Discovering Latent Representations of Relations for Interacting Systems

DOHAE LEE¹, YOUNG JIN OH², AND IN-KWON LEE.³
¹Yonsei University, Seoul (e-mail: dlehgo1414@gmail.com)
²LG Electronics, Seoul (e-mail: skrcjstk@gmail.com)
³Yonsei University, Seoul (e-mail: iklee@yonsei.ac.kr)

Corresponding author: In-Kwon Lee (e-mail: iklee@yonsei.ac.kr).

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2021-2018-0-01419) supervised by the IITP (Institute for Information and Communications Technology Planning and Evaluation) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT). (No. NRF-2020R1A2C2014622)

ABSTRACT

Systems whose entities interact with each other are common. In many interacting systems, it is difficult to observe the relations between entities which is the key information for analyzing the system. In recent years, there has been increasing interest in discovering the relationships between entities using graph neural networks. However, existing approaches are difficult to apply if the number of relations is unknown or if the relations are complex. We propose the DiScovering Latent Relation (DSLR) model, which is flexibly applicable even if the number of relations is unknown or many types of relations exist. The flexibility of our DSLR model comes from the design concept of our encoder that represents the relation between entities in a latent space rather than a discrete variable and a decoder that can handle many types of relations. We performed the experiments on synthetic and real-world graph data with various relationships between entities, and compared the qualitative and quantitative results with other approaches. The experiments show that the proposed method is suitable for analyzing dynamic graphs with an unknown number of complex relations.

INDEX TERMS

graph neural network, relational inference, unsupervised learning

I. INTRODUCTION

Most entities in nature are related to each other, interacting based on their relationship, and changing their states over time based on such interactions. An interacting system can be represented as a dynamic graph [1] in which the attributes of the entities change over time.

In most cases, although changes in the states of the entities can be observed, the relations among the entities have rarely been observed. It is important to discover these hidden relationships between entities in a dynamic graph, which helps us to better understand the system and predict future states. For example, inferring the relationship between physically interacting objects helps us understand the entire system and predict the future movements of objects. We can also analyze the specific motions by inferring the intrinsic relationships between the joints of the human body, and more accurately predict the future motion of each joint.

Several attempts have recently been made to infer the relationships between entities by observing the states of such entities in a dynamic graph, and to predict the future states of entities based on their relationships [2]–[5]. It is possible to cluster the relations between entities and accurately predict their future states. However, the existing models are designed to have a dedicated update function for each relation, making it difficult to model the system if the number of relationships is not known in advance. In addition, existing models can only discretely classify different relations, making them difficult to apply to data with complex relations. Furthermore,
although existing models can distinguish different relations, it is difficult to analyze the relationships between these relations, such as their similarities. These issues are not trivial because entities in nature may have a complex and unknown number of relations.

To overcome these challenges, we propose a DiScovering Latent Relation (DSLR) model composed of two graph neural networks (GNNs) [6]–[9]. With DSLR, a relation encoder, i.e., the first GNN, embeds the relations between entities in the latent space. The relation decoder, i.e., the second GNN, predicts the future states of the entities using the recognized relations. The concept of DSLR is depicted in Fig. 1. The figure on the left shows the dynamics of physical objects interacting with each other. The color of the arrows between objects indicates the type of relation between objects. Each relation between objects can be represented in the relation latent space shown on the right side of the figure, in which the same relations are close to each other, and different relations are distant from each other.

Unlike previous methods that cluster relations discretely using softmax, our DSLR can better explain complex relations by embedding relations into a relation latent space. Even if the number of relations is unknown in advance, DSLR can not only accurately discover the relations, it can also infer the number of relationships in an interacting system. In addition, the relation decoder of DSLR predicts the future states of entities for all relation types; thus, even if the number of relations is large or complex, there is no need to modify or scale the model up.

The main contributions in our work consist of:
- We design a new relational inference model that can be applied even when the number of relationships in the system is unknown.
- Our method is more efficient for the interacting system with very complex relationships between entities, such as real-world motion capture data and basketball data.

The remainder of the paper is organized as follows. The related works for better understanding the proposed framework are discussed in Section II. Section III discusses the proposed DSLR method, and Section IV shows the experimental results. Finally, Section V offers the conclusion.

II. RELATED WORK

A. RELATIONAL INDUCTIVE BIAS

With the advent of deep learning, many important problems have been solved in various fields, including image processing, natural language processing, and control [10]–[13]. In recent years, there has been increasing interest in using relational inductive biases with deep learning architectures to solve relational reasoning problems [14]. For example, to allow multiple agents to cooperate in maximizing common rewards, a CommNet [15] model designed to allow agents to communicate with each other was proposed. An interaction network [16] model was also proposed to predict the future movement of physically interacting objects. Some researchers have addressed a visual question answering task that requires consideration of the relationship between objects [17], and attempts have been made to solve various relation-based tasks using a self-attention technique [18]–[20]. In addition, a few-shot learning technique that considers relational information has been proposed [21], and a method for distinguishing and grouping objects from images was proposed [22]. In addition, a number of GNN-based methods [23]–[27] with a strong relational inductive bias have been proposed to solve various relational reasoning tasks. However, these methods did not explicitly infer the relations between entities.

B. INFERRING EXPLICIT RELATION

Meanwhile, there have been attempts to explicitly infer the relations between entities. For example, several studies have explored how to infer relations in the fields of causal reasoning [28] and computational neuroscience [29], [30]. Methods for clustering the relations between entities and predicting the future states of such entities using a GNN [6]–[9] with a dynamic graph showing the changes in the states of the entities over time have recently been reported. The neural relational inference (NRI) [2] model, which is an unsupervised neural network model that can discretely distinguish relations between entities and predict the dynamics of such entities in interacting systems, has been proposed. The factorized NRI (fNRI) [3] model, which complements the NRI model and efficiently handles combinations of independent interactions, was also proposed, as was the SUGAR [4] model, which modifies the NRI model to consider global interactions with various structured priors. In addition, a dynamic NRI (dNRI) model [5], a method that can better infer the changing relations between entities in an interaction system, was developed.

These NRI-based methods can discover relations between entities and predict the future states of such entities in various interacting systems. However, because these methods are designed to handle each relation type with a dedicated update function, they may be difficult to apply to systems with unknown or large numbers of relationships. By contrast, our DSLR model is designed to represent relations in a latent space and handle all relations with a single update function; thus, it can be applied to systems with complex or unknown numbers of relations in a flexible manner.

III. DISCOVERING LATENT RELATION MODEL

A. MODEL DETAILS

The DSLR model consists of two subnetworks, i.e., a relation encoder and a relation decoder (see Fig. 2).

1) Relation Encoder

The first component of the DSLR is the relation encoder (see Fig. 2(b)), which is a network that infers the relations between nodes by observing the states of the nodes (entities) in interacting systems for a certain period of time. The relation encoder first infers the edge states before inferring the relation states containing the relation information between two nodes. The edge state is information that serves as a clue to infer the
relation state, which contains state information during several time steps of the edges between nodes. In a system with \( N \) nodes, the edge state \( s_{ij}^t \) between the \( i \)-th node \( n_i^t \) and the \( j \)-th node \( n_j^t \) at time \( t \) is defined as follows:

\[
  s_{ij}^t = \mathcal{F}(n_i^t, n_j^{t-1}), \quad i, j = 1, ..., N, \quad i \neq j, \quad (1)
\]

where \( \mathcal{F} \) is a function that updates the edge state \( s_{ij}^t \) to \( s_{ij}^{t+1} \) from the edge state and node states of the previous time step.

Next, the relation encoder infers the relation state between the nodes based on the edge state. We use a reparameterization trick of a variational auto-encoder [31] to enable the stochastic representation of the latent vectors. We model each component in the latent vector to have a prior with a normal distribution having a mean of zero and a standard deviation of 1. When the graph is observed for \( T_E \) time steps from the first time-step, the relation state \( r_{ij} \) between the \( i \)-th and \( j \)-th nodes is obtained as follows:

\[
  r_{ij} \sim \mathcal{G}(s_{ij}^{T_E}), \quad (2)
\]

where \( \mathcal{G} \) is a function that models the distribution of the relation state from the edge state, and \( \sim \) is a sampling operation using the reparameterization trick.

The relation encoder can infer the relation centrality, which is the importance of the relation. The relation encoder obtains the relation centrality from the edge state in the same way in which the relation state is inferred. The relation centrality \( c_{ij} \) between nodes \( i \) and \( j \) is obtained as follows:

\[
  c_{ij} = \sigma(\mathcal{H}(s_{ij}^{T_E})), \quad (3)
\]

where \( \mathcal{H} \) is a function that infers the relation centrality from the edge state, and \( \sigma \) is a sigmoid function. The relation centrality \( c_{ij} \) is a scalar value between zero and 1, and a large \( c_{ij} \) indicates that the relation between \( i \) and \( j \)-th nodes is important, and vice versa.

2) Relation Decoder

The relation decoder predicts the future states of the nodes from the current states of the nodes and the relation state between them, as inferred by the relation encoder. The relation decoder first infers the influences exchanged between nodes, aggregates all influences, and predicts the future state of the nodes when considering all effects applied to each node. The influence of the \( j \)-th node on the \( i \)-th node at time \( t \), \( f_{ij}^t \), is modeled as follows:

\[
  f_{ij}^t = \mathcal{J}(c_{ij})\mathcal{K}(n_i^t, n_j^t, r_{ij}), \quad (4)
\]

where \( \mathcal{K} \) is a function that computes the influence of nodes on each other from the states of the nodes and their relation state, and \( \mathcal{J} \) is a function that calculates the noise multiplied by the influence according to the relation centrality \( c_{ij} \). The noise applied to the influence \( f_{ij}^t \) is obtained as follows:

\[
  f_{ij}^t = 1 + (1 - c_{ij})\epsilon, \quad (5)
\]

where \( \epsilon \) is a random variable. That is, the larger the relation centrality (closer to 1) is, the less noise applied to the influence, and the smaller the relation centrality (closer to 0) is, the greater the amount of noise.

Next, the future state of the \( i \)-th node is calculated as follows:

\[
  n_i^{t+1} = n_i^t + \mathcal{L}\left(n_i^t, \sum_{j, i \neq j} f_{ij}^t\right), \quad (6)
\]

where \( \mathcal{L} \) represents a function that predicts change in the node states.

B. RELATION REASONING

To cluster the relations between nodes when the number of relations is small, unsupervised clustering [32] is conducted on the relation states embedded in the relation latent space. We train the \( k \)-means clustering model with the relation states obtained from the training data, and cluster the relation states of the test data with the trained model. If the number of
relations is unknown, the number of relations can be inferred using the silhouette method [33]. The larger the silhouette score, the higher the probability that \( k \) will be optimal, which means that \( k \) with the highest silhouette score is the number of relations.

C. RANDOM SAMPLING TRICK
It is desirable that the relation state contains only information about the relationship between nodes. However, when training a model, if the input of the relation encoder and input of the relation decoder are not independent of each other, the relation states learn the compressed information of input trajectories of nodes instead of relational information, which is undesirable. In a previous study [3], a model in which the module inferring the relation learns the compressed version of the input trajectories rather than the relation is called a ‘compression model.’

To prevent our relation encoder from becoming compression model, we propose a random sampling trick, i.e., while training the model, the times required to infer the relation state and to compute the influence between nodes are randomly sampled. In Fig.2(a), the relation encoder infers the relation state between nodes by observing the graph from time \( t_E \) for \( T_E \) time-steps, and the relation decoder predicts the graph of the next \( g_{t_D+1} \) from the inferred relation state \( r \) and the graph \( g_{t_D} \) at time \( t_D \). A random sampling trick is a technique that randomly samples \( t_E \) and \( t_D \) each time during training.

D. TRAINING
The DSLR model is trained without supervision of the relation states in an end-to-end manner, which is optimized based on four loss functions, i.e., the node prediction loss, KL divergence loss, relation standard deviation loss, and relation centrality loss.

The node prediction loss \( L_{NP} \) is the mean squared error between the predicted future states of the nodes from time \( t_D \) for \( T_D \) time steps and the true future states:

\[
L_{NP} = \sum_{t=t_D}^{t_D+T_D} (\hat{n}_i^t - n_i^t)^2, \tag{7}
\]

where \( \hat{n}_i^t \) is the ground truth state of the nodes at time \( t \), and \( n_i^t \) is the state of the nodes at time \( t \) predicted by the model.

KL divergence loss induces the distribution of each component in the relation states to follow a normal distribution. KL divergence loss \( L_{KL} \) is defined as follows:

\[
L_{KL} = -D_{KL} (P(r)||N(0, 1)), \tag{8}
\]

where \( P(r) \) represents the distribution of each component in the relation states, which are the outputs of the relation encoder, \( N(0, 1) \) represents the standard normal distribution, and \( D_{KL} \) represents the Kullback–Leibler divergence.

Our model deals with a dynamic graph with a static relation where the states of the nodes change over time, but the relation between nodes does not change. In a single dynamic graph, the relation between nodes is constant. Therefore, in the same graph, the relation between nodes is constant regardless of the initial time \( t_E \) of the graph input to the relation encoder (see Fig.2(a)). The relation standard deviation loss \( L_{SD} \) is the standard deviation of the relation states inferred by the relation encoder for a sequence of graphs randomly sampled \( m \) times within a dynamic graph:

\[
L_{SD} = STD(r_1, r_2, ..., r_m), \tag{9}
\]

where \( STD \) indicates the standard deviation, and \( r_i \) indicates the relation state recognized in the \( i \)-th sampled sequence of graphs.

In (4-5), the larger the relation centrality \( c_{ij} \), the smaller the noise \( J(c_{ij}) \) that occurs in the influence \( f_{ij} \) between nodes. Conversely, as the relation centrality decreases, the noise of the influences exchanged between nodes increases. Therefore, for the model to correctly predict the future states of the nodes, it is trained in the direction of increasing the relation centrality to decrease \( L_{NP} \). Contrary to the tendency of the relation centrality to become larger, we add a relation centrality loss to the objective function, which leads to a decrease in the relation centrality:

\[
L_c = -\log(1 - c), \tag{10}
\]

where \( c \) denotes the relation centrality. While the node prediction loss \( L_{NP} \) is designed to model the interacting system correctly, the relation centrality loss acts in the opposite manner. When the magnitude of the influence exchanged between nodes \( f_{ij} \) is large, the magnitude of \( L_{NP} \) is larger than that of \( L_c \), and thus the relation centrality increases to reduce the noise of \( f_{ij} \). Conversely, when \( f_{ij} \) is small, the magnitude of \( L_c \) becomes relatively smaller than that of \( L_{NP} \), and thus the relation centrality decreases. As a result, the relation with a large relation centrality has a large overall effect on the interacting system, whereas the relation with a small relation centrality has a small effect.

We combined all of the proposed losses and set the final objective function \( L \) as follows:

\[
L = \lambda_{NP} L_{NP} + \lambda_{KL} L_{KL} + \lambda_{SD} L_{SD} + \lambda_c L_c, \tag{11}
\]

where the weights \( \lambda_{NP}, \lambda_{KL}, \lambda_{SD}, \) and \( \lambda_c \) used for scaling each loss function were set to \( \lambda_{NP} = 1, \lambda_{KL} = 0.1, \lambda_{SD} = 1, \) and \( \lambda_c = 0.001. \)

E. SPARSITY PRIOR
In this section, we introduce a method to set the sparsity prior to DSLR. A graph in which most of the nodes are not connected to each other is called a sparse graph. Most interacting systems in nature are sparse, and thus it would be useful if the model can reflect the sparsity of the graph as a prior. To add a sparsity prior to the model, we redefine the relation centrality loss defined in [10]. Letting \( p \) be the sparsity prior of the graph, the relation centrality loss \( L_c \) is then redefined as follows:

\[
L_c = -\delta_{c,p} \log (1 - c) - (1 - \delta_{c,p}) \log (c). \tag{12}
\]
If the relation centrality $c$ falls within a small $p\%$ in the training batch, $\delta_{c,p} = 1$; otherwise, $\delta_{c,p} = 0$. That is, the modified relation centrality loss induces a smaller relation centrality within a small $p\%$ (which indicates no connection relation), and a larger relation centrality within a large $(1-p)\%$ (which indicates a meaningful relation).

IV. EXPERIMENTS

The DSLR was optimized using the Adam optimizer [34] and implemented using Pytorch [35]. The learning rate was scheduled using a one-cycle learning rate policy [36]. All experiments were conducted using a 3090 GPU, and the numerical results were the average of three trials with different seeds.

A. PHYSICS SIMULATION

To evaluate our DSLR model, we conducted experiments using physically simulated data in which objects have various relationships with each other. In our experiment, we chose relationships types that are frequently used in previous studies [2], [16], [23], [24], [37]: spring, gravity, and none. Spring is a relation in which a spring force acts between objects; if the objects are far from each other, an attractive force acts on the objects; however if they are close, a repulsive force acts on them. Gravity is a relation in which an attractive force acts according to the distance between objects; however we set the attractive force inversely proportional to the distance instead of the squared distance. None indicates that there is no connection between objects where the nodes do not transmit any forces to each other. We generated physically simulated data with eight combinations of relation types for the experiments: (a) spring & none, (b) spring & gravity, (c) gravity & none, (d) spring & gravity & none, (e) 3 spring & none, (f) 3 gravity & none, (g) 100 spring, and (h) 100 spring & 100 gravity. In (a)–(d), objects are connected by one of the corresponding relations. In (e) and (f), the objects are connected in one of the four relations based on the weak, moderate, and strong spring or gravity forces and none. In (g), a system is composed of 100 types of relations between objects, where the objects are connected by springs with 100 different coefficients. In (h), a system consists of 200 types of spring relations with 100 different coefficients and gravity with 100 different coefficients between nodes. Simulations were generated by assigning the potential between two objects as in [24]. We conducted experiments to infer the relation between objects and predict the future trajectory of the objects in an interacting system in which a number of objects move in a complex manner. All datasets consisted of 5000 training sets, 500 validation sets, and 500 test sets. Each data consists of 99 time steps, of which 49 are observed to infer the relation. We set $m$ in (9) to 5 in all experiments using physically simulated data.

We conducted comparative experiments using the NRI model [2], which can discover the relation between objects. The NRI model was trained using the training schema proposed by fNRI [3] to prevent it from becoming a compression model, during which, the first half of the physical data sequence was used to infer the relation, and the second half was used for a future trajectory prediction. Because the NRI model requires a dedicated decoder for each relation, the number of relations must be entered into the model in advance. When comparing the accuracy of the relation inference, we assumed that the true number of relations $K$ is known to the NRI. We also compared the result if the wrong number of relations ($\tilde{K} = 2K$) is set to the NRI. When comparing trajectory prediction errors, both the NRI with the correct number of relations ($\tilde{K} = K$) and the NRI with an incorrect number of relations ($\tilde{K} = \lceil K/2 \rceil, \tilde{K} = 2K$) were used in the experiment. Neither the NRI model nor our DSLR model were given the sparsity prior. The DSLR model was trained with $\epsilon$ in (4) set to zero because we did not aim to infer the relation centrality in this experiment, except for the relation centrality verification experiments. The models were trained for 1000 epochs.

1) Relation Reasoning

We first explored a relational reasoning task. The results of estimating the relation type in the physical data using the DSLR and NRI models are shown in Table I. Each value represents the accuracy of correctly classifying the types of relations between objects. Both the DSLR and NRI models can estimate the relation types with high accuracy, and the superiority of the two models differs depending on the data. However, if the number of relations is greater than two (spring & gravity & none, 3 spring & none, 3 gravity & none), the accuracy of our model is significantly superior to that of NRI. When the number of relations was incorrectly entered in the NRI, the accuracy was generally low (Table I $\tilde{K} = 2K$ row).

Unlike NRI, where the number of relations must be set in advance, DSLR can infer the number of relations in the system using the silhouette score [33]. Table II shows a silhouette score for each number of relations $K$, i.e., the higher the silhouette score is, the higher the probability that $K$ is proper.
with a large number of relations, the NRI model was trained on the inferred relation states. The mean squared errors of the relation state into 2D using a principal component analysis [38]. The points indicating the dimension-reduced relation states are marked with the same color if the relation types are the same. The experimental results show that the relation states are located on the right side of the latent space, and that different relations are placed far from each other. Even when the number of relations is large, DSLR can arrange the relation states within the relation latent space, as shown in Fig. 3 after reducing the dimension size was set to 128 for $K$ = 1, 6, 16, 50, 100, where the batch size was set to 128 for $K$ = 1, but 32 for $K$ = 50, and 16 for $K$ = 100 owing to memory limitations. As $K$ increases, the future prediction error decreases; however, the number of parameters in the model increases (see Fig. 4(a)). In addition, the errors in the NRI are still greater than the errors in the DSLR, and the number of parameters in the DSLR is lower. An example of the results of predicting the future trajectory of an object is shown in Fig. 5. For data with a small number of relations, both the DSLR and NRI models predicted a future trajectory similar to the truth (see Fig. 5(a)), whereas for data with a large number of relations, the DSLR predicted the trajectories more similarly than the NRI (see Fig. 5(b)).

2) Future Trajectory Prediction

The DSLR can predict the future states of the nodes based on the inferred relation states. The mean squared errors of the future trajectory predicted by the DSLR and NRI models are listed in Table 3. For the last two datasets in Table 3 with a large number of relations, the NRI model was trained by setting the given number of relations $K$ to 8, because in our experiments, NRI with a larger $K$ requires too many computing resources, but does not significantly improve the performance.

For the first three datasets with a small number of relations, the superiority of the DSLR model over the NRI model with correct $K$ differs depending on the combinations of relations. Although, the NRI with larger $K$ is less accurate at classifying the relations, NRI can predict the future trajectories better than the other models for the first three datasets (see the row “$K = 2K$” in Table 3). If a smaller number of relations are given to the NRI than the number in reality, the error is much larger (see the row “$K = \lceil K/2 \rceil$” in Table 3). However, if the number of relations is larger than two, DSLR outperforms the other models on all datasets. In particular, when the relations are almost continuous, as in the last two datasets, DSLR is able to predict the future trajectories much more accurately than the other models. Fig. 4(b) and (c) show the prediction error of the future trajectories of the NRI model for each number of relation types given to the model in the last two datasets. We trained the NRI with $K \in \{2, 3, 4, 5, 6, 7, 8, 9, 10, 16, 50, 100\}$, where the batch size was set to 128 for $K$ = 1, 6, 16, 50, and 16 for $K$ = 100 owing to memory limitations. As $K$ increases, the future prediction error decreases; however, the number of parameters in the model increases (see Fig. 4(a)). In addition, the errors in the NRI are still greater than the errors in the DSLR, and the number of parameters in the DSLR is lower. An example of the results of predicting the future trajectory of an object is shown in Fig. 5. For data with a small number of relations, both the DSLR and NRI models predicted a future trajectory similar to the truth (see Fig. 5(a)), whereas for data with a large number of relations, the DSLR predicted the trajectories more similarly than the NRI (see Fig. 5(b)).

3) Relation Centrality

The DSLR can infer the relation centrality, which is an indicator of the importance of the relation. To verify that the relation centrality $c_{ij}$ inferred by our model correctly recognizes the importance of the relation $r_{ij}$, we trained the DSLR by setting $\epsilon$ in [5] as a random variable, and compared the average relation centrality for each relation type. Fig. 6 shows the average relation centrality for each relation type in the 3 spring & none, 3 gravity & none, and 100 spring and gravity & none datasets. The mean squared errors of the average relation centrality for each relation are listed in Table 4. The results show that the DSLR can correctly predict the importance of the relation.

### Table 1. Accuracy of estimating the types of relation between objects in physically simulated data (in %).

| Dataset                  | NRI $\bar{K} = K$ | NRI $\bar{K} = 2K$ | DSLR | NRI $\bar{K} = K$ | NRI $\bar{K} = 2K$ | DSLR |
|--------------------------|-------------------|-------------------|------|-------------------|-------------------|------|
| SPRING & NONE            | 92.52±10.69       | 87.49±8.39        | 95.33±0.46 |
| SPRING & GRAVITY         | 75.84±11.71       | 96.12±1.62        | 95.22±0.30 |
| GRAVITY & NONE           | 92.28±1.28        | 80.50±6.87        | 90.55±1.00 |
| SPRING & GRAVITY & NONE  | 84.90±0.50        | 73.63±2.10        | 88.80±3.10 |
| 3 SPRING & NONE          | 66.38±6.72        | 64.94±5.61        | 84.35±2.24 |
| 3 GRAVITY & NONE         | 64.33±4.02        | 64.43±0.92        | 74.10±1.41 |

### Table 2. Silhouette scores of DSLR by $K$ in physically simulated data.

| Dataset                  | $K=2$ | $K=3$ | $K=4$ | $K=5$ |
|--------------------------|-------|-------|-------|-------|
| SPRING & NONE            | 0.96  | 0.93  | 0.89  | 0.73  |
| SPRING & GRAVITY         | 0.97  | 0.96  | 0.94  | 0.91  |
| GRAVITY & NONE           | 0.90  | 0.85  | 0.79  | 0.72  |
| SPRING & GRAVITY & NONE  | 0.72  | 0.83  | 0.81  | 0.78  |
| 3 SPRING & NONE          | 0.62  | 0.64  | 0.73  | 0.70  |
| 3 GRAVITY & NONE         | 0.62  | 0.60  | 0.72  | 0.69  |
We trained the DSLR model using the large motion capture trials for testing. We trained the DSLR model, NRI model, and dNRI model with the sparsity prior set to 0.91 as in the previous study [2] and assumed that 91% of the pairs of joints would have a none relation. The NRI and dNRI models were trained using four relation types determined experimentally by the authors, one of which was hard-coded for the none relation. Whereas the DSLR and NRI models estimated the static relations, dNRI, which is designed to infer the dynamic relations estimated in the experiments, assumed that the relations between human joints may change over time.

| Dataset               | $\hat{K} = \lceil K/2 \rceil$ | $\hat{K} = K$ | $\hat{K} = 2K$ | DSLR          |
|-----------------------|-------------------------------|---------------|---------------|---------------|
| SPRING & NONE         | 0.956±0.020                  | 0.021±0.025   | 0.007±0.001   | 0.034±0.008   |
| SPRING & GRAVITY      | 0.798±0.011                  | 0.219±0.310   | 0.031±0.004   | 0.085±0.006   |
| GRAVITY & NONE        | 0.996±0.004                  | 0.011±0.002   | 0.008±0.001   | 0.013±0.002   |
| SPRING & GRAVITY & NONE | 0.097±0.004              | 0.040±0.007   | 0.035±0.004   | 0.034±0.005   |
| 3 SPRING & NONE       | 0.062±0.000                  | 0.018±0.003   | 0.012±0.001   | 0.011±0.001   |
| 3 GRAVITY & NONE      | 0.048±0.002                  | 0.025±0.005   | 0.021±0.001   | 0.014±0.001   |
| 100 SPRING            | 0.113±0.002                  | 0.086±0.003   | 0.081±0.005   | 0.044±0.001   |
| 100 SPRING & 100 GRAVITY | 0.343±0.013             | 0.291±0.001   | 0.271±0.000   | 0.080±0.002   |

At the inference time, the models observed the system for 49 time-steps to estimate the relation states between joints, and then predicted the future motion of the joints for 50 time-steps. Because it is not certain whether the system has a static relation, the relational standard deviation loss was not used, and $m$ in (9) was set to 1. All models were trained for 2000 epochs.

The first row in Table 3 represents the errors of the predicted future positions of the joints in the human body as predicted by the DSLR and previous methods. The experimental results showed that the DSLR most accurately predicted the future movements of the joints. The DSLR can forecast the movement of the joints even better than dNRI, which was designed to infer dynamic relations. This is because the DSLR is more suitable for dealing with complex relations in reality because relation states are represented as continuous latent variables, whereas comparative models represent the relations discretely.

B. MOTION CAPTURE DATA

We trained the DSLR model using the large motion capture data provided by Carnegie Mellon University [39] to infer the relationship between the joints of the human body and predict the future motion of the joints. As in previous studies [2], [5], we experimented with the walking motion data of the 35th subject: 11 trials for training, 4 trials for validation, and 7 trials for testing. We trained the DSLR model, NRI model [2], and dNRI model [5] with the sparsity prior set to 0.91 as in the previous study [2]; and assumed that 91% of the pairs of joints would have a none relation. The NRI and dNRI models were trained using four relation types determined experimentally by the authors, one of which was hard-coded for the none relation. Whereas the DSLR and NRI models estimated the static relations, dNRI, which is designed to infer the dynamic relations estimated in the experiments, assumed that the relations between human joints may change over time.

The first row in Table 3 represents the errors of the predicted future positions of the joints in the human body as predicted by the DSLR and previous methods. The experimental results showed that the DSLR most accurately predicted the future movements of the joints. The DSLR can forecast the movement of the joints even better than dNRI, which was designed to infer dynamic relations. This is because the DSLR is more suitable for dealing with complex relations in reality because relation states are represented as continuous latent variables, whereas comparative models represent the relations discretely.
Fig. 7 shows two experimental results, where the red dots indicate the true positions of the joints, and the black dots indicate the positions predicted by each model. In both cases, the DSLR model predicted the joint movement of the human body would be more similar to the ground truth than the comparative models.

Fig. 8 shows the visualization of the edge centrality between joints estimated by the DSLR model: the larger the edge centrality is, the thicker the red line, and the smaller the edge centrality is, the thinner the blue line. Because the sparsity prior was applied to the model, most of the joints were connected by a weak edge centrality. The arms and legs were mainly connected by thick red lines, indicating that the DSLR judged that the relations between arms and legs are the most important during a walking motion (see Fig. 8(c)). In addition, the upper body and legs were connected by a bluish-red line, which means that the DSLR determined that the relations between the upper body and legs are less important than the relations between the arms and legs during a walking motion (see Fig. 8(d)). Finally, other relations are represented as light blue lines, which means that the DSLR seemed to be one of the correct interpretations given that the movements of the arm and legs are most pronounced when a person walks.

The DSLR and comparative models observed the first 40 steps to infer the relationship between players, and then predicted the movement of each player for 9 steps. The second row in Table 4 represents the average of the prediction errors for the nine steps predicted by each model. The numerical results show that the DSLR can predict future trajectories more accurately than the other models. Fig. 9 shows two examples of trajectories predicted by each model in the basketball dataset. In the first case, the DSLR model predicted the movement of the red and blue players better than the dNRI, whereas the error of the pink player’s movement was larger with the DSLR. There was no significant difference between the two models in predicting the movements of the light blue and green players. In the second case, DSLR predicted most player movements better than dNRI. In both cases, NRI had a larger error than the other models. Overall, DSLR was able to predict the movements of players more accurately than the other models, which is congruent with the numerical results.

Fig. 10 visualizes the relation centrality inferred by our model in the two cases, where important relations with a relation centrality of 0.985 or more are represented in red lines; here, the thicker the line is, the higher the relation centrality is.
centrality. In the first case, DSLR selected only the relationship between the red and blue players as the most important relationship. In the second case, the DSLR method judged the relationship between the light blue and the pink players as being the most important. In addition, the DSLR also considered the relationships between the green and light blue players, the pink and the blue players, and the blue and red players as important. Because there is no ground-truth of the relationship between players, it is difficult to determine whether the relation centrality inferred by the DSLR is correct; however, it is highly likely to be a meaningful interpretation because the DSLR best predicted the future path based on the inferred relation.

Fig. 3(j) shows a visualization of the relation states between players in a two-dimensional relation latent space. The less important blue relation states are placed vertically across the middle of the latent space, and the important red relation states are placed obliquely across the right side of the latent space.

D. ABLATION STUDY

1) Random Sampling Trick

We propose the “random sampling trick” in Section III-C to prevent the model from becoming a compression model when training the DSLR. The “w/o RST” row in Table 5 shows the relational reasoning results of the DSLR trained without applying a random sampling trick. The results of the DSLR trained without a random sampling trick were worse for all datasets than those trained with a random sampling trick. In particular, when the number of relations is large, the performance is significantly improved when a random sampling trick is applied.

2) Relation Standard Deviation Loss

The relation standard deviation loss improves the performance of the relational inference by inducing the relation states to have similar values when they represent the same relation. As can be seen in the “w/o RSDL” row in Table 5, the DSLR trained with the relation standard deviation loss estimated the relations more accurately than the DSLR trained without it.

V. CONCLUSION

In this paper, we propose a DSLR model that can infer the relation between entities that have various relations with each other without supervision and predict the future state of such entities. Experimental results using physically simulated data show that the DSLR can analyze the system regardless of the number of relations and infer the number of relations in the system. In addition, in experiments using both motion capture data and basketball data, the DSLR can be better applied to real-world data with complex relations between entities. DSLR can also reflect the sparsity prior or analyze relation centrality.

DSLR can only model the interacting system in which the relations between entities are static over time. However, there may be cases in the real world in which the relationship between entities changes over time. Future studies can be conducted to extend the DSLR for application to systems with such dynamic relations.

APPENDIX A IMPLEMENTATION DETAILS

In this section, we describe the implementation details of DSLR. The source code is available online: https://github.com/dlehgo14/DSLR.

A. RELATION ENCODER

In the relation encoder, the function \( F \) in (1) which infers the edge state between two nodes is implemented using the gated recurrent unit (GRU) [41]. We used a 4-layer GRU, and the calculation process in the \( i \)-th GRU block at time \( t \) is as follows:

\[
z^i_t = \sigma (W^z_x x_t + U^z_s s^i_{t-1} + b^z),
\]

\[
e^i_t = \sigma (W^e_x x_t + U^e_s s^i_{t-1} + b^e),
\]

\[
h^i_t = \tanh (W^h_x x_t + U^h_s s^i_{t-1} + b^h),
\]

\[
\tilde{z}^i_t = \sigma (W^\tilde{z}_x x_t + U^\tilde{z}_s s^i_{t-1} + b^\tilde{z}),
\]

\[
\tilde{e}^i_t = \sigma (W^\tilde{e}_x x_t + U^\tilde{e}_s s^i_{t-1} + b^\tilde{e}),
\]

\[
\tilde{h}^i_t = \tanh (W^\tilde{h}_x x_t + U^\tilde{h}_s s^i_{t-1} + b^\tilde{h}),
\]

\[
z_t = \tilde{z}^i_t \odot \tilde{z}^i_t + \odot
\]

\[
e_t = \tilde{e}^i_t \odot \tilde{e}^i_t + \odot
\]

\[
h_t = \tilde{h}^i_t \odot \tilde{h}^i_t + \odot
\]

\[
a_t = W^a_x x_t + U^a_s h_t + b^a
\]

\[
y_t = \sigma (W^y_x x_t + U^y_s h_t + b^y)
\]
\[ r^i_t = \sigma(W^s_t x_t + U^s_t s^i_{t-1} + b^s_t), \quad (14) \]
\[ s^i_t = \tanh(W^h_t x_t + U^h_t (r^i_t \ast s^i_{t-1}) + b^h_t), \quad (15) \]
\[ s^i_t = (1 - z^i_t) \ast s^i_{t-1} + z^i_t \ast \hat{s}^i_t, \quad (16) \]

where \( W, U, \) and \( b \) represent the parameters of the GRU, \( x_t \) is the input vector, and \( s_t \) is the output vector. In addition, \( z_t \) is the update gate vector, \( r_t \) is the reset gate vector, and \( h_t \) is the candidate activation vector. Moreover, \( \sigma \) denotes the sigmoid function, and \( \ast \) denotes the element-wise product. The input of the first layer \( x^1_t \) is the concatenated vector of the two node states, and the input of the \( l \)-th layer \( x^l_t \) for \( l \geq 2 \) is the output of the previous layer \( s^i_{t-1} \). The initial edge state \( s^i_{-1} \) is initialized as a zero vector. The output vector of the last layer is the edge state. There are four dimensions for state of the nodes representing the position and velocity in a 2-D space. In motion capture data, there are six dimensions of the state of the nodes because the joints move in three-dimensional space. The number of dimensions of \( x^1_t \) is twice the number of dimensions of the node state, as the concatenated vector of the two node states. The number of dimensions of the edge state, which is the output of the GRU, was set to 128.

The \( G, H \) function in (2)–(4) that infers the relation state and relation centrality from the edge state is composed of multi-layer perceptrons with skip connections. Because \( G \) uses the reparameterization trick, it consists of two parallel networks that output the vectors of the mean and standard deviation of the relation state respectively, which are implemented as a 4-layer MLP. We use ReLU as the activation function and a linear function only for the last layer. In addition, \( H \) is also composed of a 4-layer MLP, with 1 to 3 layers shared with the mean network of \( G \), and one last layer attached. The number of input dimensions of the MLP is 128, which is the number of dimensions of the edge state, and the number of dimensions of the hidden layers is set to 196. The number of dimensions of the relation state, which is the output of \( G \), is set to 10, and the relation centrality, which is the output of \( H \), is a 1-dimensional scalar value.

### B. RELATION DECODER

In the relation decoder, \( K \) in (4), which calculates the influence \( f \) exerted by one node on another node, is implemented as a 4-layer MLP with skip connections such as \( G, K \). The input of \( K \) is a concatenated vector of the states of the two nodes, the relation states between the nodes, and the number of dimensions of the hidden layer is 196. The number of dimensions of the output, which is the influence \( f \), was set to 100. Next, to aggregate all influences received by the \( i \)-th node, we sum all influences in an element-wise manner.

In (9), \( L \), which calculates the amount of change in the state of the nodes over time, is implemented in an MLP with skip connections such as \( G, K \). The input of the network is the state of the node concatenated with the aggregated influence, and the output is the amount of change in the state of the node, which has the same number of dimensions as the state of the node.

**REFERENCES**

[1] Frank Harary and Gopal Gupta. Dynamic graph models. Mathematical and Computer Modelling, 25(7):79–87, 1997.
[2] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In Int. Conf. Machine Learning, pages 2688–2697, 2018.
[3] Ezra Webb, Ben Duy, Helena Andres-Terre, and Pietro Liò. Factorised neural relational inference for multi-classification systems. arXiv preprint arXiv:1905.08721, 2019.
[4] Yaguang Li, Chuizheng Meng, Cyrus Shahabi, and Yan Liu. Structure-informed graph auto-encoder for relational inference and simulation. In Int. Conf. Machine Learning Workshop on Learning and Reasoning with Graph-Structured Representations, 2019.
[5] Colin Graber and Alexander G Schwing. Dynamic neural relational inference. In Computer Vision and Pattern Recognition, pages 8513–8522, 2020.
[6] Franco Scarselli, Sweah Liang Yong, Marco Gori, Markus Hagenbuchner, Ah Chung Tsoi, and Marco Maggini. Graph neural networks for ranking web pages. In The 2005 IEEE/WIC/ACM Int. Conf. Web Intelligence (WI’05), pages 666–672. IEEE, 2005.
[7] F. Scarselli, M. Gori, A. C. Tsii, M. Hagenbuchner, and G. Monfardini. The graph neural network model. IEEE Trans. Neural Networks, 20(1):61–80, 2009.
[8] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. Computational capabilities of graph neural networks. IEEE Trans. Neural Networks, 20(1):81–102, 2008.
[9] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. In Yoshua Bengio and Yann LeCun, editors, Int. Conf. Learning Representations, 2016.
[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Computer Vision and Pattern Recognition, pages 770–778, 2016.
[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.
[12] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. nature, 529(7587):484, 2016.
[13] Shahriar Rahman Fahim, Dristi Datta, MD. Rafiqul Islam Sheikh, Sanjay Dey, Yeahia Sarker, Subrata K. Sarker, Faisal R. Badal, and Saajal K. Das. A visual analytic in deep learning approach to eye movement for human-machine interaction based on inertia measurement. IEEE Access, 8:45924–45937, 2020.
[14] Peter W Battaglia, Jessica B Hamrick, Victor Bapt, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinskowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261, 2018.
[15] Sainbayar Sukhbaatar, arthur szlam, and Rob Fergus. Learning multi-agent communication with backpropagation. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 29, pages 2244–2252. Curran Associates, Inc., 2016.
[16] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. In Advances in Neural Information Processing Systems, pages 4502–4510, 2016.
[17] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinskowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In Advances in Neural Information Processing Systems, volume 30, pages 4967–4976. Curran Associates, Inc., 2017.
[18] Yedid Hoshen. Vain: Attentional multi-agent predictive modeling. In Advances in Neural Information Processing Systems, pages 2701–2711, 2017.
[19] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In Int. Conf. on Learning Representations, 2018.
[20] Myeongjin Kim, Taehun Kim, and Dajin Kim. Spatio-temporal slowfast self-attention network for action recognition. In IEEE Int. Conf. Image Processing, pages 2206–2210, 2020.
[21] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In Int. Conf. on Learning Representations, 2018.
