Universiteit Leiden

Master Computer Science

A GNN-Based Method for Group Detection from Spatio-Temporal Data

Name: Zhizhou Fang
Student ID: 2289164
Date: 07/07/2022

Specialisation: Advanced Data Analysis

1st supervisor: Mitra Baratchi
2nd supervisor: Gwenn Englebienne
Advisors: Maedeh Nasri, Shenghui Wang

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Group detection from spatio-temporal data is helpful in many applications, such as automatic driving and social sciences. Most previous works in this domain are based on conventional machine learning methods with feature engineering; only a few works are based on deep learning. We proposed a graph neural network (GNN) based method for group detection. Our method is an extension of neural relational inference (NRI) [1]. We made the following changes to the original NRI: (1) We applied symmetric edge features with symmetric edge updating processes to output symmetric edge representations corresponding to the symmetric binary group relationships. (2) Inspired by Wavenet [2], we applied a gated dilated residual causal convolutional block to capture both short and long dependency of the sequences of edge features. We name our method "WavenetNRI". Our experiments compare our method with several baselines, including the original NRI on two types of data sets: (1) six spring simulation data sets; (2) five pedestrian data sets. Experimental results show that on the spring simulation data sets, NRI and WavenetNRI with supervised training outperform all other baselines, and NRI performs slightly better than WavenetNRI. On the pedestrian data sets, our method WavenetNRI with supervised training outperforms other pairwise classification-based baselines. However, it cannot compete against the clustering-based methods. In the ablation study, we study the effects of our changes to NRI on group detection. We find that on the spring simulation data sets, the gated dilated residual causal convolutional block can slightly improve the performance of NRI. On the pedestrian data sets, the symmetric edge features with the symmetric edge updating processes can significantly improve the performance of NRI.

Contents

Chapter 1

Introduction

Group detection from spatio-temporal data finds groups of agents based on their spatial features at multiple time steps. For instance, group detection can be applied to social science and psychology areas. For example, in schools, detecting groups of pupils playing on a playground can help teachers find out which pupils prefer group activities and which prefer solitary activities; this can help to analyze pupils' personalities.

Groups may have different definitions in different applications and situations. E.g., we can define pedestrians walking together and sharing a common destination on streets as a group. We can also define pupils who are involved in the same activity at the same time as a group. Therefore, formalising a general definition of groups and describing the standard underlying features that determine groups is difficult. In our work, we focus on detecting groups in two different situations: pedestrians and spring simulations. In the pedestrian case, groups are defined as the pedestrians walking close to each other and sharing the same destination. The spring simulation is used to simulate a playground of particles. The groups of particles are randomly generalised; the particles in the same group have a high probability of interacting; in contrast, the particles in different groups have a low probability of interacting.

Most previous works of group detection rely on conventional machine learning methods with feature engineering, such as [3, 4]; the usage of deep learning in group detection is relatively unexplored. Feature engineering often requires domain knowledge and is limited to specific data types and applications, limiting the generalisation of the methods. Since graph neural networks (GNNs) show strong potential for relational reasoning [5]; i.e., GNNs can model agents as nodes and their relationships as edges, we believe that group detection can benefit from GNNs. A recent work [6] proposes a method that applies a graph convolutional network (GCN) to detect conversational groups. A conversational group refers to a group of agents involved in the same conversation; the agents in the conversational groups are static, and this work does not consider the movements of the agents. Since we focus on detecting groups of moving agents, this work is not within the scope of our research. Our work is the first to apply GNN

to spatio-temporal data to detect groups to the best of our knowledge. Kipf et al [1] proposed a GNN-based method Neural Relational Inference (NRI), which applies a GNN to infer the interactions between particles given their spatio-temporal sequences in a physical system. The differences between the inference of pairwise interactions in a physical system and group detection are: (1) The interactions in a physical system are constant through the given time window. In contrast, the interactions in group detection tasks can change over time. (2) The group relationships are transitive, meaning that a group member does not need to interact with all other group members. (3) The group relationships are imbalanced; the number of pairs in a group is relatively small [7]. We made the following changes to the original NRI to tackle these differences. (1) The single-layer 1D convolutional layer in NRI is replaced with a gated dilated residual causal convolutional block, proposed by Wavenet [2]. We expect this architecture to help capture both short and long dependence in the group detection tasks where the interactions change over time. (2) We apply the Louvain community detection algorithm to transform the pairwise interactions into clusters denoting groups. (3) The weighted cross-entropy loss function is applied to deal with the imbalanced group relationships. The original NRI builds and updates edge features by simply concatenating the node features, which does not satisfy the symmetric property of group relationships. We use symmetric temporal edge features and symmetric edge updating to tackle this problem.

Our contributions are:

- We proposed a framework for group detection based on NRI. We extended NRI by applying the Louvain community detection algorithm to transform the predicted interactions into predicted groups.
- We evaluated the framework using six spring simulation data sets and five pedestrian data sets. We compared our method with the following baselines Yamaguchi et al [4], Solera et al [3], GD-GAN [8] and the original NRI.
- We verified the effects of our changes to the original NRI by ablation study. The results showed that the gated dilated residual causal convolutional block could improve the performance of the spring simulation data sets. The symmetric temporal edge features with symmetric edge updating processes can improve the performance of the pedestrian data sets.

The rest of this thesis is organised as follows. In Chapter 2 we will formalise the problem. In Chapter 3 we will discuss the related works. In Chapter 4 we will present our methods. In Chapter 5 the experiments will be discussed. In Chapter 6 we will make conclusions and plan future work.

Chapter 2

Problem Formulation

The group detection for spatio-temporal data can be defined as follows: given N agents in a time window with a duration of T time steps, the measurement of one agent $i \in 1, ..., N$ at a time step $t \in 1, ..., T$ is denoted as X_i^t. The spatial-temporal sequences of all agents can be denoted as $X_{1:N}^{1:T}$. The goal is to detect the existing groups $C = \{c_j | j = 1, ..., K\}$ of these agents, where $K \leq N$ is the number of the groups. We assume that the group relationships are constant in a time window. The problem is formulated as a pairwise binary classification with a community detection algorithm. A GNN encoder is applied to predict the pairwise interactions $\hat{I}$ between agents of the time window given $X_{1:N}^{1:T}$, i.e., $P(\hat{I}|X)$. The Louvain community detection algorithm is applied to transform the predicted pairwise interactions $\hat{I}$ into predicted groups $\hat{C}$. In our work, we will try to train the GNN encoder in both supervised and unsupervised ways. In the supervised way, we will use the ground truth pairwise group relationships G as labels to train the GNN encoder. $G_{(i,j)} = 1$ means agent i and agent j are in the same group while $G_{(i,j)} = 0$ means agent i and agent j are in different groups. The difference between G and $\hat{I}$ will be minimised during training. In unsupervised learning, we assume that the movement of an agent i is influenced by its group members with high probability and by the outsiders with low probability. The movement of the agent i at time step t can be modelled as the increments $\Delta X_i^t = X_i^{t+1} - X_i^t$. A GNN decoder will predict ΔX^t given the predicted interactions $\hat{I}$ to reconstruct $X_{1:N}^{1:T}$, i.e., $P(\hat{X}_{1:N}^{1:T}|\hat{I})$; here, the interactions refer to the influences on movements of the agents. By joint training the GNN encoder and GNN decoder, i.e., minimising the differences between $X_{1:N}^{1:T}$ and $\hat{X}_{1:N}^{1:T}$, the GNN encoder can predict the pairwise influences $\hat{I}$.

Chapter 3

Related work

3.1 Group Detection

In this section, we discuss the related works for detecting groups of moving agents. We found that most previous works, such as [4, 3], are based on conventional machine learning methods with hand-crafted features. Creating hand-crafted features needs domain knowledge, and these hand-crafted features usually depend on particular data types and applications. E.g., the feature engineering applied to coordinate data may not be suitable for velocity and acceleration data. The features created for detecting pedestrians walking in groups on streets may not apply to detecting children playing in groups on a playground. Yamaguchi et al [4] proposed an SVM-based framework applying normalised histograms of distances, velocity and direction features to classify the binary group detection. Solera et al [3] proposed a structural SVM [9] framework, which finds groups of pedestrians by applying supervised clustering based on hand-crafted features such as distance, motion causality, trajectory shape and paths convergence. Zhao et al [10] extended the work of Solera et al [3] by adding more hand-crafted features such as velocity and orientation. Chen et al [11] proposed a framework using graph clustering algorithm to detect groups of indoor activities, where the edge features denoting the similarity between agents, which is based on Feature Engineering of acceleration, audio and location features from sensors. One deep learning-based work is GD-GAN [8], which applies a LSTM-based generator to predict future trajectories. The groups can be detected by clustering the hidden states of the generator.

We categorise the related works of group detection into two categories: (1) *Clustering-based* and (2) *Pairwise classification-based*. Most previous works are clustering-based, such as [3, 8, 10, 11], which apply unsupervised or supervised clustering methods to output the clusters denoting groups. The Pairwise classification-based methods do not predict the groups directly; they predict the pairwise binary group relationships or interactions between agents. One example of Pairwise classification-based method is the work of Yamaguchi et al [4].

Our work uses a GNN encoder to predict the pairwise interactions; therefore, our work belongs to the pairwise classification-based methods. The main advantage of the pairwise classification-based methods is their simplicity. I.e., the models can be trained directly with the ground truth group relationships without special optimisation algorithms, such as the Block-coordinate Frank-Wolfe (BCFW) algorithm.

3.2 Graph Neural Networks for Spatio-Temporal Data

This section will discuss the related works using GNN for Spatio-temporal data. Based on the application of the GNN, we categorise the related works into two categories: (1) *Edge-centric* and (2) *Node-centric*.

Edge-centric works focus on predicting the edges, which often model the edges as categorical variables. The edges can denote the interaction or relation types between nodes. Examples of edge-centric works include the encoder part of NRI [1], which applies a GNN-encoder to predict the interaction types between particles in a physical system, and TrafficGraphNet [12], which predicts the interaction types between traffic actors.

Node-centric works focus on learning the representations of nodes by aggregating the neighbourhood of the nodes. Examples of node-centric works include the work of Kipf et al [13] and GraphSAGE [14], which use GNN to generate node representations for node classification tasks; the decoder part of NRI aggregates types of incoming interactions predict the trajectories of the particles. STGAT [15], Social-STGCN [16] and GraphTCN [17] use GNN to aggregate the neighbourhood context into the representation of nodes, which denote pedestrians, and then predict the future trajectories of the pedestrians. The final representation of nodes will usually be applied to downstream tasks such as node classification, node clustering and node states prediction.

Most GNN-based works for Spatio-temporal data focus on forecasting tasks, such as TrafficGraphNet [12], STGAT [15], Social-STGCN [16] and GraphTCN [17]. These works are node-centric and do not directly model the pairwise interactions or group relationships of agents. The edge representations cannot indicate group relationships, making them unsuitable for group detection tasks. In our work, we extended the encoder part of NRI, which predicts the interactions between particles, for group detection; the details will be discussed in Chapter 4. Our work is the first to apply GNN on Spatio-Temporal Sequences to the best of our knowledge to detect groups.

Chapter 4

Methodology

In this chapter, we will discuss the architecture of our method. We will first discuss the approach to model the interactions between agents with a GNN encoder. Then we will discuss the methods to train the GNN encoder, including supervised and unsupervised training methods. Since the GNN encoder is based on NRI [1] and we applied a gated Residual Dilated Causal Convolutional Block, proposed by the architecture of Wavenet [2], we name our model "WavenetNRI". The visualisation of the training approaches of our model is shown in Figure 4.1.

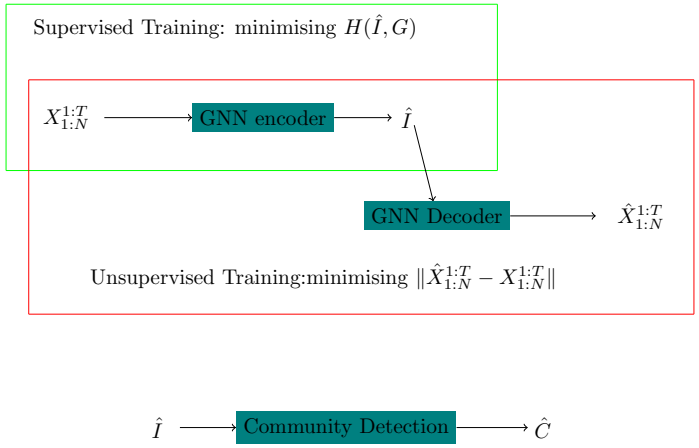


Figure 4.1: Diagram of our proposed model (WavenetNRI). The GNN encoder takes the spatio-temporal sequences $X_{1:N}^{1:T}$ as input and outputs predicted pairwise interactions $\hat{I}$. In the supervised training method (shown in the light green block), the cross-entropy loss between the predicted interactions $\hat{I}$ and ground truth pairwise group relationships labels G, which is denoted by $H(\hat{I}, G)$, will be minimised. In the unsupervised training method (shown in the red block), a GNN decoder will take the predicted interactions $\hat{I}$ to reconstruct the spatio-temporal sequences $X_{1:N}^{1:T}$. The GNN encoder and GNN decoder will be trained jointly. After training, a community detection algorithm will be applied to transform the pairwise predicted interaction $\hat{I}$ to predicted clusters $\hat{C}$ denoting groups of the agents.

4.1 GNN Encoder: Interactions Modelling

In this section, we will present our approach to modelling the interactions between agents. Our method is based on NRI [1]. The core part of our method is a GNN encoder, which predicts the distribution of the interaction and non-interaction edges. The encoder can be trained in a supervised way, i.e., we will train the encoder with ground truth group relationships by minimising the binary classification loss function. In an unsupervised way, i.e., we will jointly train the GNN encoder and a GNN decoder, which reconstructs the spatio-temporal sequences based on predicted interactions.

The $X_{1:N}^{1:T}$ denotes spatio-temporal sequences of the all N agents from time steps 1 to T. The spatio-temporal sequence of agent i is denoted as $X_i = [X_i^1, ..., X_i^T]$. The measurement of an agent i at one time step t is denoted as X_i^t. The initial edge feature of the agents i and j at one time step t is denoted as $e_{(i,j)}^t$. The sequence of the edge features of the agents i and j of from the time step 1 to the time step T is denoted as $e_{(i,j)}^{1:T}$. A neural network will be applied to the sequence $e_{(i,j)}^{1:T}$ to get a vector representation $h_{(i,j)}^1$. The vector representation $h_{(i,j)}^1$ will be passed to node and edge updating functions to get the final edge representation $h_{(i,j)}^2$, which denotes the logits of the two edge types, where $h_{(i,j),1}^2$ denotes the logit of the non-interaction edge type and $h_{(i,j),2}^2$ denotes the logit of the interaction edge type.

In NRI, the initial edge features and edge updating are implemented by concatenating the features of the end nodes. which is shown in Equation 4.1 and Equation 4.3, respectively. $[\cdot, \cdot]$ denotes concatenation. f_e denotes edge updating function. h_j^1 denotes the node representation of the agent j. h_j^1 computed by passing the aggregated coming edges $\sum_{i \neq j} h_{(i,j)}^1$ of node j to a node updating function f_v, which is shown in Equation 4.2. The edge and node updating functions f_e and f_v are multilayer perceptrons (MLPs).

$$e_{(i,j)}^t = [X_i^t, X_j^t](t \in 1, ..., T) \tag{4.1}$$

$$h_j^1 = f_v(\sum_{i \neq j} h_{(i,j)}^1) \tag{4.2}$$

$$h_{(i,j)}^2 = f_e([h_i^1, h_j^1]) \tag{4.3}$$

NRI applies a 1D convolutional layer to transform the edge sequence $e_{(i,j)}^{1:T}$ into the edge vector representation $h_{1,(i,j)}$, which is shown in Equation 4.4.

$$h_{(i,j)}^1 = f_{CNN}(e_{(i,j)}^{1:T}) \tag{4.4}$$

The function f_{CNN} consists of a 1D convolutional layer with attentive pooling. The architecture of f_{CNN} is explained in Figure 4.2.

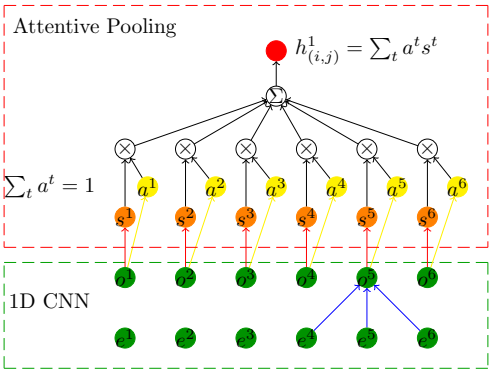


Figure 4.2: 1D Convolutional layer with Attentive Pooling. The 1D convolutional layer is shown in the green dashed line block. The attentive pooling is shown in the red dashed line block. The edge index (i, j) is ignored in the figure for clarity. The sequence of edges features $e^{1:T}$ will be fed into a 1D convolutional layer with kernel size=3, denoted by the blue arrows in the green dashed line block, to get hidden states $o^{1:T}$. The hidden states $o^{1:T}$ will be fed into two 1D CNNs f_{pred} and f_{score} with kernel size=1 separately, denoted by the red and yellow lines in the red dashed line block, respectively. f_{pred} predicts the edge representation s^t; f_{score} predicts the attention score a^t, where $\sum_t a^t = 1$. The edge representation is $h^1_{(i,j)} = \sum_t a^t s^t$. The number of time steps is six here.

There are several limitations in the GNN encoder of the original NRI: (1) Building and updating edge features and representations by simply concatenating the node features (shown in Equation 4.1,4.3) cannot explicitly model the spatial differences of agents, and the results are not symmetric, which may not satisfy the symmetric group relationships. (2) Using only one convolutional (shown in Figure 4.4) layer cannot capture the long-term interactions of the sequences of edge features. To tackle these limitations, we made the following changes to the original NRI:

- We include the spatial differences between agents and temporal increments in the initial temporal edge features $e^t_{(i,j)}$, and update the edge features by element-wise product of the end nodes' representations. In this way, the final edge vector representations $h^2_{(i,j)}$ are symmetric and can capture both spatial differences between the agents and their movements.
- We replace the single 1D convolutional layer in NRI with a gated dilated residual causal convolutional Block to learn the temporary edge features. Compared with the 1D convolutional layer in NRI, we expect this architecture can capture both short and long-term interactions of the sequences of edge features.

4.1.1 Symmetric edge features and updating

In our work, we construct the edge features by concatenating the spatial differences of the node measurements and the temporal increments, i.e., movements

of each node, shown in Equation 4.5. The Euclidean distances between agent measurements are used to model the spatial difference between agent i and agent j, which is denoted by $\|X_i^t - X_j^t\|$; the temporal increments, i.e., the movements are modelled by element-wise production of the increments of the two agents, which is denoted by $\Delta X_i^t \odot \Delta X_j^t$, where $\Delta X_i^t = X_i^{t+1} - X_i^t$. In this way, the temporal edge $e_{(i,j)}^t$ captures the spatial difference between agent i and agent j as well as the temporal increments of the nodes. Another benefit is that the edge features are symmetric, i.e., $e_{(i,j)}^t = e_{(j,i)}^t$, corresponding to the symmetric properties of the pairwise group relationships.

$$e_{(i,j)}^t = [\|X_i^t - X_j^t\|, \Delta X_i^t \odot \Delta X_j^t](t \in 1, ..., T-1) \tag{4.5}$$

The edge sequences $e_{(i,j)}^{1:T}$ are passed to a gated residual dilated causal convolutional block to get the vector representations of edges, denoted by $h_{(i,j)}^1$. The details will be discussed in Section 4.1.2.

For a node j, the vector representation $h_{(i,j)}^1$ of incoming edges will be aggregated and fed to a node updating function f_v to get higher level node representation h_j^1 of the node j, which is the same as the node updating process in NRI, shown in Equation 4.2. These node representations will be combined by element-wise production and fed to another neural network f_e to get final edge representations $h_{(i,j)}^2$, which represents the logits of categorical distributions of edges, shown in Equation 4.7. Through this process, the final edge representation $h_{(i,j)}^2$ not only capture the interaction between node i and node j, but also the interactions of node i and node j with other nodes [1].

The GNN encoder can be trained in both supervised and unsupervised ways, which will be discussed in Section 4.2. After training, a community detection algorithm can be applied to the interaction graphs to find clusters denoting groups. In our work, we choose the Louvain community detection algorithm [18].

$$h_j^1 = f_v(\sum_{i \neq j} h_{(i,j)}^1) \tag{4.6}$$

$$h_{(i,j)}^2 = f_e([h_{(i,j)}^1, h_i^1 \odot h_j^1]) \tag{4.7}$$

4.1.2 Gated dilated residual causal convolutional block

Instead of using a single convolutional layer like NRI (shown in Equation 4.4), we apply a gated residual dilated causal convolutional block to transform the edge sequences $e_{(i,j)}^{1:T}$ into the vector representation $h_{(i,j)}^1$, shown in Equation 4.8. The architecture is proposed by Wavenet [2] to learn the raw audio data, and is also used in other works for spatio-temporal data, such as [17].

$$h_{(i,j)}^1 = f_{WavenetCNN}(e_{(i,j)}^{1:T-1}) \tag{4.8}$$

The causal convolution preserves the order of the edge sequences by using features from past time steps. With dilated convolutional kernels, the receptive

fields can be expanded exponentially by staking convolutional layers [2]. The skip connection, a 1D CNN, solves the gradient vanishing problem when we attempt to increase the number of layers [19]. The gating activation function regulates the information flow and performs significantly better than rectified linear activation (ReLU) [2]. The gating activation is described in Equation 4.9. l is the layer index. W_l^1 and W_l^2 are two different learnable 1D-convolution parameters of the layer l; e_l denotes the hidden states of edge features of the layer l. $*$ denotes the convolutional operation. σ and $\odot$ denote sigmoid function and element-wise multiplication, respectively. A 1D convolutional layer with attentive pooling over all timesteps is applied to get the vector representations of the edges $h_{(i,j)}^1$. The visualisation of this process is shown in Figure 4.3.

$$e_{l+1} = tanh(W_l^1 * e_l) \odot \sigma(W_l^2 * e_l) \tag{4.9}$$

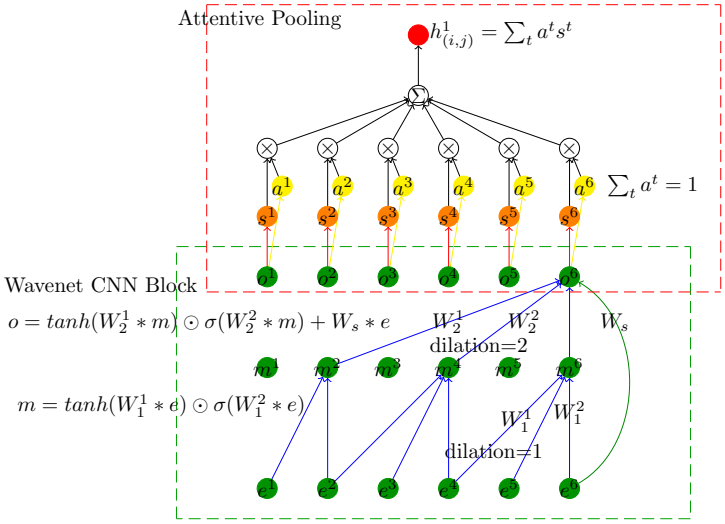


Figure 4.3: 1D gated residual dilated causal CNN block with Attentive Pooling. The CNN block consists of two convolutional layers, shown in the green dashed line block. The attentive pooling is shown in the red dashed line block. The edge index (i, j) is ignored in the Figure for clarity. The sequence of edges features $e^{1:T}$ will be fed into a gated dilated 1D CNN block with skip connections to get hidden states $o^{1:T}$. W_l^1 and W_l^2 denote two different learnable convolutional parameters with kernel size=3 of the layer l, shown by the blue arrows. The dilation of the first convolutional layer is 1, and that of the second layer is 2. W_s denotes the skip connection, which is a 1D CNN with kernel size=1, shown by the green arrow. The hidden states $o^{1:T}$ will be fed into two 1D CNNs f_{pred} and f_{score} with kernel size=1 separately. f_{pred} predicts the edge representation s^t, shown by the red arrows; f_{score} predicts the attention score a^t, shown by the yellow arrows, where $\sum_t a^t = 1$. The edge representation is $h_{(i,j)}^1 = \sum_t a^t s^t$. The number of time steps is six here.

4.2 Training Methods

In this section, we will discuss the training methods for the GNN encoder. The GNN encoder can be trained by both the supervised and unsupervised methods.

4.2.1 Supervised Training

In supervised training, the ground truth pairwise group relationships $G_{i,j}$ will be used as labels; i.e., $G_{(i,j)} = 1$ denotes agent i and agent j are in the same group while $G_{(i,j)} = 0$ denotes agent i and agent j are not in the same group. Due to the imbalanced distribution of the labels, the weighted cross-entropy is used as a loss function, where the rare labels are assigned higher weights. The weight of each label is computed based on the label distribution on the training data set. We use w_G to denote the weight of the group label and $w_{\bar{G}}$ to denote the weight of the non-group label. The calculation of the weights are shown in Equation 4.10 and Equation 4.11, where n_G and $n_{\bar{G}}$ denotes the number of group labels and non-group labels in the training dataset, respectively.

$$w_G = \frac{n_G + n_{\bar{G}}}{2n_G} \tag{4.10}$$

$$w_{\bar{G}} = \frac{n_G + n_{\bar{G}}}{2n_{\bar{G}}} \tag{4.11}$$

The predicted distribution of interaction types is denoted as $\hat{I}_{(i,j)}$ and described in Equation 4.12. $\hat{I}_{(i,j),1}$ denotes the probability of non-interaction while $\hat{I}_{(i,j),2}$ denotes the probability of interaction.

$$\hat{I}_{(i,j)} = softmax(h^2_{(i,j)}) \tag{4.12}$$

The weighted cross-entropy $H(\hat{I}, G)$ is described in Equation 4.13. By minimising the weighted cross-entropy, we expect the encoder to detect the interactions between the agents of the same group while ignoring the interactions between the agents of different groups.

$$H(\hat{I}, G) = -\sum_{(i,j)} [w_G G_{(i,j)} log(I_{(i,j),2}) + w_{\bar{G}}(1 - G_{(i,j)}) log(I_{(i,j),1})] \tag{4.13}$$

4.2.2 Unsupervised Training

In the unsupervised training, we will jointly train the GNN encoder with a GNN decoder as a variational autoencoder (VAE), the same as NRI. The GNN decoder will take the sampled predicted interactions $z_{(i,j)}$ as input and reconstruct the spatio-temporal sequences $X^{1:T}_{1:N}$; we used Gumbel Softmax [20] to sample the logits $h^2_{(i,j)}$, which is described in Equation 4.14, where g denotes the Gumbel noise and τ denotes the temperature. The temperature τ controls

the "smoothness" of the sampling; the sampled results converge to one-hot distributions when $\tau \to 0$. We follow the original NRI and set τ to 0.5. More details about Gumbel Softmax can be found at [20].

$$z_{(i,j)} = softmax((h^2_{(i,j)+g})/\tau) \tag{4.14}$$

We used the same GNN decoder introduced in NRI, in which the separate neural networks are used for each edge type. This idea is also used by some works using interactions to predict the trajectories of traffic actors, such as TrafficGraphNet [12] and the work of Lee et al [21], where the idea of using separate edge networks for each edge type is named "Typed Graph Network".

Since we focus on detecting groups of agents, the interaction edges and non-interaction edges need to be explicitly distinguished. In our work, the sampled latent variable $z_{(i,j)}$ is a 2-dimensional variable where $z_{(i,j),1}$ denotes the non-interaction edge type and $z_{(i,j),2}$ denotes the interaction edge type where $z_{(i,j),1} > 0$, $z_{(i,j),2} > 0$ and $z_{(i,j),1} + z_{(i,j),2} = 1$. The non-interaction edge type $z_{(i,j),1}$ will be ignored in the "Typed Graph Network" of the decoder, i.e., we only consider using $z_{(i,j),2}$ to predict the replacement of the agents. This process is formulated in Equation 4.15 and Equation 4.16; $\tilde{f}_e$ and $\tilde{f}_v$ denote the edge network and node network in the decoder respectively. $\tilde{h}^t_{(i,j)}$ denotes the edge representation of the edge (i, j).

$$\tilde{h}^t_{(i,j)} = z_{(i,j),2}\tilde{f}_e([X^t_i, X^t_j]) \tag{4.15}$$

$$\hat{X}^{t+1}_j = X^t_j + \tilde{f}_v(\sum_{i \neq j} \tilde{h}^t_{(i,j)}) \tag{4.16}$$

The loss function consists of two parts: the reconstruction error and the Kullback-Leibler (KL) divergence of the predicted distribution of edge types $\hat{I}_{(i,j)}$ and a prior distribution of edge types, where $\hat{I}_{(i,j),k}$ denotes the probability of the kth edge type. The computation of $\hat{I}_{i,j}$ is described in Equation 4.12. We use a uniform prior for the distribution of the edge types; the KL divergence is the negative entropy of distribution $\hat{I}_{(i,j)}$ added with a constant $Klog(K)$ where K denotes the number of edge types. In our work, $K = 2$ since there are two edge types and we ignore the constant $Klog(K)$ in the loss function. The loss function is described in Equation 4.17.

$$loss = \sum_i \sum_{t=2}^{T} \frac{\|X^t_i - \hat{X}^t_i\|}{2\sigma^2} + \sum_i \sum_{k=1}^{2} \hat{I}_{(i,j),k} log(\hat{I}_{(i,j),k}) \tag{4.17}$$

By minimising the loss function, we expect the encoder to detect the interactions that influence agents' movements. The assumption is that the movements of the agents are influenced by their group members with high probability and influenced by the outsiders with low probability; i.e., the influences on movements can indicate the group relationships.

Chapter 5

Experiments

We are interested in answering the following questions with our experiments:

1. Can symmetric edge features with the symmetric edge updating processes improve the performance for group detection with supervised training?
2. Can the gated residual dilated causal convolution block improve the performance for group detection with supervised training?
3. How is the ranking of our method compared with other methods?
4. How do the predicted interactions relate to the ground truth group relationships for the supervised pairwise classification-based methods?
5. In unsupervised training, to what extent can the influences between the agents on their movements indicate their group relationships?

5.1 Datasets

In this section, we will discuss the data sets in our experiments.

5.1.1 Spring Simulation

Kipf et al [1] proposed a spring simulation, where a number of particles are put in 2D box and randomly connected by springs, which are used to simulate the interactions between the particles. We extended the spring simulation of Kipf et al [1] by defining groups of particles. In our the spring simulation; there are $N \in \{5, 10\}$ particles in a 10×10 2D box, which can be regarded as a "playground" of particles. The initial measurements, i.e., locations and velocities of the particles and the relations between them, i.e., interactions or non-interactions will be randomly generated at the beginning of the simulation. We assume that a particle interacts with its group members with high probability and interacts with outsiders with low probability. We use $G_{(i,j)}$ to denote

the group relationship between particle v_i and v_j. $G_{(i,j)} = 1$ if v_i and v_j are in the same group otherwise $G_{(i,j)} = 0$. The interaction between v_i and v_j is denoted by $I_{(i,j)}$. The probability that v_i and v_j interact with each other given their group relation $G_{(i,j)}$ is given by Equation 5.1.

$$P(I_{(i,j)} = 1|G_{(i,j)} = 1 - exp(-a(G_{(i,j)} + b))(a > 0, b > 0) \tag{5.1}$$

The movements of the particles are determined by Hooke's law $F_{(i,j)} = -k(r_i - r_j)I_{(i,j)}$ where $F_{(i,j)}$ is the force from particle v_j to particle v_i; k is the spring constant and r_i is the location vector of particle v_i; $I_{(i,j)}$ is an indicator function; $I_{(i,j)} = 1$ if there is interaction between v_i and v_j, i.e., there is a spring connnecting v_i and v_j; otherwise $I_{(i,j)} = 0$.

The values of a and b control the probabilities of Group Interaction and Non-Group Interaction. The value of a controls the overall magnitude of the probabilities, and the value of b has great impact on the non-group interaction probability. According to Equation 5.1, the group interactions probabilities are always greater than non-group interaction probabilities. During simulation, the pairwise interactions between particles will be dynamically evaluated at each time step according to Equation 5.1. Since the group relations are symmetric and transitive, i.e., if $G_{(i,j)} = 1$, then $G_{(j,i)} = 1$ and if $G_{(i,j)} = 1 \wedge G_{(j,k)} = 1$, then $G_{(j,k)} = 1$; it is not reasonable to independently initialize the pairwise group relations G. Instead, we randomly initialize a group assignment matrix GA at the beginning of each simulation. $GA_{(i,k)} = 1$ if particle v_i belongs to group k otherwise $GA_{(i,k)} = 0$. The group relations G can be derived from the group assignment matrix GA; i.e. if $GA_{(i,k)=1} \wedge GA_{(j,k)=1}$, then $G_{(i,j)} = 1$.

In our experiments, we use $N \in \{5, 10\}$ particles. The value of a is set to 3 to create big group interaction probability ($\geq 95.3\%$). The values of b vary from 0.02 to 0.05, and the corresponding non-group interaction probabilities range from 5.8% to 13.9%. The six data sets are described in Table 5.1.

Dataset	num of particles	a	b	Group Interact	Non-Group Interact
sim1	5	3	0.02	95.3%	5.8%
sim2	8	3	0.02	95.3%	5.8%
sim3	10	3	0.02	95.3%	5.8%
sim4	10	3	0.03	95.4%	8.6%
sim5	10	3	0.04	95.6%	11.3%
sim6	10	3	0.05	95.7%	13.9%

Table 5.1: Five simulation data sets. From left to right, the columns denote the name of the dataset, the number of particles, the value of a, the value of b, and the probability of group interaction and non-group Interaction, respectively.

We study the effect of non-group interaction probability on the performance of our model by comparing the results of the data sets *sim3* to *sim6*. By comparing the results of *sim1* and *sim3*, we can study the effect of the number of agents on the performance of group detection performance. By comparing

the results of *sim3* to *sim6*, we can study the effect of the non-group interaction probabilities. Every data set has 2500 simulations, and the duration of each simulation is 20 seconds corresponding to 50 time steps. In our experiments, 60% of the examples will be randomly chosen for training; 20% will be randomly chosen as validation, and the left 20% are testing set. The input features of these data sets are the locations and velocities of the particles. The code to generate the spring simulation data sets can be found in this Github repository[1].

5.1.2 Pedestrian Datasets

We select 5 public pedestrian datasets, namely *zara01*, *zara02* and *students03* from Lerner et al [22], *BIWI ETH* and *BIWI Hotel* from Pellegrini et al [23]. These data sets can be found at OpenTraj[2]. We will use the sequences of annotated locations of the pedestrians, i.e., the trajectories, as input features and try to detect pedestrians walking in groups. The duration, the number of pedestrians and groups of these pedestrian data sets are listed in Table 5.2.

To evaluate the performance of different methods, in each data set 60% of the time steps will be used for training, and the remaining time steps will be used for validation and test. We manually created six types of training/validation/test splits, which is explained in Figure 5.1. A sliding window with six seconds, i.e., fifteen time steps is applied in the training, validation and test period to create examples; the number of pedestrians in each example can be different.

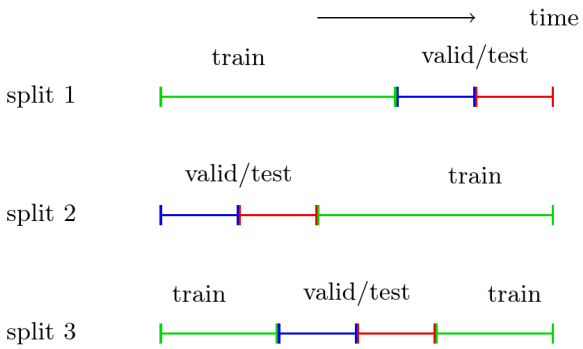


Figure 5.1: The training period, validation period and test period are denoted by green, blue and red line, respectively. We can create six splits for each pedestrian data set by exchanging the validation and test period.

5.1.3 Data Exploration

In this section, we will explore the two types of data sets. We will compare the pairwise sequence features from the same groups and those from different groups, i.e., whether these features can be used to distinguish the group relationships

[1] https://github.com/fatcatZF/WavenetNRI
[2] https://github.com/crowdbotp/OpenTraj

Dataset	Duration(s)	Pedestrians	Groups
zara01	360.4	148	45
zara02	420.4	204	58
students03	215.6	428	104
BIWI ETH	713.4	360	65
BIWI Hotel	722.4	389	41

Table 5.2: The columns from left to right denote the name, duration, the number of pedestrians and groups of the pedestrian data sets.

and non-group relationships. We will compute the binary group relationship distributions, i.e., the imbalance of the group and non-group labels.

Sequence Features

We randomly select examples from the pedestrian and spring simulation data sets. The visualisations of the trajectories of pedestrian data sets and spring simulation data sets are shown in Figure 7.1 and Figure 7.2 of Appendix 7.1.1, respectively.

As shown in Figure 7.1, the trajectories of the same groups have similar shapes and the same directions, i.e., the group members have similar movement patterns; the agents try to keep small distances from their group members.

As shown in Figure 7.2, the trajectories of the spring simulation are more complex compared with the trajectories of the pedestrians. The trajectories of the same group can have different shapes and directions; i.e., the group members can have different movement patterns. The distances between the group members can rapidly change over time.

To quantify the differences between the sequences of pedestrian data sets and the spring simulation data sets, we use dynamic time warping (DTW) [24] to compute the dissimilarity between sequences. Higher distances indicate higher dissimilarity between sequences. The DTW distances are listed in Table 5.3. We also compute Euclidean distances between agents, listed in Table 5.4.

According to Table 5.3, in the pedestrian data sets, the average DTW distances between group members are much lower than those between pedestrians from different groups. I.e., the sequences of agents are much more similar to their group members' sequences than outsiders' sequences. Therefore, the DTW distances can help distinguish group and non-group relationships. Compared with the pedestrian data sets, the differences between the DTW distances of group members' sequences and those between different groups are much smaller in the spring simulation data sets, as shown by the fourth columns of Table 5.3a and Table 5.3b. Therefore, the DTW distances may not help detect group relationships for the spring simulation data sets. According to Table 5.3b, with the increase in the number of particles and the non-group interaction probability, the differences between the average DTW distances of the same groups and that of different groups become more negligible, as shown in Table 5.3b.

Data set	Ave.dtw(G)	Ave.dtw(NG)	G/NG
zara01	3.658	18.522	0.198
zara02	3.082	16.828	0.183
students03	3.520	23.667	0.149
ETH	4.026	19.305	0.209
Hotel	2.710	17.517	0.155

(a) Average DTW distances of pedestrian data sets.

Data set	Ave.dtw(G)	Ave.dtw(NG)	G/NG
sim1	7.349	13.698	0.536
sim2	7.104	11.667	0.609
sim3	7.018	11.385	0.616
sim4	6.834	9.784	0.699
sim5	6.783	8.486	0.799
sim6	6.763	8.156	0.829

(b) Average DTW distances of spring simulation data sets.

Table 5.3: Table 5.3a presents the average DTW distances of the pedestrian data sets. Table 5.3b presents the average DTW distances of the spring simulation data sets. The columns from left to right denote the names of the data sets, average DTW distances between group members, average DTW distances between agents from different groups, and the ratios of the average group DTW distance to average non-group distances.

Data set	Ave.dist(G)	Ave.dist(NG)	G/NG
zara01	1.002	4.678	0.210
zara02	0.827	4.236	0.195
students03	0.952	6.038	0.158
ETH	1.140	4.977	0.229
Hotel	0.709	4.439	0.160

(a) Average distances of pedestrian data sets.

Data set	Avg.dist(G)	Avg.dist(NG)	G/NG
sim1	1.096	2.236	0.490
sim2	1.053	1.796	0.586
sim3	1.042	1.706	0.611
sim4	1.010	1.501	0.673
sim5	0.992	1.319	0.752
sim6	0.980	1.233	0.795

(b) Average distances of spring simulation data sets.

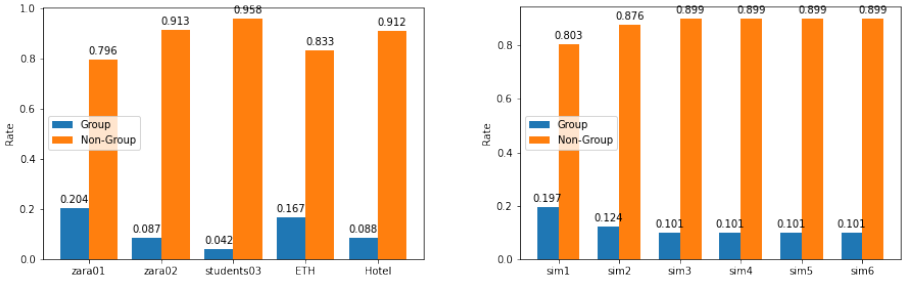
Table 5.4: Table 5.4a presents the average euclidean distances of the pedestrian data sets. Table 5.4b presents the average euclidean distances of the spring simulation data sets. The columns from left to right denote the names of the data sets, average distances between group members, average distances between agents from different groups, and the ratios of the average group distance to average non-group distances.

According to Table 5.4, in the pedestrian data sets, the average Euclidean distances between group members are much lower than those between pedestrians from different groups. I.e., the pedestrians are closer to their group members than outsiders. Therefore, the Euclidean distances can help detect group relationships. Compared with the pedestrian data sets, the differences between the euclidean distances of the same groups and that of different groups are insignificant in the spring simulation data sets. Therefore, the Euclidean distances do not help detect group relationships. With the increase in the number of particles and the non-group probability, the differences between the average distances of group members and those from different groups become more negligible, as shown in Table 5.4b.

Distributions of Group Relationships

The visualisation of the distributions of the pairwise group relationship labels and the pairwise non-group relationship labels is shown in the bar charts of Figure 5.2.

According to Figure 5.2, Both pedestrian data sets and spring simulation data sets are very imbalanced, i.e., the non-group labels account for the majority of the labels.



(a) Pedestrian **(b)** Spring Simulation

Figure 5.2: Visualisation of Pairwise Labels Distributions. Figure 5.2a shows the label distribution of the pedestrian data sets; Figure 5.2b shows the label distribution of the spring simulations. The blue bars denote the rates of the group labels and the orange bars denote the rates of the non-group labels.

5.2 Baselines

We will compare the results of our method with the following four baselines:

- Yamaguchi et al (2011) [4]: a linear SVM classifies the binary group relationships based on hand crafted histograms of distance, direction and velocity. The regularisation parameter C of the SVM is set to 10.
- Solera et al (2015) [3]: a structured SVM (SSVM) predict the clusters of the pedestrians based on distance, temporal causality, trajectory similarity and common goals from motion. The SSVM consists of a SVM predicting the pairwise similarities of the agents and a correlation clustering component predicting the clusters. This SVM is trained with the Block-Coordinate Frank Wolfe (BCFW) [25] algorithm. The regularisation parameter C of the SVM is set to 10.
- GD-GAN (2018) [8]: a LSTM-based GAN predicting the future trajectory of agents. The DBSCAN algorithm is applied to the hidden states of the LSTM to find the groups. The dimensions of hidden states are 256 in our experiment. The assumption of this baseline is that the agents in the same group have similar movement patterns.

- NRI (2018) [1]: to evaluate the effects of our changes to the original NRI, i.e., the symmetric edge features with symmetric edge updates and the gated residual dilated causal convolutional block; we extended the original NRI by applying the Louvain community detection algorithm to transform the predicted pairwise interactions to clusters denoting groups. The kernel size of the 1D convolutional layer is set to 5. The node updating and edge updating processes are implemented with multiple layer perceptrons (MLPs). The hidden dimension size of the MLPs is set to 256.

5.3 Evaluation Metric

We apply Group Mitre $\Delta_{GM}(C, \hat{C})$ proposed by Solera et al [3] to measure the quality of the predicted groups, where C and $\hat{C}$ are disjoint sets denoting the true groups and predicted groups respectively. One choice to measure the quality of group detection is pairwise loss $\Delta_{PW}(C, \hat{C})$, which is defined as the ratio between the number of pairs on which C and $\hat{C}$ disagree on their group membership and the number of all possible pairs of elements. There are two problems of the metric pairwise loss $\Delta_{PW}(C, \hat{C})$: (1) $\Delta_{PW}(C, \hat{C})$ only considers positive intra-group relations and neglects singletons[3][8], (2) because of the quadratic number of links that exist among agents, $\Delta_{PW}(C, \hat{C})$ tends to be imprecise when dealing with large number of agents[3]. Group Mitre $\Delta_{GM}(C, \hat{C})$ can overcome these two problems. Group Mitre $\Delta_{GM}(C, \hat{C})$ is an extension of the Mitre loss[26]. Mitre loss represents groups as spanning trees instead of complete graphs, which results in a linear amount of positive and negative links rather than the quadratic number of links in pairwise loss $\Delta_{PW}(C, \hat{C})$[26]. Group Mitre loss $\Delta_{GW}(C, \hat{C})$ solves the problem of neglecting singletons by adding each agent α_i a fake counterpart α_i'; the agent α_i will be in the same group with its fake agent α_i' if α_i is a singleton. The computation of recall of Group Mitre $\Delta_{GM}(C, \hat{C})$ is shown in Algorithm 1; to compute precision of Δ_{GM}, we can simply exchange C and $\hat{C}$ in Algorithm 1.

5.4 Implementation Details

In our experiments, we set the kernel size of the gated residual dilated causal convolutional block to five, which is explained in Equation 4.9 and Equation 4.8. The node and edge functions are multiple layer perceptrons (MLPs). The hidden dimension size of the node functions of Equation 4.6,4.16 and the edge functions of Equation 4.7,4.15 is set to 256. The stochastic gradient descent (SGD) with momentum is applied for optimisation. The momentum is set to 0.9. In each experiment, the number of training epochs is 200. The code to implement WavenetNRI can be found in this Github repository[3].

[3]https://github.com/fatcatZF/WavenetNRI

5.5 Results

In this section, we will discuss the results of the experiments. For each spring simulation data set, every method will be evaluated fifteen times. For each pedestrian data set, every method will be evaluated three times for each type of training/validation/test splits. I.e., every method will have eighteen experimental results on each pedestrian data set. Therefore, each method has ninety experimental results on the six spring simulation data sets and the five pedestrian data sets. To compare the methods on multiple data sets, Friedman test is applied to check the significant differences between rank means and Nemenyi post-hoc test is then applied to check the significant pairwise differences in average ranks. The rankings of the methods are visualised by critical differences (CD) diagrams [27] with a significance level of 0.05. More details about CD diagrams can be found in the work of Demsar et al [27]. In the tables and figures, we use the text "un" in brackets to denote unsupervised training of WavenetNRI and NRI, respectively.

We will first discuss the ablation study, which studies the effects of our changes to the original NRI, corresponding to the research questions 1 and 2. Then we will compare our method WavenetNRI with other baselines, corresponding to the research question 3. We will compare the confusion matrices of the supervised pairwise classification-based methods, corresponding to research question 4. Finally, we will discuss the unsupervised training of NRI and WavenetNRI, corresponding to the research question 5.

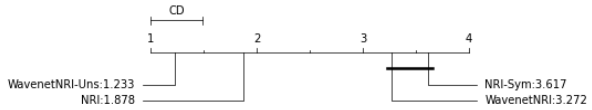
5.5.1 Ablation Study

This section will study the effects of our changes to the original NRI. To test the effects of the symmetric edge features and symmetric edge updating process, we use a method applying the same 1D convolutional as the original NRI with the symmetric edge features and the symmetric edge updating process; we name this method "NRI-Sym". To test the effects of the gated dilated residual causal convolutional block, we use another method named "Wavenet-Uns", which applies the gated dilated residual causal convolutional block with the same edge features and edge updating process as the original NRI. We will compare these two methods with our method WavenetNRI and the original NRI on the spring simulation data sets and the pedestrian data sets. The results of the spring simulation and pedestrian data sets are listed in Table 5.5 and Table 5.6, respectively. The corresponding CD diagrams of the mean rankings are shown in Figure 5.3 and Figure 5.4, respectively. All the methods are trained supervised.

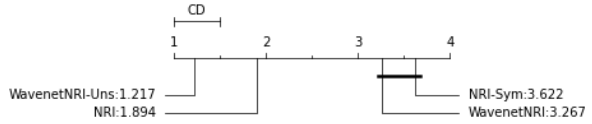
	sim1		*sim2*		*sim3*		*sim4*		*sim5*		*sim6*	
	R	P	R	P	R	P	R	P	R	P	R	P
NRI	0.995 ±0.002	0.994 ±0.003	0.992 ±0.005	0.988 ±0.005	0.997 ±0.002	0.994 ±0.002	0.998 ±0.001	0.994 ±0.002	0.997 ±0.003	0.995 0.003	**0.998** ±0.001	0.996 ±0.001
NRI-Sym	0.990 ±0.004	0.987 ±0.006	0.982 ±0.007	0.971 ±0.010	0.981 ±0.007	0.964 ±0.013	0.976 ±0.011	0.960 ±0.014	0.981 ±0.007	0.966 ±0.009	0.981 ±0.007	0.961 ±0.009
Wavenet NRI-Uns	**0.998** ±0.002	**0.997** ±0.001	**0.997** ±0.003	**0.995** ±0.005	**0.999** ±0.001	**0.997** ±0.002	**0.999** ±0.001	**0.997** ±0.001	**0.999** ±0.001	**0.997** ±0.001	**0.998** ±0.001	**0.997** ±0.001
Wavenet NRI	0.990 ±0.010	0.988 ±0.013	0.987 ±0.004	0.976 ±0.007	0.985 ±0.005	0.970 ±0.010	0.986 ±0.006	0.965 ±0.012	0.983 ±0.005	0.970 ±0.008	0.986 ±0.004	0.972 ±0.007

Table 5.5: Ablation study results of spring simulation data sets. The columns denote recall (R) and precision (P) based on Group Mitre Δ_{GW}. The best average values of recall and precision are highlighted with bold text.

Spring simulation data sets



(a) CD diagram of recall based on group mitre Δ_{GW} of different methods on the spring simulation data sets.

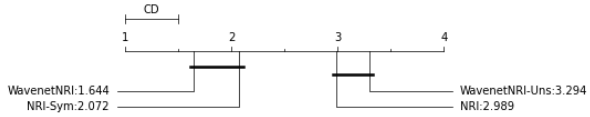


(b) CD diagram of precision based on group mitre Δ_{GW} of different methods on the spring simulation data sets.

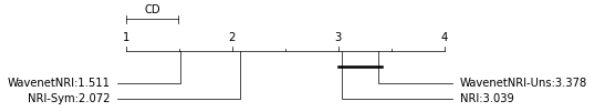
Figure 5.3: CD diagrams of different methods on the spring simulation data sets. Figure 5.3a represents the average ranking of different methods of recall based on group mitre Δ_{GW}; Figure 5.3b represents the ranking of precision based on group mitre Δ_{GW}. The numbers in the diagrams denote the mean ranks (lower means better). The average ranks with non-significant difference are connected with a horizontal line.

According to Figure 5.3 and the results listed in Table 5.5, the method WavenetNRI-Uns performs slightly better than NRI and the performance of NRI-Sym is lower than NRI. Therefore, the gated dilated residual causal convolutional block can slightly improve the performance of NRI on the spring simulation data sets, and the symmetric edges and the symmetric edge updating process have negative effects on the original NRI.

Pedestrian data sets



(a) CD diagram of recall based on group mitre Δ_{GW} of different methods on the pedestrian data sets.



(b) CD diagram of precision based on group mitre Δ_{GW} of different methods on the pedestrian data sets.

Figure 5.4: CD diagrams of different methods on the pedestrian data sets. Figure 5.4a represents the average ranking of different methods of recall based on group mitre Δ_{GW}; Figure 5.4b represents the ranking of precision based on group mitre Δ_{GW}. The numbers in the diagrams denote the mean ranks (lower means better). The average ranks with non-significant difference are connected with a horizontal line.

	zara01		*zara02*		*students03*		*ETH*		*Hotel*	
	R	P	R	P	R	P	R	P	R	P
NRI	0.801 ±0.096	0.737 ±0.108	0.720 ±0.050	0.673 ±0.078	0.610 ±0.048	0.469 ±0.046	0.663 ±0.083	0.669 ±0.080	0.577 ±0.122	0.565 ±0.122
NRI-Sym	0.851 ±0.093	0.813 ±0.091	0.780 ±0.081	0.749 ±0.079	**0.761** ±0.045	**0.704** ±0.061	0.679 ±0.094	0.686 ±0.096	0.708 ±0.121	0.739 ±0.115
Wavenet NRI-Uns	0.719 ±0.138	0.625 ±0.165	0.718 ±0.059	0.658 ±0.106	0.622 ±0.053	0.492 ±0.051	0.542 ±0.146	0.530 ±0.147	0.566 ±0.169	0.554 ±0.163
Wavenet NRI	**0.893** ±0.090	**0.900** ±0.107	**0.804** ±0.051	**0.776** ±0.061	0.722 ±0.050	0.650 ±0.049	**0.793** ±0.078	**0.815** ±0.079	**0.748** ±0.106	**0.790** ±0.086

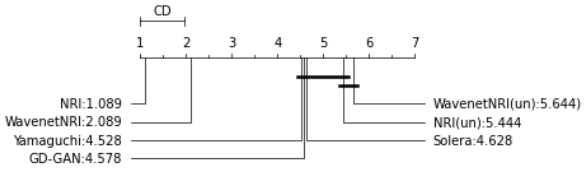
Table 5.6: Ablation study results of pedestrian data sets. The columns denote recall (R) and precision (P) based on Group Mitre Δ_{GW}. The best average values of recall and precision are highlighted with bold text.

According to Figure 5.4 and the results listed in Table 5.6, the method NRI-Sym performs better than the NRI and the method WavenetNRI-Uns has similar performance to NRI on the pedestrian data sets. Therefore, the symmetric edge features with the symmetric edge updating process can improve the performance of NRI on the pedestrian data sets, and the gated dilated residual causal convolutional block does not have significant effect on the performance of NRI on the pedestrian data sets.

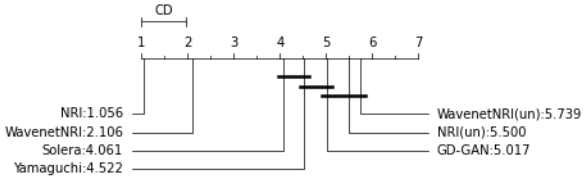
5.5.2 WaventNRI versus baselines

In this section we will compare our method WavenetNRI with the baselines. The results of the spring simulation data sets are listed in Table 5.7. The corresponding CD diagram is shown in Figure 5.5. The results of the pedestrian data sets are listed in Table 5.8. The corresponding CD diagram is shown in Figure 5.6.

Spring simulation data sets



(a) CD diagram of recall based on group mitre Δ_{GW} of different methods on the spring simulation data sets.



(b) CD diagram of precision based on group mitre Δ_{GW} of different methods on the spring simulation data sets.

Figure 5.5: CD diagrams of different methods on the spring simulation data sets. Figure 5.5a represents the ranking of different methods of recall based on group mitre Δ_{GW}; Figure 5.5b represents the ranking of precision based on group mitre Δ_{GW}. The numbers in the diagrams denote the mean ranks (lower means better). The mean ranks with a non-significant difference are connected with a horizontal line.

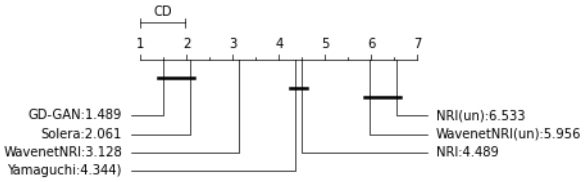
According to Table 5.7 and Figure 5.5, NRI and WavenetNRI outperform all other baselines, and NRI performs slightly better than WavenetNRI with supervised training. With unsupervised training, NRI and WavenetNRI have similar performances, which are lower than other baselines. According to the columns of the fifth and the seventh rows in Table 5.7, both recall and precision based on group mitre Δ_{GW} of NRI(un) and WavenetNRI(un) decrease from the data set *sim1* to *sim6*. Since the data sets *sim1*, *sim2* and *sim3* have the same probabilities of group and non-group interactions and the different number of particles. We notice the trend that the performances of NRI(un) and WavenetNRI(un) decrease with the increasing number of particles. The data sets *sim3*, *sim4*, *sim5* and *sim6* have the same number of particles and they have different probabilities of non-group interactions; the non-group interaction probability increases from *sim3* to *sim6*. We can notice the performances of NRI(un) and

	sim1		*sim2*		*sim3*		*sim4*		*sim5*		*sim6*	
	R	P	R	P	R	P	R	P	R	P	R	P
Yamaguchi et al	0.579 ±0.017	0.481 ±0.020	0.521 ±0.017	0.399 ±0.021	0.512 ±0.009	0.388 ±0.015	0.511 ±0.006	0.387 ±0.006	0.512 ±0.004	0.387 ±0.004	0.511 ±0.006	0.386 ±0.005
Solera et al	0.664 ±0.075	0.600 ±0.067	0.543 ±0.028	0.463 ±0.033	0.529 ±0.039	0.413 ±0.017	0.462 ±0.055	0.392 ±0.019	0.472 ±0.028	0.374 ±0.033	0.459 ±0.037	0.382 ±0.030
GD-GAN	0.531 ±0.003	0.430 ±0.004	0.525 ±0.005	0.401 ±0.009	0.514 ±0.003	0.383 ±0.004	0.511 ±0.002	0.382 ±0.004	0.511 ±0.003	0.381 ±0.004	0.512 ±0.003	0.383 ±0.004
NRI	**0.995** ±0.002	**0.994** ±0.003	**0.992** ±0.005	**0.988** ±0.005	**0.997** ±0.002	**0.994** ±0.002	**0.998** ±0.001	**0.994** ±0.002	**0.997** ±0.003	**0.995** ±0.003	**0.998** ±0.001	**0.996** ±0.001
NRI(un)	0.988 ±0.009	0.985 ±0.010	0.539 ±0.026	0.415 ±0.024	0.431 ±0.013	0.259 ±0.015	0.344 ±0.015	0.174 ±0.008	0.318 ±0.010	0.151 ±0.006	0.290 ±0.014	0.138 ±0.005
Wavenet-NRI	0.990 ±0.010	0.988 ±0.013	0.987 ±0.004	0.976 ±0.007	0.985 ±0.005	0.970 ±0.010	0.986 ±0.006	0.965 ±0.012	0.983 ±0.005	0.970 ±0.008	0.986 ±0.004	0.972 ±0.007
Wavenet-NRI(un)	0.985 ±0.010	0.981 ±0.011	0.553 ±0.097	0.452 ±0.100	0.401 ±0.048	0.242 ±0.032	0.331 ±0.019	0.169 ±0.006	0.298 ±0.022	0.138 ±0.012	0.277 ±0.016	0.131 ±0.011

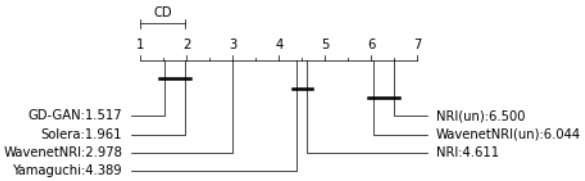
Table 5.7: Experimental results of spring simulation data sets. The columns denote recall (R) and precision (P) based on Group Mitre Δ_{GW} of the baselines and our method (WavenetNRI) on spring simulation data sets. The text "un" in the bracket of NRI and WavenetNRI denotes unsupervised training. The best average values of recall and precision are highlighted with bold text.

WavenetNRI(un) decrease with increasing non-group interaction probability.

Pedestrian data sets



(a) CD diagram of recall based on group mitre Δ_{GW} of different methods on the pedestrian data sets.



(b) CD diagram of precision based on group mitre Δ_{GW} of different methods on the pedestrian data sets.

Figure 5.6: CD diagrams of different methods on the pedestrian data sets. Figure 5.6a represents the ranking of different methods of recall based on group mitre Δ_{GW}; Figure 5.6b represents the ranking of precision based on group mitre Δ_{GW}. The numbers in the diagrams denote the mean ranks (lower means better). The mean ranks with a non-significant difference are connected with a horizontal line.

According to Table 5.8 and Figure 5.6, the baseline GD-GAN and Solera et al outperform all other methods in both recall and precision of group mitre Δ_{GW}. With supervised training, our method WavenetNRI outperforms the original NRI, and the baseline Yamaguchi et al. With unsupervised training, WavenetNRI and NRI rank lower than other baselines.

5.5.3 Confusion matrices of the supervised pairwise classification methods

In this section, we will compare the confusion matrices of our method WavenetNRI with other two supervised pairwise classification-based baselines: Yamaguchi et al and NRI. Since the pariwise classification-based methods predict the pairwise interactions, we are interested in how the predicted interactions relate to the ground truth group relationships. The more the predicted interactions relate to the ground truth group relationships, the more likely the community detection algorithms can produce good clustering results based on the predicted interactions. We define the following measurements: true negative rate (tn), false positive rate (fp), true positive rate (tp) and false negative rate (fn). We compute the average tn, fp, tp and fn on all the test data sets and visualise the results with confusion matrices.

	zara01		*zara02*		*students03*		*ETH*		*Hotel*	
	R	P	R	P	R	P	R	P	R	P
Yamaguchi et al	0.889 ±0.076	0.879 ±0.077	0.555 ±0.117	0.443 ±0.145	0.512 ±0.052	0.404 ±0.056	0.745 ±0.067	0.746 ±0.087	0.833 ±0.072	0.841 ±0.068
Solera et al	0.893 ±0.026	0.906 ±0.033	**0.879** ±0.037	**0.876** ±0.037	0.805 ±0.085	0.798 ±0.112	0.887 ±0.027	0.911 ±0.021	0.925 ±0.024	0.927 ±0.030
GD-GAN	**0.949** ±0.046	**0.934** ±0.051	0.850 ±0.077	0.838 ±0.084	**0.857** ±0.019	**0.832** ±0.032	**0.931** ±0.037	**0.950** ±0.028	**0.925** ±0.084	**0.944** ±0.058
NRI	0.801 ±0.096	0.737 ±0.108	0.720 ±0.050	0.673 ±0.078	0.610 ±0.048	0.469 ±0.046	0.663 ±0.083	0.669 ±0.080	0.577 ±0.122	0.565 ±0.122
NRI(un)	0.509 ±0.120	0.398 ±0.106	0.511 ±0.102	0.373 ±0.086	0.512 ±0.048	0.353 ±0.051	0.436 ±0.136	0.406 ±0.125	0.357 ±0.112	0.349 ±0.124
Wavenet-NRI	0.893 ±0.090	0.900 ±0.107	0.804 ±0.051	0.776 ±0.061	0.722 ±0.050	0.650 ±0.049	0.793 ±0.078	0.815 ±0.079	0.748 ±0.106	0.790 ±0.086
Wavenet-NRI(un)	0.567 ±0.131	0.480 ±0.144	0.543 ±0.097	0.423 ±0.079	0.541 ±0.064	0.374 ±0.068	0.513 ±0.171	0.495 ±0.172	0.502 ±0.092	0.497 ±0.097

Table 5.8: Experimental results of pedestrian data sets. The columns denote the recall (R) and precision (P) based on group mitre Δ_{GW} of the baselines and our method (WavenetNRI) on pedestrian data sets. The text "un" in the brackets of NRI and WavenetNRI denotes unsupervised training. The best average values of recall and precision are highlighted with bold text.

The true negative rate (tn) is defined as the ratio of the predicted negative interactions, which are equal to the corresponding ground truth non-group relationships to the total negative ground truth group relationships.

$$tn = \frac{\sum_{(i,j)} \mathbf{1}_{\hat{I}_{(i,j)}=0 \wedge \hat{I}_{(i,j)}=G_{(i,j)}}}{\sum_{(i,j)} \mathbf{1}_{G_{(i,j)}=0}} \tag{5.2}$$

The false positive rate (fp) is defined as:

$$fp = 1 - tn = 1 - \frac{\sum_{(i,j)} \mathbf{1}_{\hat{I}_{(i,j)}=0 \wedge \hat{I}_{(i,j)}=G_{(i,j)}}}{\sum_{(i,j)} \mathbf{1}_{G_{(i,j)}=0}} \tag{5.3}$$

The true positive rate (tp) is defined as the ratio of the predicted positive interactions, which are equal to the corresponding ground truth group relationships to the total positive ground truth group relationships.

$$tp = \frac{\sum_{(i,j)} \mathbf{1}_{\hat{I}_{(i,j)}=1 \wedge \hat{I}_{(i,j)}=G_{(i,j)}}}{\sum_{(i,j)} \mathbf{1}_{G_{(i,j)}=1}} \tag{5.4}$$

The false negative rate (fn) is defined as:

$$fn = 1 - tp = 1 - \frac{\sum_{(i,j)} \mathbf{1}_{\hat{I}_{(i,j)}=1 \wedge \hat{I}_{(i,j)}=G_{(i,j)}}}{\sum_{(i,j)} \mathbf{1}_{G_{(i,j)}=1}} \tag{5.5}$$

The layout of the confusion matrices is explained in Table 5.9.

	$\hat{I} = 0$	$\hat{I} = 1$
$G = 0$	tn	fp
$G = 1$	fn	tp

Table 5.9: The layout of confusion matrices. The rows denote the ground truth group relationships. The columns denote the predicted interactions. The diagonal elements denote the ratio of the predicted interactions equal to the ground truth relationships to the total given ground truth relationships.

The higher the values of tns and tps are, the more the predicted interactions $\hat{I}$ relate the corresponding ground truth group relationships G. High tn values indicate that high proportion of the true non-group relationships ($G = 0$) can be covered by the predicted non-interaction edges ($\hat{I} = 0$). High tp values indicate that high proportion of the true positive group relationships ($G = 1$) can be covered by the predicted interaction edges ($\hat{I} = 1$).

Spring simulation data sets

The measurements tn, fp, fn and tp of the supervised pairwise classification-based methods on the spring simulation data sets are listed in Table 5.10. The corresponding confusion matrices are shown in Figure 7.3 of Appendix 7.1.2.

	sim1		*sim2*		*sim3*		*sim4*		*sim5*		*sim6*	
	tn	tp	tn	tp	tn	tp	tn	tp	tn	tp	tn	tp
Yamaguchi et al	**0.999**	0.001	**1**	0.001	**1**	0	**1**	0	**1**	0	**1**	0
NRI	0.997	**0.997**	0.996	**0.994**	0.998	**0.998**	0.999	**0.995**	0.999	**0.998**	0.998	**0.995**
WavenetNRI	0.996	**0.997**	0.993	0.991	0.993	0.993	0.995	0.991	0.996	0.992	0.989	0.981

Table 5.10: Average true negative rate (tn) and true positive rate (tp) of the methods Yamaguchi et al, NRI and WavenetNRI on the spring simulation data sets. The highest values in each column are highlighted by bold texts.

According to Table 5.10 and Figure 7.3, the baseline Yamaguchi et al has high true negative rates (tn) on all the spring simulation data sets. However, the true positive rates (tp) are low (≈ 0), suggesting that the baseline Yamaguchi et al can hardly retrieve positive group relationships. Both NRI and WavenetNRI have high true negative rates and true positive rates on all the spring simulation data sets. The true negative rates and true positive rates of NRI are slightly higher than WavenetNRI on most spring simulation data sets.

Pedestrian data sets

The measurements tn, fp, fn and tp of the supervised pairwise classification-based methods on the pedestrian data sets are listed in Table 5.11. The corre-

sponding confusion matrices are shown in Figure 7.4 of Appendix 7.1.2.

	zara01		*zara02*		*students03*		*ETH*		*Hotel*	
	tn	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*
Yamaguchi et al	**0.939**	0.855	**0.977**	0.189	**0.974**	0.337	**0.86**	0.761	**0.947**	0.904
NRI	0.739	0.818	0.823	0.839	0.823	0.948	0.771	0.618	0.727	0.829
WavenetNRI	0.885	**0.915**	0.883	**0.999**	0.914	**0.97**	0.853	**0.938**	0.91	**0.987**

Table 5.11: Average true negative rate (*tn*) and true positive rate (*tp*) of the methods Yamaguchi et al, NRI and WavenetNRI on the pedestrian data sets. The highest values in each column are highlighted by bold texts.

According to Table 5.11 and Figure 7.4, the baseline Yamaguchi et al has higher true negative rates than NRI and WavenetNRI on all the pedestrian data sets. The WavenetNRI has higher true positive rates than Yamaguchi et al and NRI. The true negative rates of WavenetNRI are higher than NRI.

5.5.4 Unsupervised Training of NRI & WavenetNRI

This section will discuss the unsupervised training of NRI and WavenetNRI. We will compare the true negative rates (*tn*) and true positive rates (*tp*) of the unsupervised trained methods NRI(un) and WavenetNRI(un) with their corresponding supervised trained methods. The unsupervised training of these two methods is based on the assumption that the influences between the agents on their movements can indicate their group relationships. We will validate this assumption on the spring simulation and pedestrian data sets. To validate this assumption, we plot the mean squared error (MSE) between the ground truth sequences $X_{1:N}^{1:T}$ and the reconstructed sequences $\hat{X}_{1:N}^{1:T}$ and the edge accuracy during the training process. Since the GNN encoder predicts the pairwise influences on the movements and the GNN decoder reconstructs the spatio-temporal sequences $X_{1:N}^{1:T}$ given the predicted influences, the MSE between $X_{1:N}^{1:T}$ and $\hat{X}_{1:N}^{1:T}$ can indicate whether the GNN encoder correctly predicts the pairwise influences. We define edge accuracy as the ratio of the number of predicted edges equal to the corresponding ground truth pairwise group relationships to the total edges. Suppose there are N agents; the number of total edges is $N(N-1)$. The edge accuracy is defined as:

$$acc = \frac{\sum_{(i,j)} \mathbf{1}_{\hat{I}_{(i,j)}=G_{(i,j)}}}{N(N-1)} \tag{5.6}$$

Spring simulation data sets

The true negative rate, false positive rate, false negative rate and true positive rate of the method NRI and WavenetNRI with unsupervised training on the spring simulation data sets are listed in Table 5.12. The corresponding confusion matrices are shown in Figure 7.5 of Appendix 7.1.2.

	sim1		*sim2*		*sim3*		*sim4*		*sim5*		*sim6*	
	tn	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*
NRI(un)	0.991	**0.986**	0.716	**0.982**	0.651	**0.982**	0.535	**0.987**	**0.469**	**0.986**	**0.413**	**0.988**
Wavenet NRI(un)	**0.995**	0.971	**0.773**	0.978	**0.662**	0.973	**0.543**	0.974	0.45	0.971	0.392	0.969

Table 5.12: Average true negative rate (*tn*) and true positive rate (*tp*) of the methods NRI and WavenetNRI with the unsupervised training on the spring simulation data sets. The highest values in each column are highlighted by bold texts.

According to Table 5.12 and Figure 7.5, the true positive rates of NRI and WavenetNRI with unsupervised training on the spring simulation data sets are close to their corresponding supervised training results, listed in Table 5.10. The true negative rates of NRI and WavenetNRI decrease from *sim1* to *sim6*, which shows that the true negative rates of these two methods with unsupervised training decrease with the increasing of the number of agents and the value of non-group interaction probability.

The average MSE plots and the average edge accuracy of the spring simulation data sets are shown in Figure 7.7 and Figure 7.8 of Appendix 7.1.3, respectively. According to Figure 7.7, the MSEs of NRI and WavenetNRI on all the spring simulation data sets decrease during training, which suggests that the GNN encoder correctly predicts the pairwise influences. The corresponding edge accuracy of the data sets *sim1*, *sim2* and *sim3* increase during training (shown in Figure 7.8a to Figure 7.8c), which suggests that the pairwise influences on the movements indicate the pairwise group relationships. However, the indication becomes weaker with the increase of the number of particles since the edge accuracy will converge at lower values. With the increase of the non-group probability, the corresponding edge accuracy will stop increasing and converge at lower values (shown in Figure 7.8d to Figure 7.8f), which indicates that when non-group interaction probability increases, the influences' indication of the group relationships become weaker.

Pedestrian data sets

The true negative rate, false positive rate, false negative rate and true positive rate of the method NRI and WavenetNRI with unsupervised training on the pedestrian data sets are listed in Table 5.13. The corresponding confusion matrices are shown in Figure 7.6 of Appendix 7.1.2.

According to Table 5.13 and Figure 7.6, the true negative rates and true positive rates of NRI and WavenetNRI with unsupervised training on the pedestrian data sets are much smaller than their corresponding supervised training results, listed in Table 5.11.

The average MSE plots and the average edge accuracy of the pedestrian data sets are shown in Figure 7.9 and Figure 7.10 of Appendix 7.1.3, respectively. According to Figure 7.9, the average MSEs of NRI and WavenetNRI on all he

	zara01		*zara02*		*students03*		*ETH*		*Hotel*	
	tn	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*	*tn*	*tp*
NRI(un)	**0.597**	0.386	**0.62**	0.447	0.62	0.263	0.498	**0.507**	0.501	0.432
Wavenet NRI(un)	0.576	**0.606**	0.487	**0.944**	**0.622**	**0.471**	**0.643**	0.473	**0.557**	**0.459**

Table 5.13: Average true negative rate (*tn*) and true positive rate (*tp*) of the methods NRI and WavenetNRI with the unsupervised training on the pedestrian data sets. The highest values in each column are highlighted by bold texts.

pedestrian data sets decrease, which indicates the GNN encoders of NRI and WavenetNRI can correctly predict the pairwise influences on their movements. However, the corresponding edge accuracy on all the pedestrian data sets fluctuates around low values (shown in Figure 7.10a to Figure 7.10e). Therefore, the influences on the movements cannot reflect the group relationships in the pedestrian data sets.

5.6 Discussion

In this section, we will first compare the results of the spring simulation and the pedestrian data sets. Then we will discuss the problems of pairwise classification-based methods. Finally, we will discuss the limitations of our work.

5.6.1 Comparison between the two types of data sets

According to Table 5.7 and Table 5.8, the baselines GD-GAN [8] and Solera et al [3] perform much better on the pedestrian data sets than on the spring simulation data sets. GD-GAN [8] is based on the assumption that the group members have similar movement patterns, which is invalid in the spring simulation data sets; therefore, GD-GAN [8] can not correctly predict the groups on the spring simulation data sets. The baseline solera et al [3] is based on the hand-crafted features, which are designed for pedestrians and not useful for the spring simulation data sets.

On the other hand, NRI and WavenetNRI with the supervised training perform much better on the spring simulation data sets than on the pedestrian data sets. With unsupervised training, NRI and WavenetNRI can retrieve the most positive group relationships on the spring simulation data sets (shown in Table 5.12 and Figure 7.5). However, in the pedestrian case, NRI and WavenetNRI have low true positive rates (*tp*) (shown in Table 5.13 and Figure 7.6). I.e., with unsupervised training, NRI and WavenetNRI cannot retrieve most positive group relationships on the pedestrian data sets. The reason for these phenomenons is the oversimplified interaction patterns on the spring simulation data sets. In the spring simulation data sets, the agents have a high probability of interactions with all their group members and a low probability of interactions with outsiders (shown in Equation 5.1). I.e., the interaction graphs of the

groups are cliques. However, in the pedestrian case, not every group member needs to be connected with every other members [3] due to the transitivity property of group relationships. On all the spring simulation data sets, the group interaction probabilities ($\geq 95.3\%$) are much higher than the non-group interaction probabilities ($\leq 13.9\%$), which satisfies the assumption that the movements of agents are influenced by their group members with high probabilities and by outsiders with low probabilities. However, with the increase of the number of particles and the value of the non-group probabilities, the influences' indication of the group relationships becomes weaker, which is discussed in Section 5.5.4. The reason is that the true negative rates (tn) become smaller with the increase of the number of particles and non-group interaction probabilities (shown in Table 5.12 and Figure 7.5). On the pedestrian data sets, the true negative rates (tp) of NRI and WavenetNRI with unsupervised training are low (shown in Table 5.13 and Figure 7.6), which suggests that the a large number of non-group relationships cannot be retrieved. The outsiders could influence the movements of group members in the pedestrian case. The influences from the outsiders can have different patterns with the influences from the group members; e.g., a pedestrian usually follows their group members and avoids collisions with the outsiders. The following and avoidance of collision are different patterns of influences on movements.

5.6.2 Problems with pairwise classification-based methods

The baseline Yamaguchi et al, NRI and our method WavenetNRI are pairwise classification-based. According to Table 5.8, the performances of these pairwise classification-based methods are unstable compared with the clustering-based baselines GD-GAN and Solera et al. I.e., the pairwise classification-based methods generally have high standard deviations of recall and precision based on group mitre (Δ_{GW}), which can limit the application of these methods in practice. According to Finley et al [7], the pairwise classification-based methods have the following problems:

- Pairwise classifiers cannot directly optimise the clustering objects. In our works, the goal of the pairwise classification-based methods such as WavenetNRI, NRI and Yamaguchi et al, optimise the edge accuracy defined by Equation 5.6, rather than our evaluation metris, i.e., recall and precision based on group mitre (Δ_{GW}). Compared with pairwise classification-based methods, the supervised clustering based baseline Solera et al [3] can directly optimise the group mitre Δ_{GW} with the BCFW algorithm [25].

- The imbalanced distribution of group/non-group labels can lead to underestimation of pairwise similarity. In our work, the imbalanced label distributions are shown in Figure 5.2. The non-group labels account for the majority, which could be the potential reason for the low true positive rates of Yamaguchi et al on all the spring simulation data sets and some

pedestrian data sets, i.e., *zara02* and *students03* (shown in Table 5.10 and Table 5.11). We applied the weighted cross-entropy loss function to address the imbalance problem in NRI and WavenetNRI. However, our work has not studied the effects of the weighted cross-entropy loss.

- Pairwise classifiers usually assume the pairs of agents are independent and cannot use the dependencies between the pairs. Although in our works, WavenetNRI and NRI can apply GNN to model the dependencies between different pairs by iterative node and edge updates and WavenetNRI can model the symmetric properties of the pairs, the transitive property of the group relationship cannot be learned effectively.

5.6.3 Limitations of our work

There are several limitations of our work:

- We only applied the Louvain community detection algorithm to transform the predicted pairwise interactions into groups. The effects of different community detection algorithms have not been studied.
- We have not explored the optimal hyperparameters in our work. Since our work is based on NRI, we use the hyperparameters suggested by Kipf et al (2018) [1], such as the hidden dimensions of the neural networks. However, in different cases, the optimal hyperparameters could be different. E.g., the number of examples of the pedestrian data sets is smaller than that of the spring simulation data sets, so the optimal hidden dimensions for the pedestrian data sets could be smaller than that for the spring simulation data sets.
- We have not studied the effect of the weighted cross-entropy loss function. I.e., whether the weighted cross-entropy loss function can help to solve the imbalanced group relationship problem.

Chapter 6

Conclusion and future work

In this work, we explored the application of graph neural networks (GNN) for group detection. We extended the work Neural Relational Inference (NRI) [1] for group detection. We tried to improve the performances for group detection by applying symmetric edge features with symmetric edge updating processes and replacing the 1D convolution layer with a gated dilated residual causal convolution block, as proposed by Wavenet [2]. We tested the effects of our changes to the original NRI on the performance of group detection in the ablation study. We found that on the spring simulation data sets, the gated dilated residual causal convolution block can slightly improve the performance of NRI. At the same time, the symmetric edge features with symmetric edge updating processes negatively affect the performance of NRI. On the pedestrian data sets, the symmetric edge features with symmetric edge updating processes can improve the performance of NRI, while the gated dilated residual causal convolution block has no significant effect on NRI. We compared our method WavenetNRI with other baselines on the six spring simulation data sets and five pedestrian data sets. NRI and WavenetNRI with supervised training outperform all other baselines on the spring simulation data sets. On the pedestrian data sets, although our method WavenetNRI cannot compete against the strong clustering-based baselines GD-GAN [8] and Solera et al [3], with supervised training, our method can outperform the pairwise classification-based method Yamaguchi et al [4] and the original NRI. We validated the assumption of the unsupervised training of NRI and WavenetNRI. The assumption is that the movements of the agents are to be influenced by their group members with high probabilities and by outsiders with low probabilities; i.e., the influences on movements can indicate group relationships. Although on the spring simulation data sets, the predefined interaction pattern satisfies this assumption, with the increase of the number of agents and the non-group interaction probabilities, the influences' indication of group membership becomes weaker. The assumption is invalid for the pedestrian data sets.

In future work, we will tackle the limitations of our current work. We will study the effects of different community detection algorithms. We will explore

optimising the hyperparameters of our model. We will study the effects of the weighted cross-entropy functions. It is worth studying how to extend our current pairwise classification-based method to supervised clustering; the corresponding optimisation algorithms need to be developed.

Bibliography

[1] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pages 2688–2697. PMLR, 2018.

[2] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[3] Francesco Solera, Simone Calderara, and Rita Cucchiara. Socially constrained structural learning for groups detection in crowd. *IEEE transactions on pattern analysis and machine intelligence*, 38(5):995–1008, 2015.

[4] Kota Yamaguchi, Alexander C Berg, Luis E Ortiz, and Tamara L Berg. Who are you with and where are you going? In *CVPR 2011*, pages 1345–1352. IEEE, 2011.

[5] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[6] Sydney Thompson, Abhijit Gupta, Anjali W Gupta, Austin Chen, and Marynel Vázquez. Conversational group detection with graph neural networks. In *Proceedings of the 2021 International Conference on Multimodal Interaction*, pages 248–252, 2021.

[7] Thomas Finley and Thorsten Joachims. Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, pages 217–224, 2005.

[8] Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. Gd-gan: Generative adversarial networks for trajectory prediction and group detection in crowds. In *Asian conference on computer vision*, pages 314–330. Springer, 2018.

[9] Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, page 104, 2004.

[10] Yingli Zhao, Zhengxi Hu, Lei Zhou, Meng Liu, and Jingtai Liu. Data-driven online group detection based on structured prediction. In *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*, pages 1220–1225. IEEE, 2020.

[11] Hao Chen, Seung Hyun Cha, and Tae Wan Kim. A framework for group activity detection and recognition using smartphone sensors and beacons. *Building and Environment*, 158:205–216, 2019.

[12] Sumit Kumar, Yiming Gu, Jerrick Hoang, Galen Clark Haynes, and Micol Marchetti-Bowick. Interaction-based trajectory prediction over a hybrid traffic graph. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5530–5535, 2021.

[13] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[14] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[15] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6272–6281, 2019.

[16] Abduallah Mohamed, Kun Qian, Mohamed Elhoseiny, and Christian Claudel. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14424–14432, 2020.

[17] Chengxin Wang, Shaofeng Cai, and Gary Tan. Graphtcn: Spatio-temporal interaction modeling for human trajectory prediction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3450–3459, 2021.

[18] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[21] Donsuk Lee, Yiming Gu, Jerrick Hoang, and Micol Marchetti-Bowick. Joint interaction and trajectory prediction for autonomous driving using graph neural networks. *arXiv preprint arXiv:1912.07882*, 2019.

[22] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library, 2007.

[23] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th international conference on computer vision*, pages 261–268. IEEE, 2009.

[24] Meinard Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[25] Simon Lacoste-Julien, Martin Jaggi, Mark Schmidt, and Patrick Pletscher. Block-coordinate frank-wolfe optimization for structural svms. In *International Conference on Machine Learning*, pages 53–61. PMLR, 2013.

[26] Marc Vilain, John D Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*, 1995.

[27] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Chapter 7

Appendix

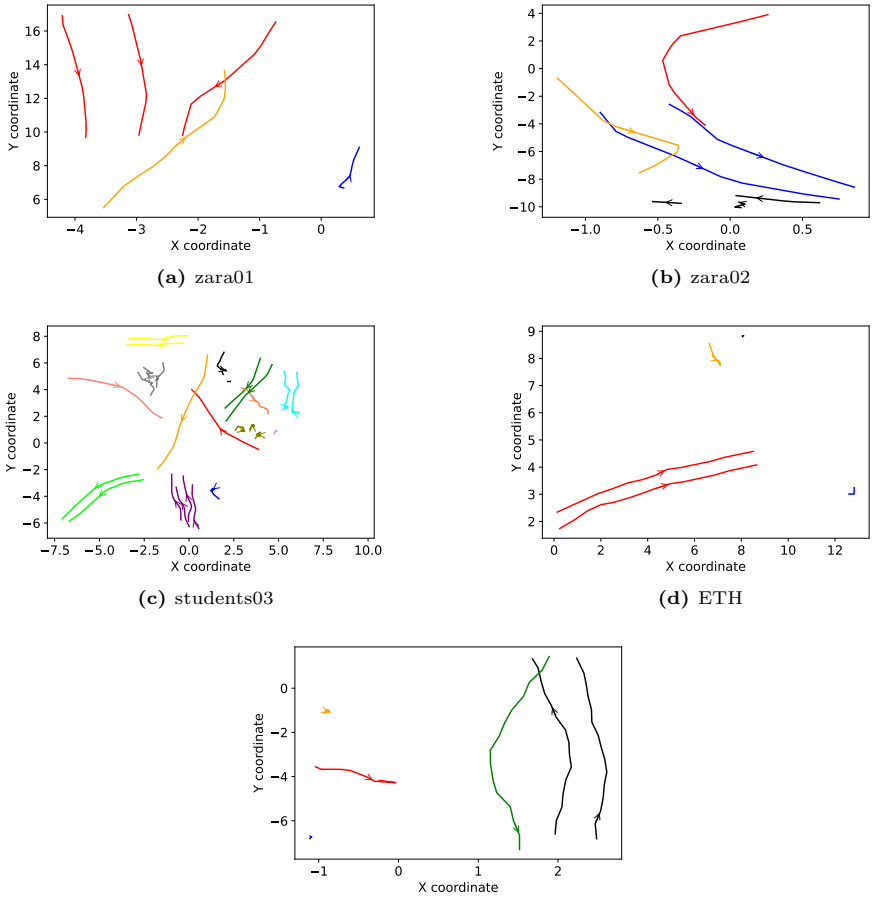
The figures and algorithms taking too much space are included in this appendix.

7.1 Figures

This appendix is used to provide access to the figures of trajectory visualisation, the confusion matrices of the pairwise classification-based methods and the plots of the mean squared error (MSE) and edge accuracy of the unsupervised training processes of NRI [1] and WavenetNRI.

7.1.1 Visualisation of Trajectories

The visualisation of the trajectories of the agents of the pedestrian data sets and that of the spring simulation data sets is shown in Figure 7.1 and Figure 7.2, respectively.



(a) zara01

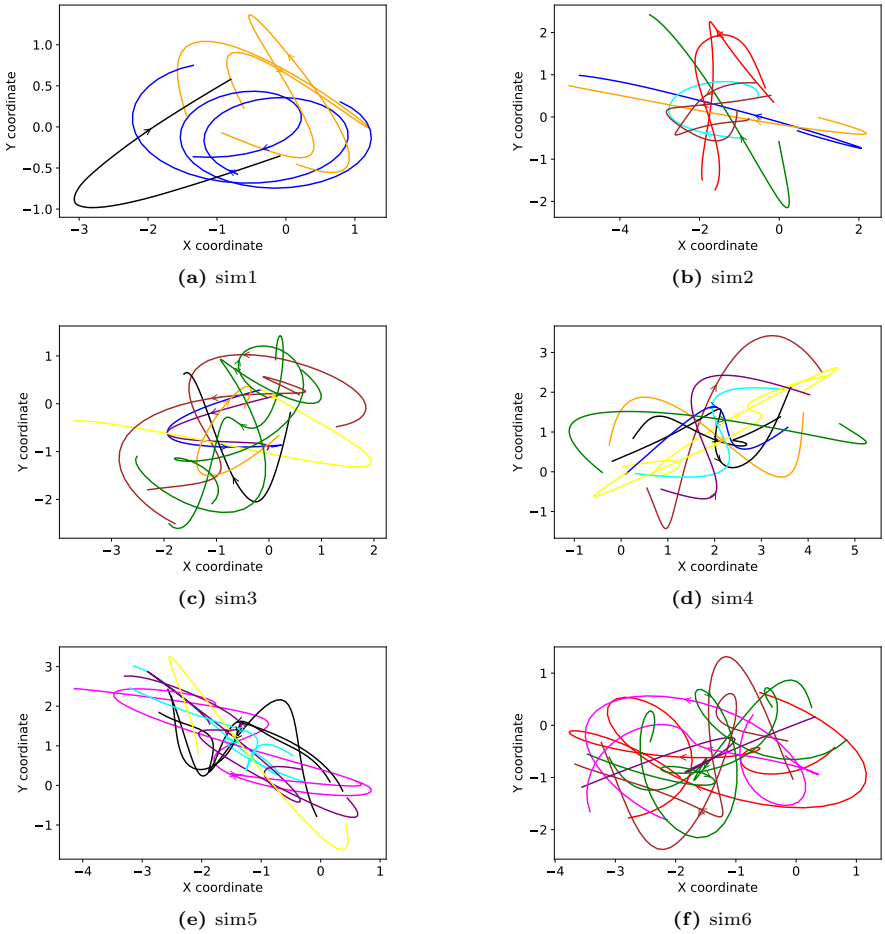
(b) zara02

(c) students03

(d) ETH

(e) Hotel

Figure 7.1: Visualisation of trajectories of pedestrian data sets. The colours denote the groups and the pedestrians or particles in the same group are labeled by the same colour; the arrows denote the directions of the trajectories.



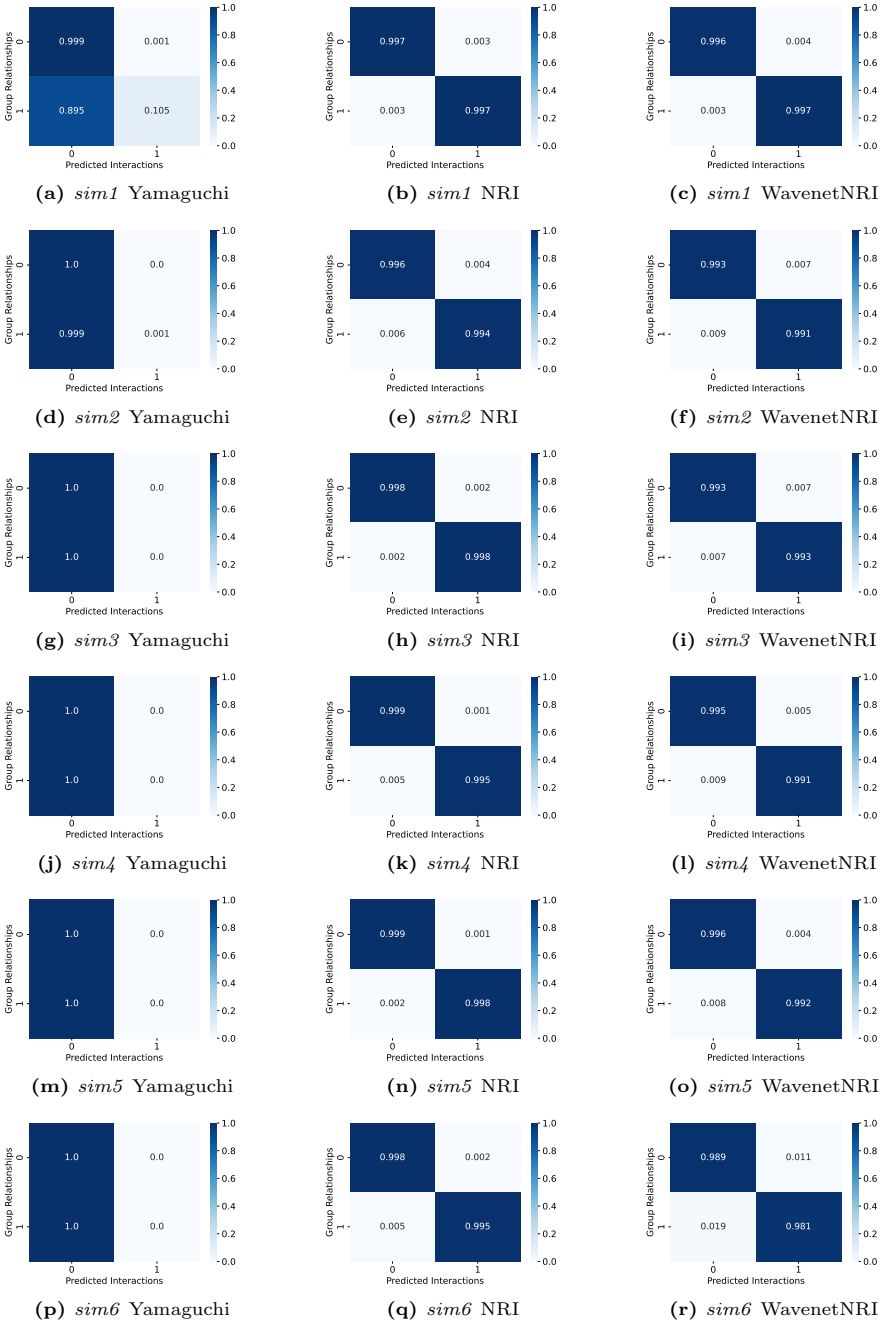
(a) sim1 **(b)** sim2

(c) sim3 **(d)** sim4

(e) sim5 **(f)** sim6

Figure 7.2: Visualisation of trajectories of spring simulation data sets. The colours denote the groups and the pedestrians or particles in the same group are labeled by the same colour; the arrows denote the directions of the trajectories.

7.1.2 Confusion Matrices of WavenetNRI and NRI



(a) *sim1* Yamaguchi **(b)** *sim1* NRI **(c)** *sim1* WavenetNRI

(d) *sim2* Yamaguchi **(e)** *sim2* NRI **(f)** *sim2* WavenetNRI

(g) *sim3* Yamaguchi **(h)** *sim3* NRI **(i)** *sim3* WavenetNRI

(j) *sim4* Yamaguchi **(k)** *sim4* NRI **(l)** *sim4* WavenetNRI

(m) *sim5* Yamaguchi **(n)** *sim5* NRI **(o)** *sim5* WavenetNRI

(p) *sim6* Yamaguchi **(q)** *sim6* NRI **(r)** *sim6* WavenetNRI

Figure 7.3: Confusion matrices of Yamaguchi, NRI and WavenetNRI on the spring simulation data sets.

(a) *zara01* Yamaguchi **(b)** *zara01* NRI **(c)** *zara01* WavenetNRI

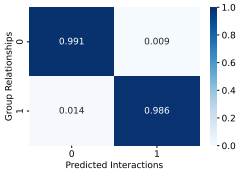
(d) *zara02* Yamaguchi **(e)** *zara02* NRI **(f)** *zara02* WavenetNRI

(g) *students03* Yamaguchi **(h)** *students03* NRI **(i)** *students03* WavenetNRI

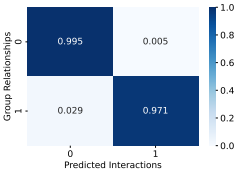
(j) *ETH* Yamaguchi **(k)** *ETH* NRI **(l)** *ETH* WavenetNRI

(m) *Hotel* Yamaguchi **(n)** *Hotel* NRI **(o)** *Hotel* WavenetNRI

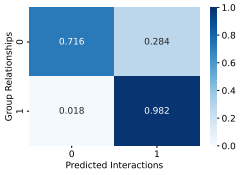
Figure 7.4: Confusion matrices of Yamaguchi, NRI and WavenetNRI on the pedestrian data sets.



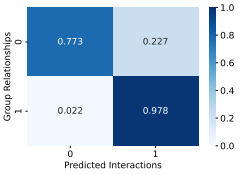
(a) *sim1* NRI(un)



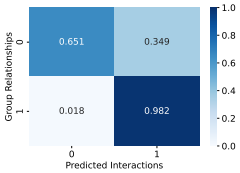
(b) *sim1* WavenetNRI(un)



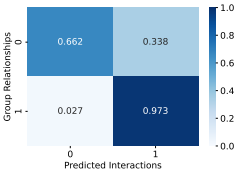
(c) *sim2* NRI(un)



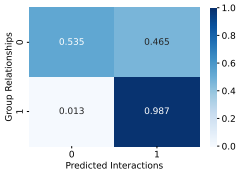
(d) *sim2* WavenetNRI(un)



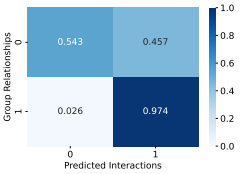
(e) *sim3* NRI(un)



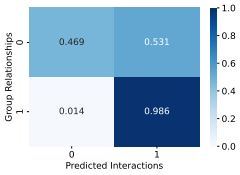
(f) *sim3* WavenetNRI(un)



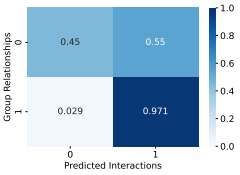
(g) *sim4* NRI(un)



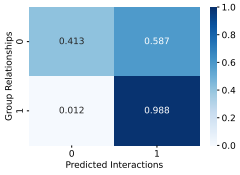
(h) *sim4* WavenetNRI(un)



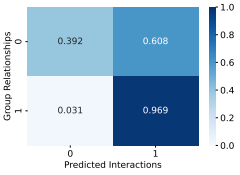
(i) *sim5* NRI(un)



(j) *sim5* WavenetNRI(un)

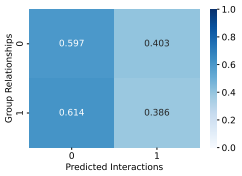


(k) *sim6* NRI(un)

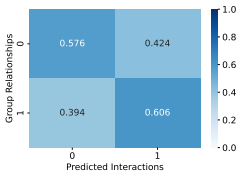


(l) *sim6* WavenetNRI(un)

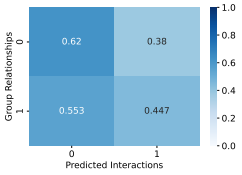
Figure 7.5: Confusion matrices of NRI and WavenetNRI with unsupervised training on the spring simulation data sets.



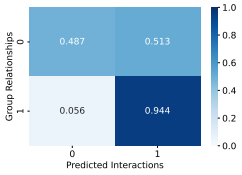
(a) *zara01* NRI(un)



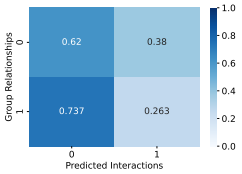
(b) *zara01* WavenetNRI(un)



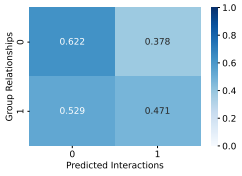
(c) *zara02* NRI(un)



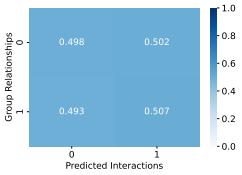
(d) *zara02* WavenetNRI(un)



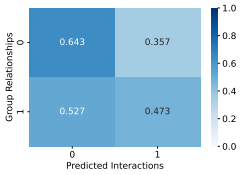
(e) *students03* NRI(un)



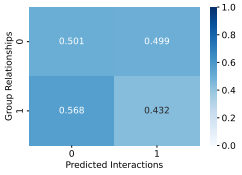
(f) *students03* Wavenet-NRI(un)



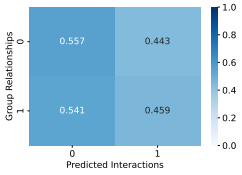
(g) *ETH* NRI(un)



(h) *ETH* WavenetNRI(un)



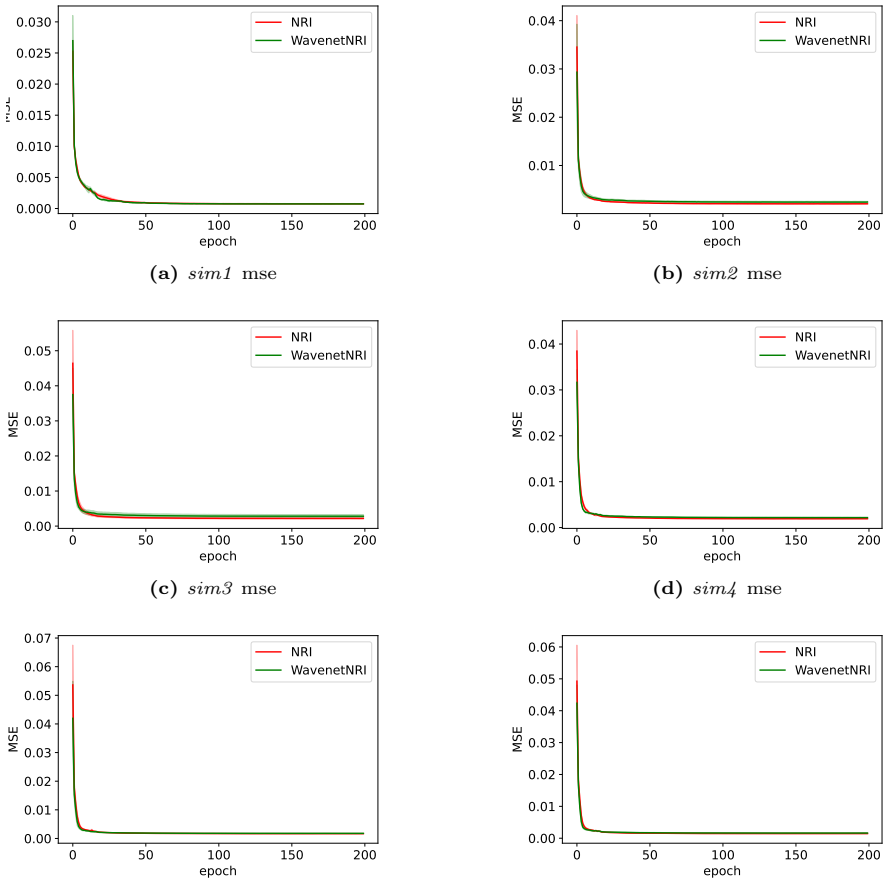
(i) *Hotel* NRI(un)



(j) *Hotel* WavenetNRI(un)

Figure 7.6: Confusion matrices of NRI and WavenetNRI with unsupervised training on the pedestrian data sets.

7.1.3 MSE and edge accuracy of NRI and WavenetNRI with unsupervised training



(a) *sim1* mse

(b) *sim2* mse

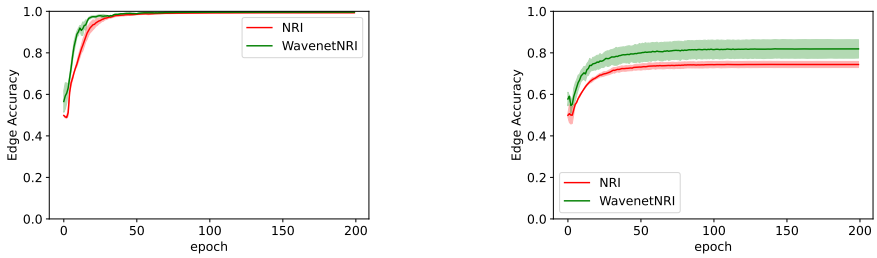
(c) *sim3* mse

(d) *sim4* mse

(e) *sim5* mse

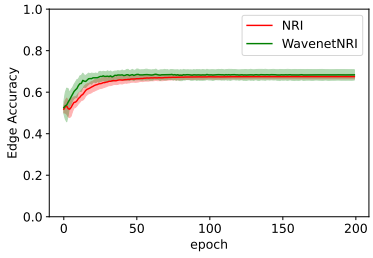
(f) *sim6* mse

Figure 7.7: MSE of spring simulation data sets with unsupervised training. Figure 7.7a to 7.7f show the training MSE of *sim1* to *sim6*. The green and red lines denote the average measurements of WavenetNRI and NRI, respectively. The corresponding shadow denotes the standard deviation.

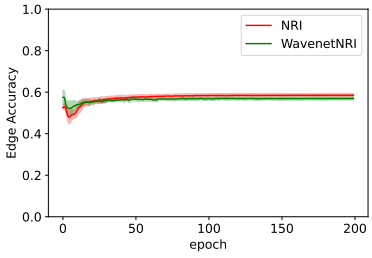


(a) *sim1* acc

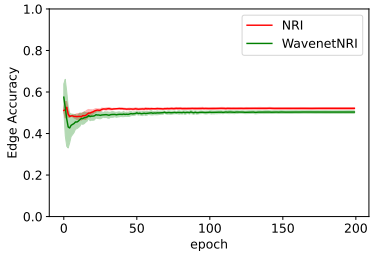
(b) *sim2* acc



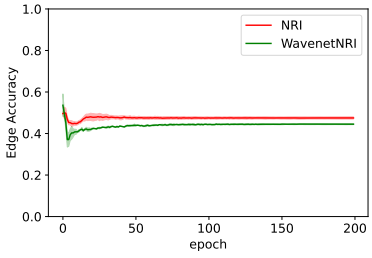
(c) *sim3* acc



(d) *sim4* acc

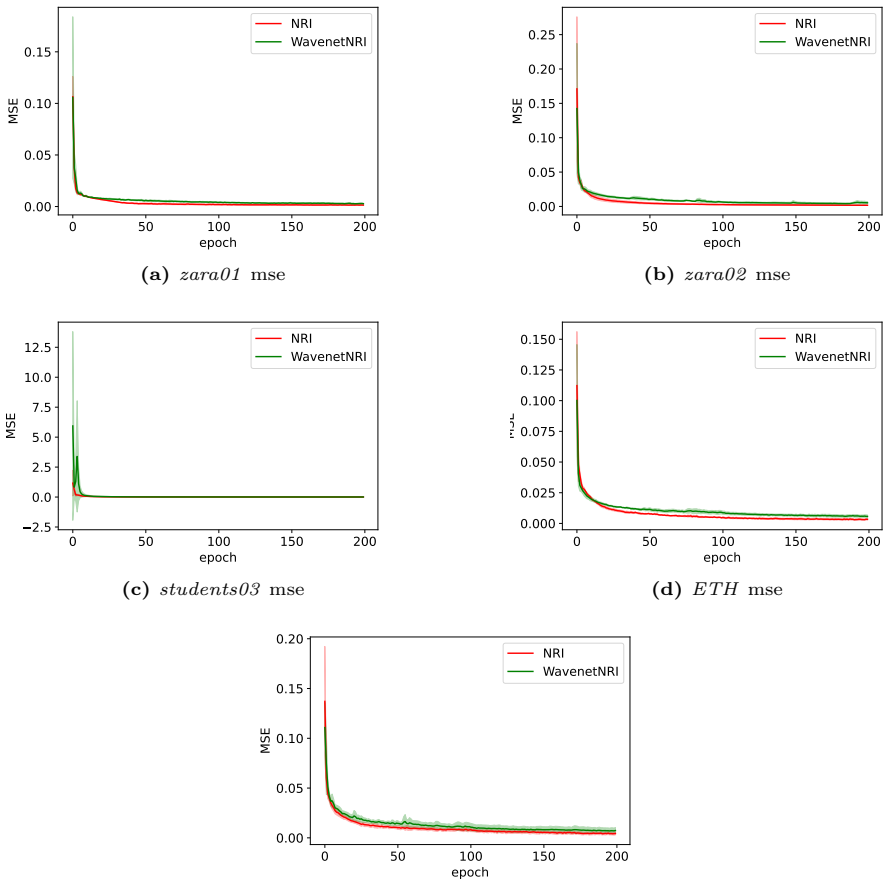


(e) *sim5* acc



(f) *sim6* acc

Figure 7.8: Edge accuracy of NRI and WavenetNRI of spring simulation data sets with unsupervised training. Figure 7.8a to 7.8f show the corresponding training edge accuracy. The green and red lines denote the average measurements of WavenetNRI and NRI, respectively. The corresponding shadow denotes the standard deviation.



(a) *zara01* mse

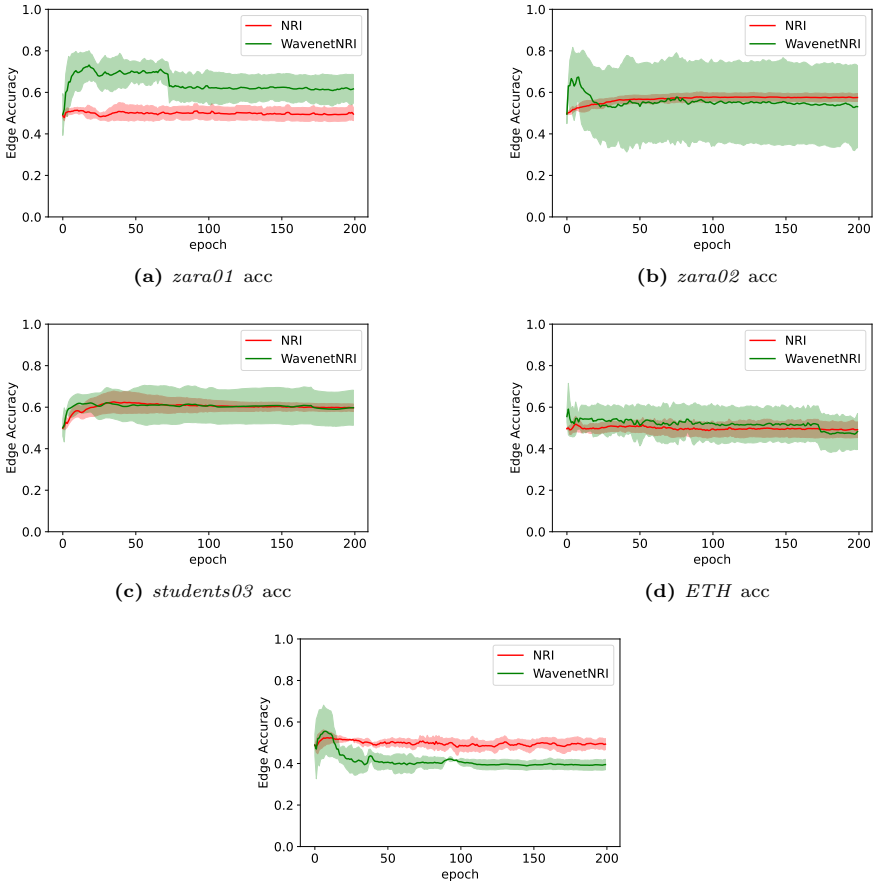
(b) *zara02* mse

(c) *students03* mse

(d) *ETH* mse

(e) *Hotel* mse

Figure 7.9: MSE of NRI and WavenetNRI on pedestrian data sets with unsupervised training. Figure 7.9a to 7.9e show the training MSE of *zara01* to *Hotel*. The green and red lines denote the average measurements of WavenetNRI and NRI, respectively. The corresponding shadow denotes the standard deviation.



(a) *zara01* acc

(b) *zara02* acc

(c) *students03* acc

(d) *ETH* acc

(e) *Hotel* acc

Figure 7.10: Edge accuracy of NRI and WavenetNRI on pedestrian data sets with unsupervised training. Figure 7.10a to 7.10e show the corresponding training edge accuracy. The green and red lines denote the average measurements of WavenetNRI and NRI, respectively. The corresponding shadow denotes the standard deviation.

7.2 Algorithms

The algorithm to compute recall of Group Mitre is listed in Algorithm 1. To compute precision of Group mitre, we can simply exchange the predicted group partition $\hat{C}$ and the label group partition C.

Algorithm 1: Computation of Recall of Group Mitre $\Delta_{GW}(C, \hat{C})$

Input: True group partition C and predicted group partition $\hat{C}$, where $C = \{c_i\}$ and $\hat{C} = \{\hat{c}_j\}$; c_i and $\hat{c}_j$ are sets denoting a single group of true and predicted partitions respectively. The elements in c_i and $\hat{c}_i$ are the agents, denoted by α; the corresponding fake agents are denoted by α'.

Result: Recall of Group Mitre Δ_{GM}

```
forall c_i ∈ C do
    if |c_i| = 1 then
        forall α ∈ c_i do
            generate α'
            c_i := c_i ∪ {α'}
        end
    else
        forall α ∈ c_i do
            generate α'
            C := C ∪ {α'}
        end
    end
end
forall ĉ_j ∈ Ĉ do
    if |ĉ_j| = 1 then
        forall αinĉ_j do
            generate α'
            ĉ_j := ĉ_j ∪ {α'}
        end
    else
        forall α ∈ ĉ_j do
            generate α'
            Ĉ := Ĉ ∪ {α'}
        end
    end
end
missing_links := 0
correct_links := 0
forall c_i ∈ C do
    num_partitions := 0
    partitions := {}
    size := |c_i|
    forall α ∈ c_i do
        forall ĉ_j ∈ Ĉ do
            if α ∈ ĉ_j then
                if partitions = ĉ_j then
                    continue
                else
                    partitions := ĉ_j
                    num_partitions := num_partitions + 1
                end
            end
        end
    end
    correct_links := correct_links + size − 1
    missing_links := missing_links + num_partitions − 1
end
Recall := (correct_links − missing_links) / correct_links
return Recall
```