Social-SSL: Self-Supervised Cross-Sequence Representation Learning Based on Transformers for Multi-Agent Trajectory Prediction

Li-Wu Tsao[1], Yan-Kai Wang[1], Hao-Siang Lin[1], Hong-Han Shuai[1], Lai-Kuan Wong[2], and Wen-Huang Cheng[1]

[1] National Yang Ming Chiao Tung University, Taiwan,
{lwtsao.eed09g,ykwang.ee09g,mrfish.eic08g,hhshuai,whcheng}@nctu.edu.tw
[2] Multimedia University, Malaysia, lkwong@mmu.edu.my

Abstract. Earlier trajectory prediction approaches focus on ways of capturing sequential structures among pedestrians by using recurrent networks, which is known to have some limitations in capturing long sequence structures. To address this limitation, some recent works proposed Transformer-based architectures, which are built with attention mechanisms. However, these Transformer-based networks are trained end-to-end without capitalizing on the value of pre-training. In this work, we propose Social-SSL that captures cross-sequence trajectory structures via self-supervised pre-training, which plays a crucial role in improving both data efficiency and generalizability of Transformer networks for trajectory prediction. Specifically, Social-SSL models the interaction and motion patterns with three pretext tasks: interaction type prediction, closeness prediction, and masked cross-sequence to sequence pre-training. Comprehensive experiments show that Social-SSL outperforms the state-of-the-art methods by at least 12% and 20% on ETH/UCY and SDD datasets in terms of Average Displacement Error and Final Displacement Error. [3]

Keywords: Trajectory Prediction, Self-Supervised Learning, Transformer, Representation Learning

1 Introduction

Human trajectory prediction plays a crucial role in path planning and collision avoidance for various applications, *e.g.*, autonomous driving and robot navigation [3,18,30,35,39]. The main challenges of trajectory prediction lie in the complexity of dynamic actions and multi-agent interactions. Early trajectory prediction works employed Kalman filter [11] and model-based methods [8,13,24,42] for learning dynamic actions of a single agent; Social Force models [8,15] learn to formulate human-like behaviors for goal navigation as classic interaction modeling. However, these hand-crafted approaches have difficulties dealing with complex multi-agent interactions in real-world applications.

[3] code available at https://github.com/Sigta678/Social-SSL

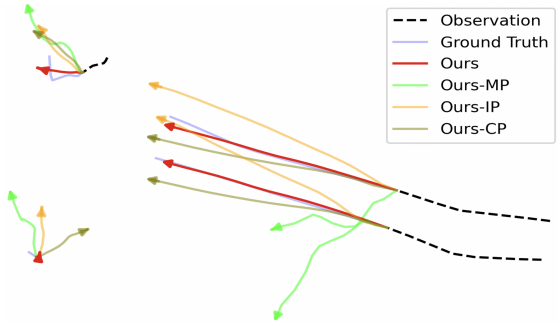


Fig. 1. An illustrative example on ZARA1 dataset. We design three pretexts, including IP (Interaction pretext), CP (Closeness pretext), and MP (Motion pretext). The figure demonstrates how three pretexts benefit the trajectory prediction. Ours-MP does not know how to correctly utilize the social representation, since there is not enough understanding of motion. Ours-IP tends to predict all trajectories pointing towards the upper left corner, since no interaction pretext guides the understanding of social behavior. Ours-CP established an inappropriate relationship between the far agent in the bottom left and the group of agents in the middle, making their predicted trajectories closer because no closeness pretext indicates the influence of mutual distance on social cues. Ours shows the effectiveness of three pretexts to maintain the agent's motion, capture the social pattern, and ignore the irrelevant social agents.

Recent trajectory prediction approaches adopt data-driven techniques based on Recurrent Neural Networks (RNN) [21], with most methods utilizing Long-Short Term Memory (LSTM) [9] as their base model. Various mechanisms such as social pooling [1], conditional variational autoencoder [35], attention [2,33,34], visual semantics [19], and graph neural networks [10,14,17,28,35] are proposed to model the social interactions between the agents so that the generated trajectory represents reasonable routes. While RNN can effectively model the trajectory of an agent as a temporal sequence to capture the dynamic actions of the agent, it is known to be inefficient in memorizing information from long sequences or a large amount of spatial cues. Social-STGCNN [25] is one of the works that moves away from using RNN to represent the spatio-temporal relationship fully as a Graph Convolutional Network (GCN) [12]. Moreover, with the great success of Transformers [41], Transformer TF [6] and STAR [43] adopted Transformers for single-agent and multi-agent trajectory predictions. A new branch of study includes the joint modeling by considering both spatial and temporal sequence in one model. The idea is proposed by AgentFormer [44], which utilizes an extremely long sequence as a complicated cross-sequence to combine all the agent sequences into one via Transformers.

However, these previous works did not capitalize on the benefit of the pre-training strategy, which is a key mechanism that contributes to the success of Transformers in Natural Language Processing. Motivated by the idea of cross-sequence that can model both spatial and temporal information, we propose Social-SSL, which is a self-supervised cross-sequence representation learning

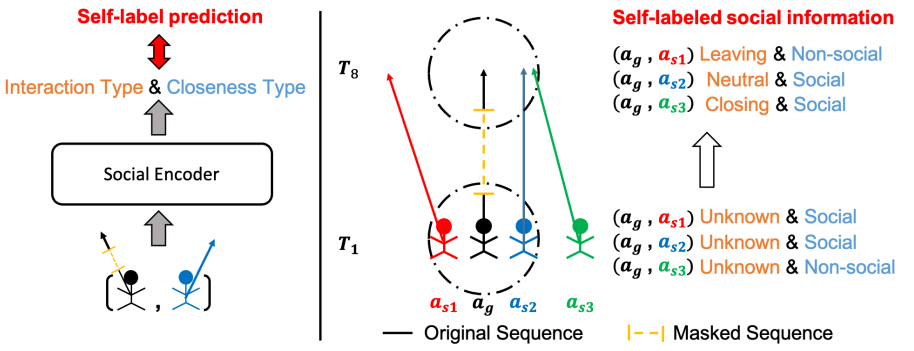


Fig. 2. Illustration of self-supervised cross-sequence representation learning. We propose to learn the interaction and motion patterns based on Transformer, using self-labeled social information and the masked section of a cross-sequence to extract representation without the complete sequence.

framework based on Transformers. As shown in Figure 2, two social-related pretext tasks and a motion-related pretext task are proposed. Specifically, interaction type prediction and closeness prediction are designed to capture the inter-relation between the target agent and each of the social agents. Meanwhile, masked cross-sequence to sequence pre-training provides the understanding of intra-relation among the remaining sequence of the target agent. In our study, combining both the inter- and intra-relation into a cross-sequence representation is effective for crowded agent scenarios and can reduce the amount of data needed for fine-tuning. To the best of our knowledge, this is the first work that proposes the pre-training strategy on a social-related trajectory model.

The core contributions of Social-SSL are highlighted as follow:

- Two social-related pretext tasks for cross-sequence inter-relation learning; interaction type prediction and closeness prediction that learns the social interaction patterns between every pair of agents in the scene.
- A motion-related pretext task for cross-sequence intra-relation learning; the masked cross-sequence to sequence pre-training that learns the motion of a target agent from the non-masked section of its own sequence, and discover the inter-relation with its surrounding social agents via the cross-sequence structure.
- By capitalizing the advantage of self-supervised representation learning, Social-SSL achieves state-of-the-art results on trajectory prediction task, even when trained on a small amount of trajectory data. In addition, it can improve the generalizability of the trajectory prediction task.

2 Related Work

Human trajectories are heavily influenced by the social interaction among agents. With the rise of deep learning, the quality and the amount of data becomes more

critical. Based on the importance of data, Social-LSTM [1] is a pioneering work that models social interaction for trajectory prediction via the concept of social pooling, which aggregates the hidden states among the nearby agents. Social-GAN [7] revises the pooling module to capture global information from all agents in the scene. SoPhie [33] proposes the idea of social attention that introduces an importance weight for each pair of agents. Different from previous works that utilize LSTM models, a recent line of studies introduced new ideas, such as graph attention networks [10, 14], for modelling the social interactions between pedestrians.

Since Transformer has its ability to handle the long-sequence problem well, using Transformers to solve the problem on trajectory prediction becomes more popular. For instance, Transformer TF [6] first introduced the use of Transformer for single-agent trajectory prediction. Capitalizing on the self-attention advantage of Transformer, STAR [43] introduced a Graph Transformer Network that models the spatial interaction on a Transformer-based graph convolution mechanism and memorizes the embeddings of each pedestrian via a temporal Transformer. AgentFormer [44] designed the spatio-temporal Transformer along with better multi-modal properties, and used all the observable agent sequence in the scene to conduct a complicated cross-sequence input with end-to-end training. Unlike previous works, we introduce a novel approach on the spatio-temporal pre-training strategy under a deterministic setting. This makes better use of the sequence data by predicting the self-labeled social types and recovering the masked cross-sequence as the supervision for pre-training. Our studies show that Social-SSL can reduce the need for a large amount of training data and improve the generalizability of trajectory prediction task. To the best of our knowledge, our work is the first to consider trajectory forecasting in the self-supervised manner, introducing brand new tasks to embed the social representation into a basic Transformer [41] model.

3 Social-SSL

We introduce Social-SSL, a self-supervised cross-sequence representation learning framework via Transformer, to better capture the interaction and motion embeddings from the proposed pretext tasks. Since Transformers are data-hungry, *i.e.*, require a large amount of training data, by creating the pretext tasks related to the downstream task whereby labels can be automatically generated, the data representation can be trained by using a huge amount of additional self-supervised label, rather than using only the downstream task label itself. In our study, these prior knowledge of social interaction and motion representations can be quickly transferred to the trajectory prediction task.

Figure 3 illustrates the architecture of our proposed Social-SSL. Given a paired sequence of the target agent and a social agent, which we called the cross-sequence in our work, the model is pre-trained using the cross-sequence with three pretext tasks to model the feature representations from the aspect of social interaction and agent's motion.

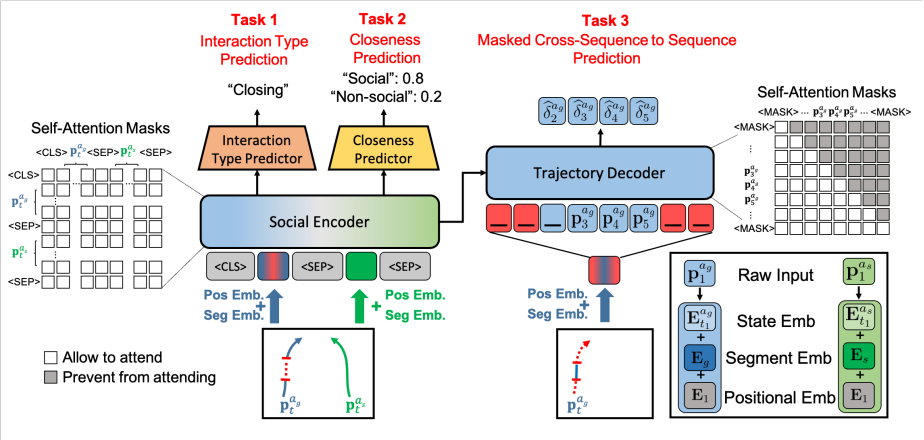


Fig. 3. The pre-training structure of Social-SSL. The input sequence for the encoder constitutes the masked sequence of the target agent a_g and the unmasked sequence of a social agent a_s, separated by <SEP>. <CLS> in front of the sequence aggregates the social representation from a_g and a_s. The red dotted line and the symbol "_" both represent <MASK> in the observation sequence of the target agent. The social-related pretext tasks, *i.e.*, interaction type recognition and closeness prediction, are designed to learn the social representation from the cross-sequence. The motion-related pretext task, *i.e.*, the masked cross-sequence to sequence pre-training, aims to learn to recover the target agent trajectory from its incomplete encoder sequence.

In Social-SSL, we set our Social Encoder and Trajectory Decoder as a simple Transformer Encoder and Decoder, which is simple yet effective for pre-training tasks. Since these three pretext tasks are trained simultaneously and share the same parameters of Social Encoder, the Trajectory Decoder side with the masked cross-sequence to sequence task, is enforced to consider both intra- and inter-relations between agents via the target agent's motion and the influential social information from the Social Encoder respectively. After training on these pretext tasks, Social-SSL can model all the observed cross-sequences as spatio-temporal embeddings and implicitly provide useful social information when predicting the unknown sequence of future timestamps.

3.1 Preliminary

Problem Definition. Given a set of N agents $\mathcal{A} = \{a_j | \ 1 \leq j \leq N, \ j \in \mathbb{N}\}$ over a past time period $(1, 2, \cdots, t, \cdots, t_c)$, where t_c is the current timestamp, the observed positions of agent a_j are re-scaled to $[0, 1]^2$ and can be represented as $(\mathbf{p}_1^{a_j}, \mathbf{p}_2^{a_j}, \cdots, \mathbf{p}_t^{a_j}, \cdots)$, where $\mathbf{p}_t^{a_j} = (x_t^{a_j}, y_t^{a_j})$ denotes the re-scaled position of agent a_j at time t. Since the model should focus on predicting the relative coordinates instead of absolute positions, we transform the absolute coordinates into the relative coordinates. Let $\boldsymbol{\delta}_t^{a_j} = (\delta_{x,t}^{a_j}, \delta_{y,t}^{a_j})$ denote the relative coordinates of agent a_j at time t, which can be calculated by $\boldsymbol{\delta}_t^{a_j} = \mathbf{p}_t^{a_j} - \mathbf{p}_{t-1}^{a_j}$.

In the pre-train phase, the Social Encoder is trained with the pretext tasks using only the observed trajectory (8 frames) of the training data to create the self-labeled social information. This setting ensures that our method can avoid memorizing the sequence information while fine-tuning on the trajectory prediction task. Meanwhile, both the interaction type predictor ($f_I(.)$) and closeness predictor ($f_C(.)$) adopt a 2-layer MLP, which plays the role of simple decoders.
Input Representation. The self-attention masks in Social Encoder allows the model to learn the social relationship from two observation sequences in a bidirectional spatio-temporal form. The hidden state corresponding to <CLS> is used as the aggregate representation of sequences a_g and a_s through self-attention for interaction type and closeness prediction tasks, denoted by $\mathbf{s}_f$. In other words, the social feature ($\mathbf{s}_f$) output by the Social Encoder at the <CLS> position, is used as the input by the interaction type predictor and closeness predictor, denoted as $f_I(\mathbf{s}_f)$ and $f_C(\mathbf{s}_f)$. The State Embeddings, Segment Embeddings, and Position Embeddings are combined as the encoder input with dimension d_{model} set to 256. The State Embeddings are represented by using a fully connected layer $\psi(.)$ that projects $(x_t^{a_j}, y_t^{a_j})$ into $\mathbb{R}^{256}$. The state definition for <CLS> is (1,0), <SEP> is (0,1), <MASK> is (0,0). This setting is based on the 2-dimension structure of the re-scaled position $(x_t^{a_j}, y_t^{a_j})$, which is between 0 and 1. Moreover, we use Segment Embeddings to differentiate the embeddings of the target agent (a_g) and its surrounding social agent (a_s) for the encoder input, where 0 and 1 represent the target and social agent respectively and are projected into $\mathbb{R}^{256}$. The Position Embeddings are the same as Transformer [41], which enforces the sequential structure in the self-attention operations.

3.2 Pretext Task

In this section, we present three pretext tasks for interaction and motion modeling: 1) interaction type prediction, 2) closeness prediction, and 3) masked cross-sequence to sequence prediction. For interaction type prediction and closeness prediction, Social-SSL learns to capture social information from the cross-sequence self-supervision with the Social Encoder. For masked cross-sequence to sequence prediction, the Trajectory Decoder of Social-SSL aims to predict the masked section of the target agent by utilizing the self-sequence and the important social relationships captured via Social Encoder. Also, by adding segment and positional embeddings, the sequence separation of two agents and spatio-temporal information are better captured from the pretext. In the following, we introduce the self-labeling mechanism to assign the label automatically for interaction type and closeness.

Task 1: Interaction Type Prediction. The idea of interaction type prediction comes from observing distance fluctuation between two agents, which can be an analogy to summarize the positive and negative sentiment from documents. For example, if a comment "good service" on a restaurant with this positive aspect term happens three times, and a comment "poor service" happens once, we might have an impression that this restaurant provides "good service". Based

on this analogy, we sum up the frequency of both agents getting closer and the frequency of them leaving away to create two classes of "closing" and "leaving". In addition, the class "neutral," is used to represent cases where two agents are not apparent whether they are closing or leaving.

Specifically, let $I_t^{(a_g,a_s)}$ denote the indicator of the sign obtained from the change in distance between the target and the social agent, a_g and a_s at time t:

$$I_t^{(a_g,a_s)} = \begin{cases} +1 & \text{if } d(\mathbf{p}_t^{a_g}, \mathbf{p}_t^{a_s}) - d(\mathbf{p}_{t-1}^{a_g}, \mathbf{p}_{t-1}^{a_s}) > 0, \\ -1 & \text{if } d(\mathbf{p}_t^{a_g}, \mathbf{p}_t^{a_s}) - d(\mathbf{p}_{t-1}^{a_g}, \mathbf{p}_{t-1}^{a_s}) < 0, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $d(.)$ is the Euclidean distance function. To track the trend of social relations, we summarize all the indicators within a period r[4] and determine the interaction type between two agents as follows.

$$y_{inter}^{(a_g,a_s)} = \begin{cases} 1\ (leaving) & \text{if } \sum_{t\in[t_c-r,t_c]} I_t > 0, \\ 2\ (closing) & \text{if } \sum_{t\in[t_c-r,t_c]} I_t < 0, \\ 3\ (neutral) & \text{otherwise.} \end{cases} \tag{2}$$

The output of interaction type prediction denoted as $\widehat{y}_{inter}^{(a_g,a_s)}$ is obtained by performing multi-class classification with cross-entropy as the loss function, L_I:

$$L_I = -\sum_{i=1}^{3} P(y_{inter}^{(a_g,a_s)} = i) \log P(\widehat{y}_{inter}^{(a_g,a_s)} = i). \tag{3}$$

Task 2: Closeness Prediction. Intuitively, an agent near the target agent has more influence on the trajectory than an agent who is far away. The closeness prediction is used to capture this social characteristic. Specifically, we separate closeness prediction into sparse prediction and dense prediction, which could better adapt to different scenarios.

Sparse prediction. At any time $t \in [1, t_c]$, if the distance between target agent a_g and social agent a_s is smaller than the distance threshold, d_{th}, we assign their closeness label as 1, otherwise 0:

$$y_{close}^{(a_g,a_s)} = \begin{cases} 1\ (nearby/social) & \text{if } \exists\ d(\mathbf{p}_t^{a_g}, \mathbf{p}_t^{a_s}) < d_{th}, \\ 0\ (faraway/non-social) & \text{otherwise.} \end{cases} \tag{4}$$

The sparse prediction as shown in Figure 4, denoted as $\widehat{y}_{cs}^{(a_g,a_s)}$, is performed using binary cross-entropy as the objective function, L_{CS}:

$$L_{CS} = -\sum_{i=0}^{1} P(y_{close}^{(a_g,a_s)} = i) \log P(\widehat{y}_{cs}^{(a_g,a_s)} = i). \tag{5}$$

[4] In the experiments, r is set to half of the input length empirically (In Appendix).

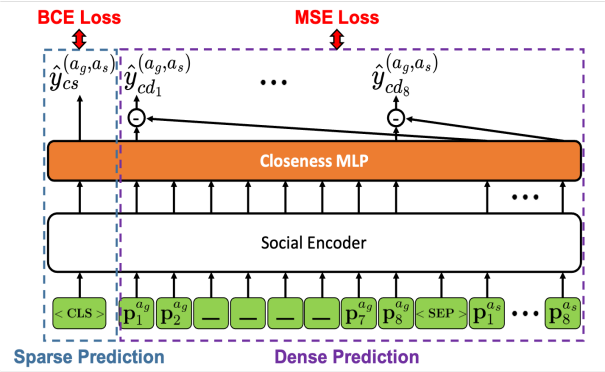


Fig. 4. Details for closeness pretext with sparse and dense prediction settings.

Dense prediction. For dense prediction, we consider the precise distance between the target agent a_g and a social agent a_s, represented by the distance difference of the corresponding timestamps, denoted as $\widehat{y}_{cd_t}^{(a_g,a_s)}$ in Figure 4. The objective function L_{CD} for guiding the dense prediction for closeness is formulated using mean square error as:

$$L_{CD} = \frac{1}{t_c}\sum_{t=1}^{t_c}(d(\mathbf{p}_t^{a_g}, \mathbf{p}_t^{a_s}) - \widehat{y}_{cd_t}^{(a_g,a_s)})^2. \tag{6}$$

Finally, L_{CS} and L_{CD} are added together to form the total loss, L_C for the closeness prediction task.

Task 3: Masked Cross-Sequence to Sequence Prediction. The third pretext task is created by randomly masking a target agent's subsequence, and the model should learn to reconstruct the masked subsequence from the cross-sequence input. Since the ground truth of the masked subsequence is known before being masked, the model can be trained in a self-supervised learning manner. The advantages of this pretext task are three-fold. First, this mechanism enhances the representation of the target agent sequence. Simultaneously pre-training the Social Encoder and Trajectory Decoder enforces the model to capture the intra-relation by reconstructing the masked subsequence of the target agent on the decoder side. Second, the model learns a robust representation that can handle a sequence with missing parts, which may happen due to occlusion. Third, the model learns to consider not only the intra-relation of the target trajectory but also the social inter-relation among social agents when predicting the missing part of a sequence. If the sequence information from the target agent itself is not sufficient to reconstruct the masked subsequence, the social cues can provide additional assistance to constrain the reconstructed trajectory.

Specifically, t_{ms} and t_{me} are denoted as the start and the end of the randomly masked subsequence timestamps. We adopt the auto-regressive structure on the decoder to predict the trajectory during fine-tuning, where the self-attention

masks are arranged in a left-to-right form [5]. The output of the decoder is represented as $\widehat{\delta}_i^{a_j}$, which denotes the reconstruction of the target sequence on the masked timestamps. The mean square error is used as the objective function for the masked cross-sequence to sequence pre-training:

$$L_{MSE} = \frac{1}{N}\sum_{j=1}^{N}(\frac{1}{t_{me}-t_{ms}+1}\sum_{i=t_{ms}}^{t_{me}}(\delta_i^{a_j}-\widehat{\delta}_i^{a_j})^2). \tag{7}$$

The overall objective function in pre-training phase can then be represented as:

$$L_{pre} = L_{MSE} + \lambda_I L_I + \lambda_C L_C, \tag{8}$$

where λ_I and λ_C are respectively the hyperparameters controlling the importance of the losses for interaction type prediction and closeness prediction.

It is worth noting that the masked language model is widely-used in NLP, *e.g.*, BERT [4], MASS [40]. Different from the masked language model that only considers the context in the target sequence, the proposed Masked Cross-Sequence to Sequence Prediction also takes the sequence from other agents as the context. As such, the proposed pretext task helps the model learn the fine-grained social importance for predicting the masked subsequence.

4 Experiments

4.1 Experimental Setup

Datasets. We perform the comparisons on the benchmark datasets: ETH [31], UCY [16], and SDD [32]. These datasets are collected from the bird's eye view with many human interactions. Specifically, ETH is a small dataset which contains the ETH and HOTEL scenes with less than 4 and 8 pedestrians in each frame, respectively. UCY is a larger dataset which contains the ZARA1, ZARA2, and UNIV scenes, with each scene having more than 10 pedestrians. For the UNIV scene, most frames contains 20 to 40 pedestrians. On the other hand, SDD is a large-scale dataset containing pedestrians, bicyclists, and vehicles.
Settings. Follow the setting of prior works [7,25], we use leave-one-out training on ETH and UCY datasets. Different from ETH and UCY, SDD is commonly-used with the standard train-test split in prior works [29,33]. The experimental setting on the observable period is 8 frames and the prediction period is 12 frames. In the pre-training phase, we use only the observable period of the training set to avoid data leakage. When fine-tuning on the trajectory prediction task, the observable period of the training set remains as the input information, whereas the future period is used as the groundtruth in computing the MSE loss for guiding the trajectory prediction.
Metrics. Similar to prior works [2,7], we use Average Displacement Error (ADE) and Final Displacement Error (FDE) to evaluate the results. ADE reflects the reaction to abrupt change, while FDE reflects more on the long-term goal.

4.2 Implementation details

Social-SSL uses 4 layers on both Transformer Encoder and Transformer Decoder, $d_{model} = 256$, and 16 attention heads. AdamW [22] optimizer is adopted for two phases; pretext task training and trajectory prediction fine-tuning.

Pretext task training: For each agent in a scene, we generate $N - 1$ agent pairs for cross-sequence learning, with the purpose of enlarging the amount of self-supervision data. We set λ_I and λ_C to 1, with the learning rate of 3e-6, for the first 700 epochs. For the remaining 300 epochs, λ_I and λ_C are reduced to 0.01, with the learning rate set to 3e-7.

Fine-tune on trajectory prediction: We fine-tune the weights of the Trajectory Decoder by freezing the Social Encoder, so that the embedded inter- and intra-relation can be preserved. Meanwhile, we add a Gumbel-Softmax to select the influential social agent for each target agent. The setting of learning rate is further discussed in our experimental study.

Inference on trajectory prediction: We perform auto-regressive decoding on the Trajectory Decoder by choosing the pair of cross-sequence from the result of Gumbel-Softmax. This produces our deterministic result for each target agent in the multi-agent scenario. The experiments are implemented using PyTorch on an RTX 2080 Ti GPU. The average results from 10 runs are reported.

Model	Venue	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Linear		1.33/2.94	0.39/0.72	0.82/1.59	0.62/1.21	0.77/1.48	0.79/1.59
Social-GAN [7]	CVPR'18	1.13/2.21	1.01/2.18	0.60/1.28	0.42/0.91	0.52/1.11	0.74/1.54
STGAT [10]	ICCV'19	0.88/1.66	0.56/1.15	0.52/1.13	0.41/0.91	**0.31**/0.68	0.54/1.11
Transformer TF [6]	ICPR'20	1.03/2.10	0.36/0.71	0.53/1.32	0.44/1.00	0.34/0.76	0.54/1.17
STAR [43]	ECCV'20	0.76/1.67	0.41/0.95	0.61/1.32	0.48/1.06	0.39/0.85	0.53/1.17
AgentFormer [44]	ICCV'21	0.99/1.96	0.37/0.76	0.64/1.34	0.46/1.00	0.38/0.84	0.57/1.18
Social-DPF [38]	AAAI'21	0.69/1.35	0.39/0.84	0.61/1.00	**0.40**/0.89	0.39/0.84	0.50/0.98
Social-SSL-S		**0.68/1.27**	0.26/0.47	0.55/1.02	0.43/0.85	0.34/0.67	0.45/0.86
Social-SSL-D		0.69/1.34	0.27/0.48	0.53/0.95	0.43/**0.84**	0.34/**0.65**	0.45/**0.85**
Social-SSL		0.69/1.37	**0.24/0.44**	**0.51/0.93**	0.42/**0.84**	0.34/0.67	**0.44/0.85**

Table 1. Comparison with state-of-the-art methods on ADE/FDE metrics. All the baselines are evaluated using only 1 sample. A lower ADE/FDE value indicates better performance. Social-SSL-S and Social-SSL-D indicate the closeness pretext using sparse and dense settings individually, and the combination of sparse and dense setting is Social-SSL.

4.3 Quantitative Results

Our proposed Social-SSL captures the deterministic social embeddings based on the self-supervision of interaction labels. To better transfer the pre-trained embeddings to the fine-tuning task, the deterministic evaluation is a better way to measure its performance. Table 1 shows the deterministic results of different baselines on ETH and UCY datasets. The results demonstrate that Social-SSL outperforms state-of-the-art methods by at least 12% in terms of ADE and FDE.

Compared with AgentFormer that uses a more "complicated" cross-sequence input structure via Transformer, we demonstrate the advantages of using "simple" cross-sequence with our pre-train strategy. From Table 1, Social-SSL emphasizes the potential disadvantage of AgentFormer under low sampling, while demonstrating the effectiveness of our proposed pretext tasks under deterministic setting with suitable supervision of cross-sequence.

Interestingly, the combination of sparse and dense settings in closeness pretext demonstrates that the social information learned by Social-SSL could better solve complex social situations with large amount of pedestrians in the scene, which indicates the improvement on the UNIV dataset. It is also worth noting that Social-SSL shows significant improvements on the HOTEL dataset, *i.e.*, at least 33% and 38% reduction in terms of ADE and FDE. Since HOTEL is the scene with 90-degree rotation change in trajectory direction, which is not seen in the other datasets, the performance of Social-SSL in this scenario could be attributed to the adoption of the self-supervised learning approach, which is known for its capability in improving generalizability [26, 27].

Model	Venue	Sampling	ADE/FDE
SoPhie [33]	CVPR'19	20	16.27/29.38
SimAug [20]	ECCV'20	20	10.27/19.71
PECNet [23]	ECCV'20	20	8.92/15.63
LB-EBM [29]	CVPR'21	20	8.87/15.61
SIT [36]	AAAI'22	20	8.59/15.27
Social-SSL		1	**6.63/12.23**

Table 2. State-of-the-art performance comparison on SDD (Evaluated in pixels).

Table 2 shows that the proposed Social-SSL outperforms state-of-the-art methods by at least 23% and 20% in terms of ADE and FDE on SDD dataset. Please note that the results of the baselines here are obtained by sampling 20 times and selecting the best prediction since these models are multi-modal. Interestingly, although Social-SSL uses only 1 sample, it outperforms all the baselines. SDD is a more challenging large-scale dataset containing the complicated interactions among heterogeneous agents, *i.e.*, pedestrians, bicyclists, and vehicles. The baseline methods learn the interaction patterns from the supervision of trajectory prediction, which is an indirect way to model and capture social interaction among heterogeneous agents. In contrast, our pre-train strategy is able to capture not only the interaction among the pedestrians but also the social interaction across different types of agents from a large dataset, leading to better performance. This result exemplifies the advantage of pre-training.

Small amount data training. Table 3 compares the performance of different methods trained on 10% and 100% data of the five datasets. We chose Social-STGCNN and SGCN as baseline comparisons because these methods claimed data efficiency on fewer amounts of data. On 10% data setting, Social-SSL outperforms Social-STGCNN and SGCN by at least 51% and 32% in terms of ADE and FDE. Also, the performance of Social-SSL using only 10% data is comparable to SGCN, demonstrating the effectiveness of our pre-train strategy.

Method	Data amount	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Social-STGCNN [25]	10%	2.01/3.08	1.90/3.08	1.30/2.04	1.15/1.85	1.06/1.68	1.48/2.35
	100%	0.92/1.81	0.76/1.49	0.63/1.26	0.52/1.06	0.44/0.90	0.65/1.30
SGCN [37]	10%	1.33/2.63	0.90/1.69	0.81/1.53	0.69/1.27	0.66/1.23	0.88/1.67
	100%	0.86/1.76	0.57/1.12	0.61/1.23	0.49/1.01	0.36/0.75	0.58/1.17
Social-SSL	10%	**0.85/1.75**	**0.46/0.85**	**0.68/1.25**	**0.50/0.96**	**0.41/0.77**	**0.59/1.14**
	100%	**0.69/1.37**	**0.24/0.44**	**0.51/0.93**	**0.42/0.84**	**0.34/0.67**	**0.44/0.85**

Table 3. Comparison on a small amount of data training. We use the same amount of data for pre-training and fine-tuning with leave-one-out setting.

4.4 Qualitative Results

Amount of data for fine-tuning. Figure 5 demonstrates the advantage of our pre-training strategy. Using 100% of data for fine-tuning takes more time, and does not seem to perform better than using 1% of data after the learning rate decays. Interestingly, with only 1% data used for fine-tuning, the training can easily converge and reach good enough performance within a few epochs. This could be attributed to the effectiveness of the masked cross-sequence to sequence pre-training task. Since we pre-trained the Trajectory Decoder in a way that is closely related to the trajectory prediction task, the pre-train model itself can rapidly transfer the knowledge to downstream tasks with simple hints.

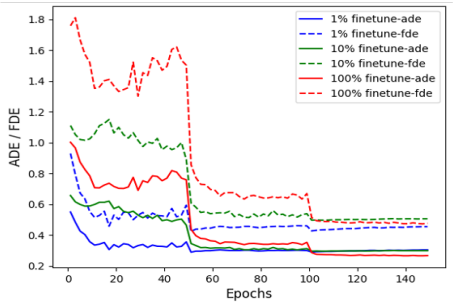
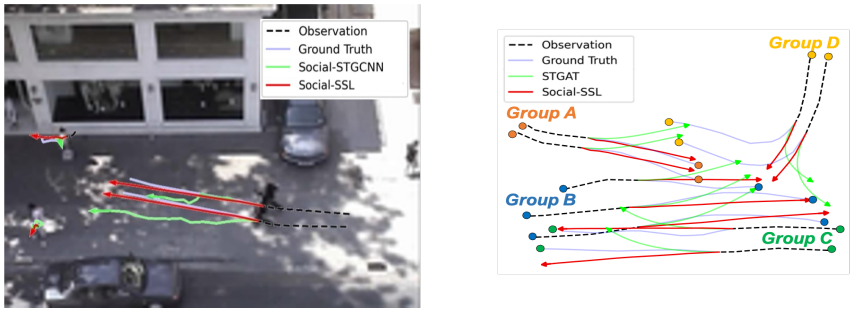


Fig. 5. Evaluation results using different amounts of data to fine-tune on HOTEL dataset. We set the initial learning rate as 3e-6 and the decay rate by 0.1 for every 50 epochs to better observe the convergence phenomenon. Please refer to Appendix for more evaluation details on other datasets.

Multiple Social Interactions. Two agents in the middle of Figure 6(a) shows that Social-STGCNN predicts one agent moves faster than the other. In contrast, Social-SSL is able to preserve the intra- and inter-relationship by embedding the "neutral" interaction type and cross-sequence to sequence representation, which maintains a better speed and distance between these two agents. For the agent at the upper left, Social-STGCNN predicts the agent moving towards the direction of the two agents in the middle. In contrast, Social-SSL predicts the leaving behavior precisely, based on the past trajectory. This shows the effectiveness of closeness pretext since the agent at the upper left is too far away to interact with the agents in the middle.



(a) Multiple social interactions (b) Multiple group behaviors

Fig. 6. Visualization of results on ZARA1 and ZARA2 datasets. The endpoint of the arrow signifies the predicted final destination. In Figure 6(b), we mark the start-point and end-point for each group with different colors and take off the background scene information for better observations. For simplicity, we only compare one baseline at a time. Please refer to Appendix for more visualization cases.

Multiple Group Behaviors. Figure 6(b) compares the performance of STGAT and Social-SSL in a multi-group case. It can be observed that STGAT can capture the social interactions within a group precisely, *e.g.*, parallel walking. However, there is still a performance gap between STGAT and our method. Specifically, STGAT regards each agent as a node and models their relationship as a complete graph, making it hard to control the trajectory of each agent in a complex social situation with multiple group behaviors. As observed, behavior of one group might affect the behaviors of other groups that are not close, leading to inferior performance. In contrast, Social-SSL predicts the target agent's trajectory via the most effective social agent and can thus aggregate their interaction into independent groups. The closeness pretext in Social-SSL ensures that agents who are too far away to interact, will not be considered, making each group's behavior independent of other groups.

4.5 Ablation Study

To further evaluate the contribution of each pretext task, we compare the performance of Social-SSL trained on different combinations of pretext tasks in Table 4. Ours-NPS is the baseline that uses the "plain" structure of Transformer encoder and decoder with an end-to-end training, which is the same as Transformer TF [6]. However, since a single-agent model does not consider social information, we extend this structure to cross-sequence without any pre-training (Ours-NPC). Result shows that the performance of Ours-NPC is worse than Ours-NPS, due to the complexity of the cross-sequence input structure, causing it unable to learn the social information directly, without any proper supervision. Moreover, the performance of purely adding the social-related pretext tasks (Ours-MP) is close to Ours-NPC. This is because the social-related pretext tasks only teach the model to understand the relation without asking the model to use the social context for predicting the trajectory. However, adding social pretext tasks

and motion-related pretext improves Ours-SP by at least 45% in terms of ADE and FDE, demonstrating that the motion-related pretext (Ours-SP) plays an essential role in making social pretext tasks useful on the trajectory prediction task. Comparing results of the social-related pretexts, Ours-IP and Ours-CP, the interaction pretext is shown to be more important between the two social features. As expected, the closeness pretext plays a minor role in distinguishing which agent to be considered more. Results of Ours-SE and Ours-SP-SE show that segment embeddings are essential to enable the Trajectory Decoder to distinguish the target and social agents in the cross-sequence of the Social Encoder.

	Description	Mask	Interaction	Closeness	Segment Embedding	ADE/FDE
Ours-NPS	Single-agent baseline without pre-training				V	16.88/31.50
Ours-NPC	Cross-agent baseline without pre-training				V	26.46/48.45
Ours-MP	Social-SSL without motion-related pretext		V	V	V	26.81/46.72
Ours-SP	Social-SSL without social-related pretext	V			V	11.96/22.36
Ours-SP-SE	Ours-SP without segment embeddings	V				17.65/31.62
Ours-IP	Social-SSL without interaction pretext	V		V	V	11.27/20.94
Ours-CP	Social-SSL without closeness pretext	V	V		V	7.64/14.27
Ours-SE	Social-SSL without segment embeddings	V	V	V		9.23/17.75
Ours	Social-SSL	V	V	V	V	**6.63/12.23**

Table 4. Ablation study on pretext tasks using SDD dataset (Evaluated in pixels).

Acknowledgement

This work was supported in part by Ministry of Science and Technology of Taiwan under the grant numbers: MOST-109-2221-E-009-114-MY3, MOST-110-2221-E-A49-164, MOST-109-2223-E-009-002-MY3, MOST-110-2218-E-A49-018 and MOST-111-2634-F-007-002, as well as the partial support from QUALCOMM TAIWAN UNIVERSTIV RESEARCH 2021 PROGRAM (NYCU). We are grateful to the National Center for High-performance Computing for computer time and facilities.

5 Conclusions

This work presents Social-SSL that employed self-supervised cross-sequence learning based on Transformers, to learn better representations for performing the downstream multi-agent trajectory prediction task. The pre-training tasks of Social-SSL enhance the inter- and intra-relationship for cross-sequence representation. By designing the pretext tasks of interaction type prediction and closeness prediction, along with masked cross-sequence to sequence pre-training, Social-SSL can handle the problem of missing values in the target sequence and capture informative social information from the cross-sequence simultaneously. Quantitative and qualitative experiments show that Social-SSL outperforms state-of-the-art approaches across the benchmark datasets. Future work includes extending Social-SSL with multi-modality via architecture modifications and the design of new pretext tasks.

References

1. Alahi, A., et al.: Social lstm: Human trajectory prediction in crowded spaces. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 961–971 (2016)
2. Amirian, J., et al.: Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 2964–2972 (2019)
3. Choi, C., Choi, J.H., Li, J., Malla, S.: Shared cross-modal trajectory prediction for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 244–253 (2021)
4. Devlin, J., et al.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long and Short Papers). pp. 4171–4186 (2019)
5. Dong, L., et al.: Unified language model pre-training for natural language understanding and generation. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS). pp. 13063–13075 (2019)
6. Giuliari, F., Hasan, I., Cristani, M., Galasso, F.: Transformer networks for trajectory forecasting. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 10335–10342. IEEE (2021)
7. Gupta, A., et al.: Social gan: Socially acceptable trajectories with generative adversarial networks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2255–2264 (2018)
8. Helbing, D., et al.: Social force model for pedestrian dynamics. Physical review E **51**(5), 4282 (1995)
9. Hochreiter, S., et al.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Huang, Y., et al.: Stgat: Modeling spatial-temporal interactions for human trajectory prediction. In: IEEE/CVF International Conference on Computer Vision (ICCV). pp. 6271–6280 (2019)
11. Kalman, R.E.: A new approach to linear filtering and prediction problems. Transactions of the ASME–Journal of Basic Engineering **82**(Series D), 35–45 (1960)
12. Kipf, T.N., et al.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)
13. Kitani, K.M., et al.: Activity forecasting. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 201–214. Springer (2012)
14. Kosaraju, V., et al.: Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 137–146 (2019)
15. Kuderer, M., et al.: Feature-based prediction of trajectories for socially compliant navigation. In: Robotics: science and systems (2012)
16. Lerner, A., et al.: Crowds by example. In: Computer graphics forum. vol. 26, pp. 655–664. Wiley Online Library (2007)
17. Li, J., et al.: Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning. Advances in Neural Information Processing Systems (NeurIPS) (2020)
18. Li, L.L., et al.: End-to-end contextual perception and prediction with interaction transformer. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5784–5791 (2020)

19. Liang, J., et al.: Peeking into the future: Predicting future person activities and locations in videos. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5725–5734 (2019)
20. Liang, J., et al.: Simaug: Learning robust representations from simulation for trajectory prediction. In: European Conference on Computer Vision (ECCV). pp. 275–292 (2020)
21. Lipton, Z.C., et al.: A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019 (2015)
22. Loshchilov, I., et al.: Decoupled weight decay regularization. In: International Conference on Learning Representations (ICLR) (2019)
23. Mangalam, K., Girase, H., Agarwal, S., Lee, K.H., Adeli, E., Malik, J., Gaidon, A.: It is not the journey but the destination: Endpoint conditioned trajectory prediction. In: European Conference on Computer Vision (ECCV). pp. 759–776 (2020)
24. Møgelmose, A., et al.: Trajectory analysis and prediction for improved pedestrian safety: Integrated framework and evaluations. In: IEEE Intelligent Vehicles Symposium (IV). pp. 330–335. IEEE (2015)
25. Mohamed, A., et al.: Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 14424–14432 (2020)
26. Mohseni, S., et al.: Self-supervised learning for generalizable out-of-distribution detection. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). vol. 34, pp. 5216–5223 (2020)
27. Mohsenvand, M.N., et al.: Contrastive representation learning for electroencephalogram classification. In: Machine Learning for Health. pp. 238–253. (PMLR) (2020)
28. Monti, A., et al.: Dag-net: Double attentive graph neural network for trajectory forecasting. In: 25th International Conference on Pattern Recognition (ICPR) (2020)
29. Pang, B., et al.: Trajectory prediction with latent belief energy-based model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11814–11824 (2021)
30. Park, S.H., Lee, G., Seo, J., Bhat, M., Kang, M., Francis, J., Jadhav, A., Liang, P.P., Morency, L.P.: Diverse and admissible trajectory forecasting through multimodal context understanding. In: European Conference on Computer Vision (ECCV). pp. 282–298 (2020)
31. Pellegrini, S., et al.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: IEEE/CVF 12th International Conference on Computer Vision (ICCV). pp. 261–268 (2009)
32. Robicquet, A., et al.: Learning social etiquette: Human trajectory understanding in crowded scenes. In: European conference on computer vision (ECCV). pp. 549–565. Springer (2016)
33. Sadeghian, A., et al.: Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1349–1358 (2019)
34. Sadeghian, A., et al.: Car-net: Clairvoyant attentive recurrent network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 151–167 (2018)
35. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In: European Conference on Computer Vision (ECCV). pp. 683–700 (2020)

36. Shi, L., Wang, L., Long, C., Zhou, S., Zheng, F., Zheng, N., Hua, G.: Social interpretable tree for pedestrian trajectory prediction. In: AAAI Conference on Artificial Intelligence (AAAI) (2022)
37. Shi, L., et al.: Sgcn: Sparse graph convolution network for pedestrian trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8994–9003 (2021)
38. Shi, X., et al.: Social dpf: Socially acceptable distribution prediction of futures. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). vol. 35, pp. 2550–2557 (2021)
39. Song, H., Ding, W., Chen, Y., Shen, S., Wang, M.Y., Chen, Q.: Pip: Planning-informed trajectory prediction for autonomous driving. In: European Conference on Computer Vision (ECCV). pp. 598–614 (2020)
40. Song, K., et al.: Mass: Masked sequence to sequence pre-training for language generation. In: International Conference on Machine Learning (ICML). pp. 5926–5936 (2019)
41. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems (NeurIPS). pp. 5998–6008 (2017)
42. Wang, J.M., et al.: Gaussian process dynamical models for human motion. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) **30**(2), 283–298 (2008)
43. Yu, C., Ma, X., Ren, J., Zhao, H., Yi, S.: Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In: European Conference on Computer Vision (ECCV). pp. 507–523 (2020)
44. Yuan, Y., Weng, X., Ou, Y., Kitani, K.M.: Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9813–9823 (2021)