embeddings of the known entity and relation to apply two-dimensional convolution on the concatenated matrix. Then, after a fully connected projection, it generates the prediction entity. By comparing it with the target entity, ConvE minimizes a binary cross-entropy loss. SENN [7] integrates the prediction tasks of head entities, relations and tail entities in a unified neural network framework based on

[2]https://www.wikidata.org/wiki/Wikidata:Main_Page
[3]https://github.com/gsp2014/WikiPeople

shared embeddings. Fully connected layers are applied to model the prediction processes. ConvKB [19] represents each triple as a three-column matrix, where each column vector corresponds to an element of the triple. This matrix is fed into a convolution layer to generate different feature maps, which are then concatenated into a single feature vector representing the triple. After passing it to a fully connected layer, ConvKB obtains a score indicating the validity of the triple.

2.2 Link Prediction on N-ary Relational Data

As n-ary relational facts are complicated and usually difficult to deal with, most of existing studies convert them into multiple triples, with the introduction of virtual entities and additional triples [1], and research works for link prediction on n-ary relational data directly are relatively much scarcer and newer.

Based on the link prediction method TransH [30] on binary relational data, m-TransH [31] is proposed to generalize it to n-ary relational data. The m-TransH method presents a mathematical definition of relations (including binary and n-ary relations) as the mappings from sequences of roles to their values. Each mapping is an instance of the corresponding relation. Then, the score function of an instance is defined by the weighted sum of the projection results from its values to its relation hyperplane, where the weights are the real numbers projected from its roles. RAE [34] further considers the relatedness of values, i.e., the likelihood that two values co-participate in a common instance, and adds this relatedness loss with a weighted hyper-parameter to the embedding loss of m-TransH. Although with the additional modeling of the relatedness of values, RAE outperforms m-TransH, it does not look into the roles when computing the relatedness. Since roles are also a fundamental part of instances, taking them into consideration may make a difference as illustrated afore.

3 THE NALP METHOD

In this paper, link prediction on n-ary relational data is transformed into estimating whether a relational fact is valid and then casted into a classification task.

In this section, we first demonstrate our representation form of n-ary relational facts, then outline the framework of the proposed NaLP method and elaborate its two key components, respectively. Subsequently, some characteristics of NaLP are discussed. And the details of model training are introduced in the following part.

3.1 The Representation of N-ary Relational Facts

In this paper, we represent each n-ary relational fact as a set of its role-value pairs. As an example, the representation of the fact that *Marie Curie* received *Willard Gibbs Award*, is as follows:

$$\{person : Marie\ Curie, award : Willard\ Gibbs\ Award\}.$$

It is an n-ary relational fact of arity 2, i.e., a binary relational fact, as it has two role-value pairs.

And the facts where some roles have more than one value are represented similarly. For example, the representation of the fact that *Marie Curie* received *Nobel Prize in Physics* in *1903* together with *Henri Becquerel* and *Pierre Curie*, is as follows:

$$\{person : Marie\ Curie, award : Nobel\ Prize\ in\ Physics, point\ in\ time : 1903, together\ with : Henri\ Becquerel, together\ with : Pierre\ Curie\}.$$

It is an n-ary relational fact of arity 5, with 5 role-value pairs.

Formally, given an n-ary relational fact with n roles, and each role r_i having m_i values ($i = 1, 2, \ldots, n$), the representation of this n-ary relational fact is the following set of role-value pairs:

$$\{r_1 : v_1, \ldots, r_1 : v_{m_1}, r_2 : v_{m_1+1}, \ldots, r_2 : v_{m_1+m_2}, \ldots, r_n : v_{m_1+m_2+\cdots+m_{n-1}+1}, \ldots, r_n : v_{m_1+m_2+\cdots+m_n}\}.$$

It is an n-ary relational fact of arity $(m_1 + m_2 + \cdots + m_n)$.

3.2 The Framework of NaLP

Then, we propose a link prediction method NaLP on n-ary relational data. It is designed based on two considerations. On one hand, a role and its value are tightly linked to each other, thus should be bound together. On the other hand, as mentioned in the beginning of Section 3, we aim to judge the validity of a relational fact, i.e., a set of role-value pairs. That is, for a set of role-value pairs, we need to determine whether it is able to form a valid relational fact. Intuitively, all the role-value pairs of a valid relational fact are closely related. And if all the role-value pairs from a set are closely related, then, we assume that this set of role-value pairs has high probability to constitute a valid relational fact. With these considerations in mind, the framework of NaLP is designed as illustrated in Figure 1, which consists of two key components, role-value pair embedding and relatedness evaluation. For clarity, in the figure, we only exemplify one n-ary relational fact in the following form:

$$Rel = \{r_1 : v_1, r_2 : v_2, \ldots, r_m : v_m\},$$

where m is the arity of the relational fact *Rel*. The role set of *Rel* is denoted as $R_{Rel} = \{r_1, r_2, \ldots, r_m\}$ and r_i may be the same to r_j ($i, j = 1, 2, \ldots, m, i \neq j$), as demonstrated in Section 3.1; the value set of *Rel* is denoted as $V_{Rel} = \{v_1, v_2, \ldots, v_m\}$. In what follows, as an example, we illustrate the learning details of this n-ary relational fact.

For the given relational fact *Rel*, the embeddings of its roles in R_{Rel} and values in V_{Rel} are looked up from the embedding matrices $\mathbf{M_R} \in \mathbb{R}^{|R|\times k}$ of roles and $\mathbf{M_V} \in \mathbb{R}^{|V|\times k}$ of values, respectively, where R is the set of all the roles on dataset; V is the set of all the values; k is the latent dimension of the vector space. Following the convention, in what follows, the embeddings are denoted with the same letters but in boldface. As depicted in Figure 1, after passing these embeddings to the role-value pair embedding component, we get the embedding matrix of the m role-value pairs, i.e., *Rel* (see Section 3.3). This resulting embedding matrix is fed into the relatedness evaluation component to compute the pair-wise relatedness of all the role-value pairs, then estimate their overall relatedness, which is used to obtain the evaluation score (see Section 3.4). Particularly, this framework equips NaLP with the ability to be permutation-invariant to the input order of the role-value pairs and deal with relational facts of different arities (see Section 3.5). Subsequently, the evaluation score is used to generate loss (see Section 3.6).

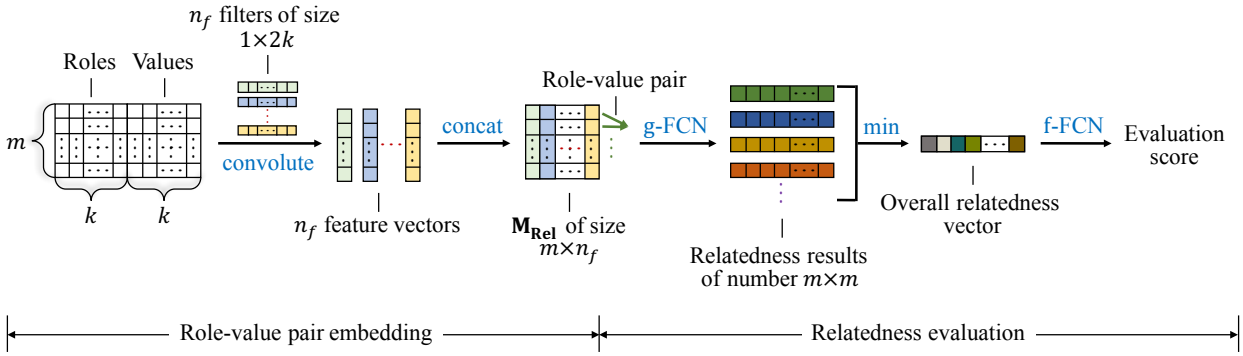

Figure 1: The framework of the proposed NaLP method.

3.3 Role-Value Pair Embedding

The role-value pair embedding component aims to obtain the embeddings of the input role-value pairs. This process contains two sub-processes, i.e., convolution and concatenation, corresponding to "convolute" and "concat" in Figure 1, respectively.

3.3.1 "convolute". As presented in Figure 1, before convolution, the embeddings of the roles r_i and the corresponding values v_i ($i = 1, 2, \ldots, m$) in the relational fact *Rel* are concatenated, and form a matrix of dimension $m \times 2k$, where each role-value pair makes up a row.

Then, this concatenated matrix is passed to the convolution with filter set Ω of n_f filters. To capture the features of the role-value pairs, the dimensions of the filters are all set to $1 \times 2k$. After convolution, we get n_f results of dimension m. And Rectified Liner Units (ReLU) [18] is applied to these results to get n_f feature vectors.

3.3.2 "concat". We concatenate the n_f feature vectors to form a matrix of dimension $m \times n_f$. Then, this matrix can be treated as the embeddings of the m role-value pairs, and each row corresponds to the embedding of one role-value pair. Specifically, each entry of a row encodes the feature of this dimension.

Formally, the embedding matrix $\mathbf{M_{Rel}} \in \mathbb{R}^{m\times n_f}$ of the role-value pairs, i.e., the relational fact *Rel* is defined as:

$$\mathbf{M_{Rel}} = \text{concat}\left(\Upsilon(\text{concat}(\mathbf{R_{Rel}}, \mathbf{V_{Rel}}) * \Omega)\right), \tag{1}$$

where Υ is the ReLU function [18], i.e., $\Upsilon(x) = \max(0, x)$; $*$ denotes the convolution operator; $\text{concat}(\mathbf{R_{Rel}}, \mathbf{V_{Rel}})$ is the aforementioned embedding concatenation process of the role-value pairs before convolution.

3.4 Relatedness Evaluation

This component computes the pair-wise relatedness of all the input role-value pairs, estimates their overall relatedness and obtains the evaluation score, corresponding to "g-FCN", "min" and "f-FCN" in Figure 1, respectively. Before introducing these sub-processes, let's see the principle behind this component.

3.4.1 The Principle. The principle of its design goes deep into determining whether a set of role-value pairs is able to form a valid relational fact. As demonstrated in Section 3.2, this is reduced to computing the overall relatedness of all the role-value pairs in the set. However, the overall relatedness of a set with more than two objects is intricate to measure. When the number of objects is large, corresponding to relational fact of high arity, the evaluation is further complicated. Whereas, computing the relatedness between two objects is relatively much simpler. Thus, how to determine the tricky overall relatedness via the simple relatedness between the role-value pairs?

Straightforwardly, if all the role-value pairs form a closely related set, i.e., a valid relational fact, then every two pairs from the set are greatly related. Thus, for any two pairs, the values of their relatedness feature vector, measuring the relatedness in many different views, are all expected to be sufficiently large. That is, for each feature dimension, the minimum over this dimension of every two pairs is not allowed to be too small. Hence, based on this observation, we apply element-wise minimizing over the pair-wise relatedness of all the role-value pairs to approximately evaluate their overall relatedness. Accordingly, the relatedness evaluation component is designed to contain the computation of the relatedness between the role-value pairs and the overall relatedness of all the role-value pairs, before estimating the final evaluation score.

3.4.2 "g-FCN". The relatedness between the role-value pairs is captured via a Fully Connected Network (FCN) with ReLU as its activation function. In particular, the widely used FCN is also adopted to infer the ways in which two objects are related and achieves marvelous performance in computer vision [8, 24]. In this paper, it turns out that a one-layer FCN already works very well. Thus, only a fully connected layer is adopted in this sub-process.

Concretely, the embeddings of any two role-value pairs are concatenated to form a vector of dimension $2n_f$, and then fed into the fully connected layer with n_{gFCN} nodes. The resulting vector of dimension n_{gFCN} is the relatedness feature vector with entries greater than or equal to 0, each dimension of which indicates the relatedness of the input two role-value pairs in some respect.

3.4.3 "min". As discussed in Section 3.4.1, the overall relatedness of all the role-value pairs is simply approximated by element-wise

minimizing over their pair-wise relatedness. Thus, the overall relatedness $\mathcal{R}_{\mathbf{Rel}}$ of Rel is defined as:

$$\begin{aligned}\mathcal{R}_{\mathbf{Rel}} &= \min_{i,j=1}^{m}\Big(\text{g-FCN}\left(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_i, [\mathbf{M}_{\mathbf{Rel}}]_j)\right)\Big) \\ &= \min_{i,j=1}^{m}\Upsilon\big(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_i, [\mathbf{M}_{\mathbf{Rel}}]_j)\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\big),\end{aligned} \tag{2}$$

where $\min(\cdot)$ is the element-wise minimizing function; $[\mathbf{x}]_i$ is the i-th entry of $\mathbf{x}$; $\mathbf{W}_{\mathbf{gFCN}}$ of dimension $2n_f \times n_{\text{gFCN}}$ and $\mathbf{b}_{\mathbf{gFCN}}$ of dimension n_{gFCN} are the weight matrix and bias vector of g-FCN, respectively. The resulting $\mathcal{R}_{\mathbf{Rel}}$ is a vector of dimension n_{gFCN}, with each entry implying the degree of the overall relatedness in terms of certain feature corresponding to that dimension.

3.4.4 "f-FCN". Then, $\mathcal{R}_{\mathbf{Rel}}$ is used to generate the evaluation score. A one-layer FCN is also found to be sufficient and applied in this sub-process. In this way, the evaluation score $s(Rel)$ of Rel is defined as:

$$\begin{aligned}s(Rel) &= \text{f-FCN}(\mathcal{R}_{\mathbf{Rel}}) \\ &= \mathcal{R}_{\mathbf{Rel}}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}},\end{aligned} \tag{3}$$

where $\mathbf{W}_{\mathbf{fFCN}}$ of dimension $n_{\text{gFCN}} \times 1$ and b_{fFCN} are the weight matrix and bias variable of f-FCN, respectively.

3.5 Look into NaLP

Note that, in m-TransH [31] and RAE [34], all relations along with their role sequences are predefined, and the inputs of the models are the instances of these relations, i.e., the sequences of ordered values. Since the i-th value of an input instance corresponds to the i-th role of its relation, the order of the input values are usually not allowed to be changed randomly. Whereas, each input of the proposed NaLP method is a set of role-value pairs, corresponding to a relational fact. Thus, the order of the objects in each input matters in m-TransH and RAE, while it has no impact on NaLP. Actually, NaLP is permutation-invariant to the input order of the role-value pairs and is able to cope with relational facts of different arities. Let's look into these two characteristics.

3.5.1 A Simple Explanation of Permutation Invariance. Suppose that the role-value pairs of Rel are input into NaLP in the original order with the indexes from 1 to m, and a permutation ρ to the indexes results in a new order with the indexes from $\rho(1)$ to $\rho(m)$.

On these grounds, we simply explain the permutation invariance as follows:

- In the process of role-value pair embedding, the embedding matrix $\mathbf{M}_{\rho(\mathbf{Rel})}$ of the permutated role-value pairs $\rho(Rel)$ is the permutation of the original $\mathbf{M}_{\mathbf{Rel}}$, with the $\rho(i)$-th row of $\mathbf{M}_{\mathbf{Rel}}$ placed in the i-th row of $\mathbf{M}_{\rho(\mathbf{Rel})}$, i.e.,

$$[\mathbf{M}_{\rho(\mathbf{Rel})}]_i = [\mathbf{M}_{\mathbf{Rel}}]_{\rho(i)}. \tag{4}$$

- In the process of relatedness evaluation, the overall relatedness $\mathcal{R}_{\rho(\mathbf{Rel})}$ of $\rho(Rel)$ is as follows:

$$\begin{aligned}&\mathcal{R}_{\rho(\mathbf{Rel})} \\ &= \min_{i,j=1}^{m}\Big(\text{g-FCN}\left(\text{concat}([\mathbf{M}_{\rho(\mathbf{Rel})}]_i, [\mathbf{M}_{\rho(\mathbf{Rel})}]_j)\right)\Big) \\ &= \min_{i,j=1}^{m}\Upsilon\left(\text{concat}([\mathbf{M}_{\rho(\mathbf{Rel})}]_i, [\mathbf{M}_{\rho(\mathbf{Rel})}]_j)\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\right) \\ &= \min_{i,j=1}^{m}\Upsilon\left(\text{concat}([\mathbf{M}_{\mathbf{Rel}}]_{\rho(i)}, [\mathbf{M}_{\mathbf{Rel}}]_{\rho(j)})\mathbf{W}_{\mathbf{gFCN}} + \mathbf{b}_{\mathbf{gFCN}}\right).\end{aligned} \tag{5}$$

That is, we obtain $m \times m$ relatedness results, the same to those in the original version, but with the order of the results changed. Since $\min(\cdot)$ is permutation-equivariant, we have:

$$\mathcal{R}_{\rho(\mathbf{Rel})} = \mathcal{R}_{\mathbf{Rel}}. \tag{6}$$

- Then, the evaluation score $s(\rho(Rel))$ of $\rho(Rel)$ is as follows:

$$\begin{aligned}s(\rho(Rel)) &= \text{f-FCN}(\mathcal{R}_{\rho(\mathbf{Rel})}) \\ &= \mathcal{R}_{\rho(\mathbf{Rel})}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}} \\ &= \mathcal{R}_{\mathbf{Rel}}\mathbf{W}_{\mathbf{fFCN}} + b_{\text{fFCN}} \\ &= s(Rel).\end{aligned} \tag{7}$$

- Therefore, NaLP is permutation-invariant to the input order of the role-value pairs.

3.5.2 Handling Relational Facts of Different Arities. Suppose that two batches of relational facts of arities m and m', respectively, are input into NaLP subsequently. In the process of role-value pair embedding, they result in the embedding matrices of the role-value pairs with m and m' rows, respectively. In the process of relatedness evaluation, correspondingly, we obtain $m \times m$ and $m' \times m'$ relatedness results. As $\min(\cdot)$ in Equation (2) returns one vector among all these results of number $m \times m$ or $m' \times m'$, the number of results makes no difference. Thus, NaLP is able to deal with relational facts of different arities.

3.6 Model Training

In the following, we present the loss function and the training process of the proposed NaLP method in detail.

3.6.1 The Loss Function. As above described, we obtain the evaluation score $s(Rel)$ of Rel. Following ComplEx [28], the loss of Rel is defined as:

$$\mathcal{L}(Rel) = \log\left(1 + e^{-I_{Rel}s(Rel)}\right), \tag{8}$$

where $I_{Rel} = 1$, if Rel is a valid relational fact, and $I_{Rel} = -1$, otherwise.

It is straightforward to optimize NaLP with the standard back-propagation. The popular stochastic optimization method Adam [14] with hyper-parameter learning rate λ is used as the optimizer in this paper.

3.6.2 The Training Process. Algorithm 1 presents the training process of the NaLP method.

Before training, the training set is reorganized into several training groups, each of which keeps the relational facts of the same arity (Line 1). In order to guarantee that NaLP masters basic relational facts of low arities before learning relational facts of high arities, we sort the above resulting groups by their arities in ascending order (Line 2). Similar operations are applied to train model on the path queries of length 1 and then all the path queries [9]. Subsequently, the embedding matrices $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$ are randomly initialized from uniform distribution, respectively, with $-\frac{1}{\sqrt{k}}$ as minimum and $\frac{1}{\sqrt{k}}$ as maximum; truncated normal distribution with 0.0 as mean and 0.1 as standard deviation is adopted as initializer to initialize all the filters in Ω [19]; $\mathbf{W}_{\mathbf{gFCN}}$ and $\mathbf{W}_{\mathbf{fFCN}}$ are randomly initialized via xavier initializer [6], respectively; $\mathbf{b}_{\mathbf{gFCN}}$ and b_{fFCN} are randomly initialized with zeros (Line 3).

Algorithm 1 The training process of the NaLP method.

Input: Training set T, role set R, value set V, max number of epochs *nepoch*, embedding dimension k, batch size β, number of filters n_f in Ω, the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$.
Output: $\mathbf{W}_{\mathbf{R}}$ and $\mathbf{W}_{\mathbf{V}}$, as well as the parameters of NaLP.

1: $T \leftarrow$ group T by the arities of the facts;
2: $T \leftarrow$ sort the groups in T by their arities in ascending order;
3: **Initialize** $\mathbf{M}_{\mathbf{R}}, \mathbf{M}_{\mathbf{V}} \leftarrow \text{uniform}(-\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{k}})$;
$\Omega \leftarrow \text{truncated\_normal}(0.0, 0.1)$;
$\mathbf{W}_{\mathbf{gFCN}}, \mathbf{W}_{\mathbf{fFCN}} \leftarrow \text{xavier\_initializer}()$;
$\mathbf{b}_{\mathbf{gFCN}}, b_{\text{fFCN}} \leftarrow \text{zeros\_initializer}()$;
4: **repeat**
5: **for** each group $T' \in T$ **do**
6: **for** $j = 1$ to $\left\lceil \frac{|T'|}{\beta} \right\rceil$ **do**
7: $S^+ \leftarrow \text{sample}(T', \beta)$; //sample β training facts from T'
8: $S^- \leftarrow \text{neg\_sample}(S^+, \beta)$; //obtain β negative samples
9: $S \leftarrow S^+ \cup S^-$;
10: $loss \leftarrow 0$;
11: **for** $\forall Rel \in S$ **do**
12: $\mathbf{R}_{\mathbf{Rel}} \leftarrow \text{look-up}(\mathbf{M}_{\mathbf{R}}, R_{Rel})$; //look up R_{Rel}
13: $\mathbf{V}_{\mathbf{Rel}} \leftarrow \text{look-up}(\mathbf{M}_{\mathbf{V}}, V_{Rel})$; //look up V_{Rel}
14: $\mathbf{M}_{\mathbf{Rel}} \leftarrow$ get the embedding matrix of the role-value pairs following Equation (1);
15: $\mathcal{R}_{\mathbf{Rel}} \leftarrow$ compute the overll relatedness of *Rel* following Equation (2);
16: $s(Rel) \leftarrow$ generate the evaluation score following Equation (3);
17: $\mathcal{L}(Rel) \leftarrow$ compute the loss of *Rel* following Equation (8);
18: $loss \leftarrow loss + \mathcal{L}(Rel)$;
19: **end for**
20: Update the embeddings of the roles and values in S (i.e., the related rows of $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$), the filters in Ω, $\mathbf{W}_{\mathbf{gFCN}}$, $\mathbf{W}_{\mathbf{fFCN}}$, $\mathbf{b}_{\mathbf{gFCN}}$, as well as b_{fFCN} via $\nabla loss$;
21: **end for**
22: **end for**
23: **until** the evaluation result on the validation set drops continuously or this process has been iterated for *nepoch* times

During training, Lines 5-22 are repeated until the evaluation result on the validation set drops continuously or reaching the max number of epochs *nepoch*. In this paper, *nepoch* is set to 5000. In each epoch, $\left\lceil \frac{|T'|}{\beta} \right\rceil$ batches of training facts are sampled from each training group T', where $\lceil \cdot \rceil$ is the ceiling function. For each selected training fact, similar to the negative sampling method adopted in TransE [2], we randomly replace one of its values with a random value that the corresponding role holds, or one of its roles is replaced with a relatively lower probability, to obtain a negative sample not contained in the dataset (Line 8). Thereafter, the selected training facts and their negative samples are used to train the model. For each such relational fact *Rel*, the embeddings of its roles and values are looked up from $\mathbf{M}_{\mathbf{R}}$ and $\mathbf{M}_{\mathbf{V}}$, respectively (Lines 12 and 13). Then, they are fed into the role-value pair embedding component to obtain the embedding matrix of the role-value pairs (Line 14). After passing this embedding matrix to the relatedness evaluation component, we obtain the overall relatedness $\mathcal{R}_{\mathbf{Rel}}$ and the evaluation score $s(Rel)$ of *Rel* (Lines 15 and 16). Subsequently, based on $s(Rel)$, loss is computed and added to the loss of the current batch (Lines 17 and 18). Finally, parameter update is performed in batch mode (Line 20).

Specifically, batch normalization [10] is applied after convolution in the role-value pair embedding component to stabilize and accelerate the training process.

4 EXPERIMENTS

In this section, we evaluate the proposed NaLP method through link prediction. Then, overall relatedness analysis is conducted to have a better understanding of NaLP. And some important hyper-parameters are analyzed their impacts on the robustness of NaLP. Before elaborating these experiments, we will first introduce the datasets and metrics, as well as the baselines and experimental settings.

4.1 Datasets and Metrics

4.1.1 Datasets. We conduct our experiments on two datasets. The first one is the public n-ary relational dataset JF17K, which is derived from the popular KG Freebase [1]. All the facts in it are in good quality with the role sequences equal to the predefined standard ones and the corresponding values existing and known. This is usually not the case. And we build a relatively more practical dataset WikiPeople, which is also our second experimental dataset. It is constructed as follows:

- We download the Wikidata dump[4] and extract the facts concerning entities of type *human*.
- Then, these facts are further denoised. For example, facts containing element related to image are filtered out, and facts containing element in {*unknown value*, *no values*} are removed.
- Subsequently, we select the subsets of elements which have at least 30 mentions. The facts related to these elements are kept. Further, each fact is parsed into a set of its role-value pairs.
- The remaining facts are randomly split into training set, validation set and test set by a proportion of 80%:10%:10%.

Actually, WikiPeople is relatively more flexible, where data incompleteness, insert and update are universal. As an example, the role sequence of some n-ary relational facts like "*Marie Curie* received *Nobel Prize in Chemistry* in *1911*" is [*person*, *award*, *point in time*]. Then, the following similar facts in Wikidata[5] correspond to the three cases of data incompleteness, insert and update, respectively:

- *Marie Curie* received *Willard Gibbs Award*.
- *Marie Curie* received *Matteucci Medal* in *1904* with *Pierre Curie*.
  Marie Curie received *Nobel Prize in Physics* in *1903* together with *Henri Becquerel* and *Pierre Curie*.
- *Marie Curie* received *Davy Medal* with *Pierre Curie*.

[4] https://archive.org/details/wikibase-wikidatawiki-20171120
[5] https://www.wikidata.org/wiki/Q7186

In the first example, the value of the role *point in time* is unknown. In the second examples, one or two new roles *together with* are in need. In the third one, the value of the role *point in time* is unknown and a new role *together with* appears. Fortunately, our representation form of relational facts, which represents each relational fact as a set of its role-value pairs, is able to cope with all these cases, as exemplified in Section 3.1.

In real scenario, since there is possible missing in the process of knowledge extraction, and knowledge also grows and updates rapidly with many new contents flowing in, in the dataset derived from one snapshot, the above phenomena of data incompleteness, insert and update are inevitable. Thus, the developed WikiPeople dataset is practical. Note that, in this paper, no special handling is done to values from continuous domain (e.g., point in time, date of birth, date of death, etc.), and they are tackled like other values from discrete domain.

The detailed statistics of the two experimental datasets are displayed in Table 1, where $\#Relation$, $\#Train$, $\#Valid$ and $\#Test$ are the sizes of the relation set, the training set, the validation set and the test set, respectively. Here, $\#Relation$ is counted according to RAE [34] to have a better understanding of the two datasets.

4.1.2 Metrics. We adopt the standard Mean Reciprocal Rank (MRR) and Hits@N as metrics for comparison. These metrics are computed in the similar way of binary dataset [2]. For each test fact, one of its values/roles is removed and replaced by all the values/roles in V/R. These corrupted facts are fed into NaLP to obtain the evaluation scores. Then, these corrupted facts are sorted according to their evaluation scores in descending order. The rank of the test fact is finally stored. This whole procedure is repeated for all the other values/roles of the test fact. Thus, MRR is the average of these reciprocal ranks and Hits@N is the proportion of ranks which are less than or equal to N. In this paper, $N \in \{1, 3, 10\}$, i.e., the results in terms of Hits@1, Hits@3 and Hits@10 are reported. The traditional mean rank (the average of these ranks) is not adopted as a metric, since it is sensitive to outliers [22]. For these chosen metrics, the higher the value of MRR/Hits@N, the better the performance of the prediction.

In the test process, except the test fact, other corrupted facts that may also be valid, i.e., exist in the training/validation/test set, are discarded before sorting the facts.

4.2 Baselines and Experimental Settings

4.2.1 Baselines. Prior works for link prediction on n-ary relational data directly are scarce, and the state-of-the-art works are m-TransH [31] and its enhanced version RAE [34]. Since RAE performs better than its simplified version m-TransH, we only compare the proposed NaLP method with RAE.

4.2.2 Experimental Settings. For the proposed NaLP method, the reported results are given for the best set of hyper-parameters after grid search on the following values, by reference to ConvKB [19]: The embedding dimension $k \in \{50, 100\}$, the batch size $\beta \in \{128, 256\}$, the learning rate $\lambda \in \{5e^{-6}, 1e^{-5}, 5e^{-5}, 1e^{-4}, 5e^{-4}, 1e^{-3}\}$, the number of filters n_f of the convolution in {50, 100, 200, 400, 500}, the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ in {50, 100, 200, 400, 500, 800, 1000, 1200}. The finally adopted optimal settings are: $k = 100$, $\beta = 128$, $\lambda = 5e^{-5}$, $n_f = 200$, $n_{\text{gFCN}} = 1000$ for JF17K, and $k = 100$, $\beta = 128$, $\lambda = 5e^{-5}$, $n_f = 200$, $n_{\text{gFCN}} = 1200$ for WikiPeople.

For the RAE method, as it does not report the ranges of hyper-parameters, we use the following ranges, by reference to its best settings on JF17K: The embedding dimension $k \in \{50, 100\}$, the margin ϵ of the first update step in $\{0.1, 0.5, 1.0, 2.0\}$, its weight hyper-parameter w of the regularization term and its threshold hyper-parameter δ in $\{1e^{-3}, 0.01, 0.1\}$, the batch sizes of the three update steps $\beta_1, \beta_2 = \beta_3 \in \{128, 256, 512, 1000\}$ (note that the second and third update steps share the same batch size), their learning rates $\lambda_1, \lambda_2, \lambda_3 \in \{5e^{-5}, 5e^{-4}, 5e^{-3}, 1e^{-3}, 1.5e^{-3}\}$. The finally adopted optimal settings are: $k = 100$, $w = 0.1$, $\delta = 1e^{-3}$, $\epsilon = 0.5$, $\beta_1 = 512$, $\beta_2 = \beta_3 = 128$, $\lambda_1 = 5e^{-3}$, $\lambda_2 = 1.5e^{-3}$, $\lambda_3 = 5e^{-4}$ for WikiPeople.

Note that, on WikiPeople, the best set of hyper-parameters are determined according to the evaluation results on the validation set in terms of MRR. And, on JF17K, since it lacks a validation set, we tune the hyper-parameters on the test set.

Particularly, we reproduce the experimental results reported in RAE on JF17K and compute more detailed results in terms of the above fine-grained metrics. Note that, before running NaLP on JF17K and RAE on WikiPeople, we need to transform the representation form of facts. It is effortless to transform each value sequence in JF17K to a set of role-value pairs, by adding the corresponding roles. To conduct the inverse transformation on WikiPeople, relations along with their role sequences are defined in advance, then only the value sequence of each fact is stored. Specifically, a new relation is defined when the aforementioned case of data incompleteness/insert/update appears.

4.3 Link Prediction

Link prediction on n-ary relational data has two tasks, i.e., value prediction and role prediction.

4.3.1 Value Prediction. In order to illustrate the performance of the proposed NaLP method on value prediction, we report the experimental results compared with RAE in terms of all the fine-grained metrics in Table 2.

From the results, it is clear that on both datasets, NaLP consistently outperforms RAE in terms of all the four metrics, which indicates the effectiveness and superiority of NaLP. Specifically, on the dataset in good quality, i.e., JF17K, NaLP improves the performance by 0.056 on MRR, 7.1% on Hits@1 and 5.7% on Hits@3. It testifies the strength of NaLP considering the relatedness of role-value pairs rather than only the relatedness of values in RAE. On the relatively more practical dataset, i.e., WikiPeople, the superiority of NaLP is significant. NaLP has considerable increase on all the four metrics, especially 0.166 on MRR, 17.0% on Hits@1 and 18.2% on Hits@3. It is not surprising, since NaLP is capable of better coping with diverse data than RAE. Data incompleteness/insert/update, which is ubiquitous on WikiPeople, is handled elegantly in NaLP, while new relation is defined in RAE, which may lead to data sparsity, when the instances of the new relation are in small number. Hence, it results in the relatively much worse performance of RAE on WikiPeople. Moreover, the overall better results of NaLP regarding more fine-grained metrics, i.e., MRR, Hits@1 and Hits@3,

Table 1: The statistics of the two datasets, JF17K and WikiPeople.

Dataset	$\|V\|$	$\#Relation$			$\#Train$			$\#Valid$			$\#Test$		
		Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall
JF17K	28,645	186	136	322	44,210	32,169	76,379	-	-	-	10,417	14,151	24,568
WikiPeople	47,765	150	557	707	270,179	35,546	305,725	33,845	4,378	38,223	33,890	4,391	38,281

Table 2: Experimental results of value prediction in terms of MRR and Hits@{1, 3, 10}.

Method	JF17K				WikiPeople			
	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10
RAE	0.310	0.219	0.334	0.504	0.172	0.102	0.182	0.320
NaLP	**0.366**	**0.290**	**0.391**	**0.516**	**0.338**	**0.272**	**0.364**	**0.466**

Table 3: Detailed experimental results of value prediction on binary and n-ary relational facts in terms of MRR and Hits@{1, 3, 10}.

Method	JF17K								WikiPeople							
	MRR		Hits@1		Hits@3		Hits@10		MRR		Hits@1		Hits@3		Hits@10	
	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary	Binary	N-ary
RAE	0.115	0.397	0.050	0.294	0.108	0.434	**0.247**	0.618	0.169	0.187	0.096	0.126	0.178	0.198	0.323	0.306
NaLP	**0.118**	**0.477**	**0.058**	**0.394**	**0.121**	**0.512**	0.246	**0.637**	**0.351**	**0.283**	**0.291**	**0.187**	**0.374**	**0.322**	**0.465**	**0.471**

Table 4: Experimental results of role prediction in terms of MRR and Hits@{1, 3, 10}.

Dataset	MRR			Hits@1			Hits@3			Hits@10		
	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall	Binary	N-ary	Overall
JF17K	0.811	0.831	0.825	0.738	0.773	0.762	0.872	0.874	0.873	0.928	0.927	0.927
WikiPeople	0.728	0.763	0.735	0.578	0.670	0.595	0.856	0.835	0.852	0.941	0.922	0.938

elucidate that NaLP is more effective in picking exactly valid values out.

To further elaborately evaluate the performance of NaLP on value prediction, we group the test set into binary and n-ary categories according to the arities of the facts. The detailed results on these two categories are presented in Table 3.

It can be seen from Table 3 that on the two datasets, NaLP demonstrates its advantage on both binary and n-ary categories. In more detail, on JF17K, the performance gap between NaLP and RAE is pronounced, especially in terms of MRR, Hits@1 and Hits@3. On WikiPeople, RAE is more significantly left behind by a large margin. These verify that NaLP is able to well deal with value prediction on both binary and n-ary categories. Notably, on JF17K, both RAE and NaLP perform much more poorly on binary category than on n-ary category. To analyze the reason behind, we dig into the dataset. It can be observed from Table 1 that JF17K has less n-ary relations. Although, the ratios between the sizes of n-ary and binary categories on the relation set ($136/186 = 0.731$) and the training set ($32,169/44,210 = 0.728$) are similar and less than 1, JF17K has much more test facts on n-ary category and the above ratio is greater than 1 ($14,151/10,417 = 1.358$). Since the test facts are used to tune the hyper-parameters (due to the lack of validation set), the relatively more facts on n-ary category in the test set encourage the models to do prediction on n-ary category much better. Thus, it is reasonable that both RAE and NaLP obtain worse performance on binary category than on n-ary category in JF17K.

4.3.2 Role Prediction. As there is no other method that conducts role prediction on n-ary relational data, and even RAE is deliberately designed only for value prediction, we conduct role prediction experiment only on the proposed NaLP method. Table 4 demonstrates the detailed experimental results in terms of all the four metrics on binary category, n-ary category and the whole dataset. The experimental results also illustrate the power of NaLP. NaLP achieves excellent results on all the metrics. We attribute this to the reasonable modeling of role-value pairs. Thus, it not only enhances the performance of value prediction, but also benefits role prediction.

4.4 Overall Relatedness Analysis

In the proposed NaLP method, the overall relatedness vector of a set of role-value pairs, i.e., a relational fact, is the crucial intermediate result. We conduct further analyses to dig deep into what it has learned.

According to Section 3.2, a valid relational fact is expected to have an overall relatedness vector of large value, while an invalid relational fact is on the contrary. And how to evaluate the degree

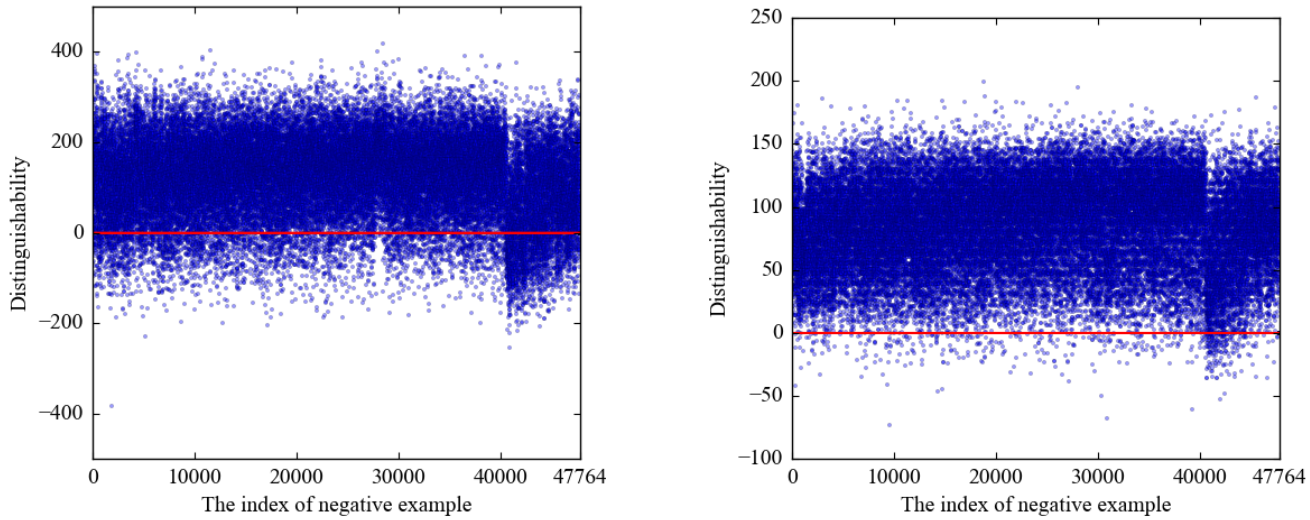

Figure 2: The visualization of the distinguishability results of Case-1 concerning the binary relational fact Fact-1: {*son* : *Michael Douglas*, *father* : *Kirk Douglas*} (left) and Case-2 concerning the n-ary relational fact Fact-2: {*person* : *Marie Curie*, *award* : *Nobel Prize in Physics*, *point in time* : *1903*, *together with* : *Henri Becquerel*, *together with* : *Pierre Curie*} (right).

of largeness? Actually, the relative magnitude between a valid relational fact and its negative samples is relatively more meaningful. Specifically, it is expected that a valid relational fact has larger values in a majority of dimensions of the overall relatedness vector compared to its negative samples, and thus the valid relational fact is distinguishable from its negative samples. Therefore, we propose the following metric to measure this type of distinguishability:

$$d(Rel^+, Rel^-) = \text{sum}\left(\text{sgn}(\mathcal{R}_{\text{Rel}^+} - \mathcal{R}_{\text{Rel}^-})\right) - \text{sum}\left(\text{sgn}(\mathcal{R}_{\text{Rel}^-} - \mathcal{R}_{\text{Rel}^+})\right), \tag{9}$$

where Rel^+ is a valid relational fact, and Rel^- is one of its negative samples; $\mathcal{R}_{\text{Rel}^+}$ and $\mathcal{R}_{\text{Rel}^-}$ are the overall relatedness vectors of Rel^+ and Rel^-, respectively; $\text{sgn}(x)$ is the function that returns 1, if $x > 0$, otherwise, returns 0; $\text{sum}(\cdot)$ is the element-wise sum function. In Equation (9), the left part of the minus sign counts the number of dimensions that $\mathcal{R}_{\text{Rel}^+}$ is larger than $\mathcal{R}_{\text{Rel}^-}$, and the right part is defined similarly. Hence, $d(Rel^+, Rel^-)$ measures the relative amount that $\mathcal{R}_{\text{Rel}^+}$ has more dimensions of larger values than $\mathcal{R}_{\text{Rel}^-}$.

Based on this proposed metric, we select two typical cases from the validation set, one binary case and one n-ary case, which are predicted correctly by NaLP, to carry out overall relatedness analysis. Note that, we only sample cases regarding value prediction, since role prediction is much simpler due to the much smaller role set. The sampled cases are as follows:

- Case-1: Predict *Michael Douglas*, given the role *son* and the remaining role-value pair of the binary relational fact {*son* : *Michael Douglas*, *father* : *Kirk Douglas*}, denoted as Fact-1.
- Case-2: Predict *Nobel Prize in Physics*, given the role *award* and the remaining role-value pairs of the n-ary relational fact {*person* : *Marie Curie*, *award* : *Nobel Prize in Physics*, *point in time* : *1903*, *together with* : *Henri Becquerel*, *together with* : *Pierre Curie*}, denoted as Fact-2.

Similar to the evaluation procedure, we replace *Michael Douglas* in Fact-1 and *Nobel Prize in Physics* in Fact-2 with all the values in V, then these corrupted facts are fed into NaLP to obtain the overall relatedness vectors. Subsequently, the distinguishability metric is computed following Equation (9). Figure 2 depicts these distinguishability results.

As exhibited in Figure 2, most distinguishability results lie in the area above 0. It visually corroborates that the overall relatedness vector of a fact really captures many discriminant features to further estimate the validity of the fact. Therefore, our conjecture on relatedness (see Section 3.2) makes sense to a certain degree. Due to the diversity of relational facts, the dimension of the overall relatedness vector is encouraged to be large. In this way, there is sufficient dimensions to encode various features. This is consistent with the large optimal value of n_{gFCN} (see Section 4.2.2). Specifically, in the left part of Figure 2, corresponding to the binary Case-1, the distinguishability metric of the negative sample indexed as 1844, corresponding to *Kirk Douglas* is extremely small. As the two role-value pairs *son* : *Kirk Douglas* and *father* : *Kirk Douglas* share the same part *Kirk Douglas*, there is no doubt that their relatedness is large, and the negative sample formed by this two role-vale pairs is easy to be judged as valid. NaLP is unable to filter out such situation. Actually, one value usually cannot correspond to two or more roles of the same fact. Thus, some rules may helpful to deal with the above deficiency. We leave it for future work. From the right part of Figure 2, corresponding to the n-ary Case-2, we have the following two observations:

- The negative samples indexed as 9556, 30913 and 39270, corresponding to *Marie Curie*, *Henri Becquerel* and *1903*, obtain the smallest, the second smallest and the third smallest

distinguishability results, respectively. These three values participate in some other roles of Fact-2, which is similar to the situation of *Kirk Douglas* in Case-1.

- Besides the negative samples corresponding to values of type *human*, some negative samples corresponding to values of type *time*, *club* or *institution* also achieve relatively smaller distinguishability results. It is acceptable, since NaLP does not take value type into consideration.

4.5 Hyper-parameter Analysis

To investigate the robustness of the proposed NaLP method, we further analyze the impacts of its important hyper-parameters, i.e., the number of filters n_f in the role-value pair embedding component and the hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ in the relatedness evaluation component. Without loss of generality, these experiments are all conducted on WikiPeople concerning value prediction.

4.5.1 The Impact of the Number of Filters n_f. To make the comparison fair, we keep all the hyper-parameters except n_f to be the optimal settings, and only vary the value of n_f in {50, 100, 200, 400, 500}, as mentioned in Section 4.2.2. To be clear, the evaluation results on the validation set with regard to only MRR are illustrated in the left part of Figure 3.

From the left part of Figure 3, it can be observed that the value of n_f affects the results indistinctively when the value lies in {100, 200, 400, 500}. Thus, NaLP is insensitive to the value of n_f, when the value is not too small, and is robust. The best performance with respect to MRR is obtained when n_f is set to 200.

4.5.2 The Impact of the Hyper-parameter n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$. Similarly, we vary the value of n_{gFCN} in {50, 100, 200, 400, 500, 800, 1000, 1200} and adopt all the other optimal hyper-parameter settings. We also demonstrate the evaluation results on the validation set in terms of only MRR in the right part of Figure 3.

In the right part of Figure 3, it turns out that different values of n_{gFCN}, no less than 400, demonstrate insignificant difference on performance and NaLP is thus robust. Among all these setting, $n_{\text{gFCN}} = 1200$ results in slightly better performance.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we represented each n-ary relational fact as a set of its role-value pairs and proposed a method to cope with Link Prediction on N-ary relational data, called NaLP. The design of the framework equips NaLP with the feature of permutation invariance to the input order of the role-value pairs, and the ability to well handle relational facts of different arities. By evaluating the pair-wise relatedness of all the role-value pairs in the same relational fact, NaLP is able to estimate their overall relatedness approximately. The resulting overall relatedness vectors further enable NaLP to pick up many determinant features to decide the validity of the input relational facts. Furthermore, since publicly available n-ary relational datasets are limited, we developed a practical one, WikiPeople, and it was published for further research. Experimental results on the public dataset and the newly developed WikiPeople manifest the merits and superiority of the proposed NaLP method. On value prediction, compared to the state-of-the-art method, NaLP improves all the metrics significantly, especially on WikiPeople. Specifically, NaLP improves the performance even by 7.1% in terms of Hits@1 on the public dataset, and has more than 17.0% increase in terms of Hits@1 and Hits@3 on WikiPeople.

For future work, on one hand, we will explore more expressive neural network models to capture more favorable features for validity evaluation. On the other hand, in this paper, we use only relational facts to conduct link prediction, and in the future we will introduce additional information, such as rules and value types, to further improve the developed model.

ACKNOWLEDGMENTS

The work is supported by the National Key Research and Development Program of China under grant 2016YFB1000902, the National Natural Science Foundation of China under grants 61772501, 61572473, 61572469, and 91646120, and the GFKJ Innovation Program.

REFERENCES

[1] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD'08)*. 1247–1250.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*. 2787–2795.

[3] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*. 1811–1818.

[4] Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL'18)*. 110–121.

[5] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'14)*. ACM, 601–610.

[6] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13rd International Conference on Artificial Intelligence and Statistics (AISTATS'10)*. 249–256.

[7] Saiping Guan, Xiaolong Jin, Yuanzhuo Wang, and Xueqi Cheng. 2018. Shared embedding based neural networks for knowledge graph completion. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM'18)*. 247–256.

[8] Nicholas Guttenberg, Nathaniel Virgo, Olaf Witkowski, Hidetoshi Aoki, and Ryota Kanai. 2016. Permutation-equivariant neural networks applied to dynamics prediction. *arXiv preprint arXiv:1612.04530* (2016).

[9] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 318–327.

[10] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'15)*. 448–456.

[11] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL'15)*. 687–696.

[12] Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 985–991.

[13] Yantao Jia, Yuanzhuo Wang, Hailun Lin, Xiaolong Jin, and Xueqi Cheng. 2016. Locally adaptive translation for knowledge graph embedding. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 992–998.

[14] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*. 705–714.

[16] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. 2181–2187.

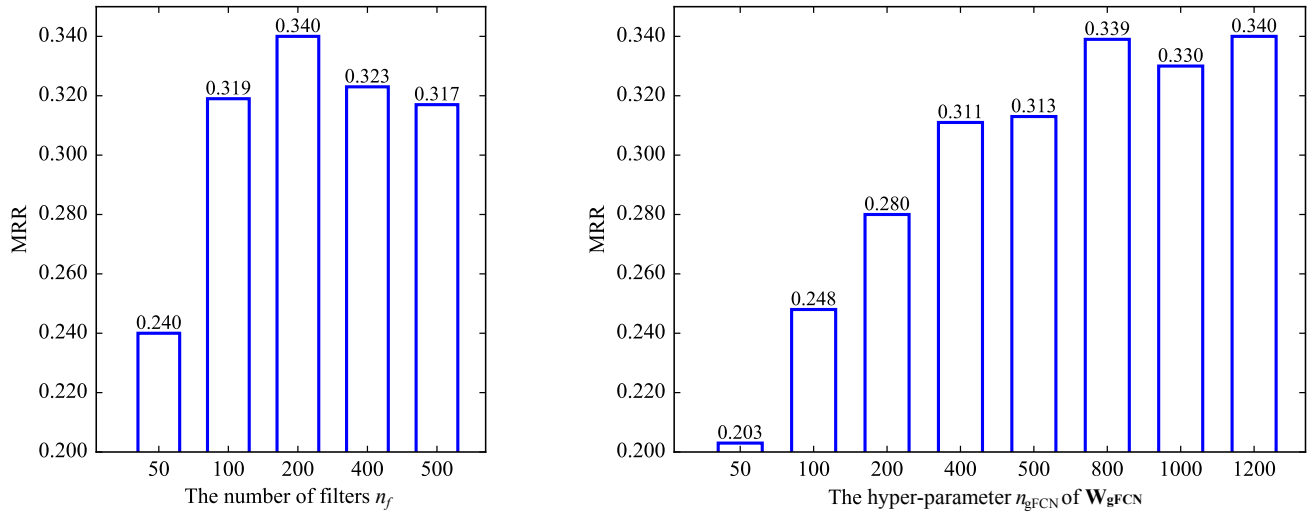

Figure 3: Performance comparison of the proposed NaLP method over different numbers of filters n_f (left) and different hyper-parameters n_{gFCN} of $\mathbf{W}_{\mathbf{gFCN}}$ (right) on WikiPeople concerning value prediction.

[17] Hitoshi Manabe, Katsuhiko Hayashi, and Masashi Shimbo. 2018. Data-dependent Learning of Symmetric/Antisymmetric Relations for Knowledge Base Completion. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI'18)*. 3754–3761.

[18] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML'10)*. 807–814.

[19] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'18)*, Vol. 2. 327–333.

[20] Maximilian Nickel, Xueyan Jiang, and Volker Tresp. 2014. Reducing the rank in relational factorization models by including observable patterns. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*. 1179–1187.

[21] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2016. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2016), 11–33.

[22] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI'16)*. 1955–1961.

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. 809–816.

[24] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Tim Lillicrap. 2017. A simple neural network module for relational reasoning. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NIPS'17)*. 4967–4976.

[25] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th Extended Semantic Web Conference (ESWC'18)*. 593–607.

[26] Baoxu Shi and Tim Weninger. 2017. ProjE: Embedding projection for knowledge graph completion. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI'17)*. 1236–1242.

[27] Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Falk Brauer. 2017. Random semantic tensor ensemble for scalable knowledge graph link prediction. In *Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM'17)*. 751–760.

[28] ThÃĩo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on Machine Learning (ICML'16)*, Vol. 48. 2071–2080.

[29] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[30] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI'14)*. 1112–1119.

[31] Jianfeng Wen, Jianxin Li, Yongyi Mao, Shini Chen, and Richong Zhang. 2016. On the representation and embedding of knowledge bases beyond binary relations. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. 1300–1307.

[32] Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. TransG: A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL'16)*. 2316–2325.

[33] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR'15)*.

[34] Richong Zhang, Junpeng Li, Jiajie Mei, and Yongyi Mao. 2018. Scalable instance reconstruction in knowledge bases via relatedness affiliated embedding. In *Proceedings of the 27th International Conference on World Wide Web (WWW'18)*. 1185–1194.