# Human Trajectory Prediction via Neural Social Physics-Supplementary Material

Jiangbei Yue[1], Dinesh Manocha[2], and He Wang[1](✉)

[1] University of Leeds, Leeds, UK
H.E.Wang@leeds.ac.uk
[2] University of Maryland at College Park, College Park, USA

## 1 Additional Experiments

### 1.1 Generalization to Unseen Scenarios

We use the collision rate to evaluate prediction plausibility. We first elaborate on the definition of the collision rate and then show more experimental results. Provided there are N agents in a scene, we consider their collision rates during a period of time such as 4.8 seconds which is widely used to evaluate trajectory predictions [2,1,4]. We count one collision if the minimum distance between two agents is smaller than 2r at any time, where r is the radius of a disc representing an agent. The maximum possible number of collisions is $N(N-1)/2$. The final collision rate is defined as:

$$R_{col} = \frac{M}{N(N-1)/2} \tag{1}$$

where $M$ is the number of collisions.

We show more results on the scene, coupa0, with different numbers of agents. We chose this scene because it is a relatively large space and can theoretically accommodate many people. The highest number of people simultaneously in the environment in the original data is merely 11. Therefore, this is a good scene to show how different methods can generalize to higher densities when learning from low density data.

In each experiment, the agents are randomly initialized with different initial positions, initial velocities and goals near the boundary of the scene, which is sufficient for our method to simulate. Therefore, we use NSP to predict trajectories of 30 seconds (t = 0 to 29) at FPS = 10 for all agents. We sample three intervals out of every trajectory, from t = 0 to 8, t = 4 to 12 and t = 8 to 16, where the density in the central area reaches the highest during t=8 to 16. For each interval (8 seconds long), we subsample at FPS = 2.5 to get 20 frames, where the first 8 frames are used as input for Y-net [1] and S-CSR [4]. The remaining 12 frames and the predictions (12 frames) of Y-net and S-CSR are used to calculate the collision rate. Before prediction, all methods are trained on the training dataset of SDD under the same setting explained in the main paper.

The results are shown in Table 1. We tested 50, 74, 100, 150 and 200 agents on the aforementioned three methods including ours. We can see that our method

**Table 1.** Collision rates of the generalization experiments on Coupa0. Results of Y-net, S-SCR and NSP on 50, 74, 100, 150 and 200 agents are shown in corresponding tables, where (1), (2) and (3) denote three intervals for calculating the collision rate.

(a) 50 Agents

| Methods | (1) | (2) | (3) | avg |
|---|---|---|---|---|
| Y-net | 2.8% | 2.9% | 3.8% | 3.2% |
| S-CSR | 2.5% | 1.7% | 1.9% | 2.0% |
| NSP(ours) | 0.6% | 0.6% | 0.6% | 0.6% |

(b) 74 Agents

| Methods | (1) | (2) | (3) | avg |
|---|---|---|---|---|
| Y-net | 3.8% | 4.8% | 3.0% | 3.9% |
| S-CSR | 0.9% | 0.9% | 1.5% | 1.1% |
| NSP(ours) | 0.2% | 0.3% | 0.5% | 0.3% |

(c) 100 Agents

| Methods | (1) | (2) | (3) | avg |
|---|---|---|---|---|
| Y-net | 4.2% | 5.2% | 7.6% | 5.7% |
| S-CSR | 0.9% | 1.1% | 0.8% | 0.9% |
| NSP(ours) | 0.3% | 0.7% | 0.4% | 0.5% |

(d) 150 Agents

| Methods | (1) | (2) | (3) | avg |
|---|---|---|---|---|
| Y-net | 4.9% | 4.0% | 3.4% | 4.1% |
| S-CSR | 0.6% | 1.0% | 1.7% | 1.1% |
| NSP(ours) | 0.2% | 0.5% | 1.0% | 0.6% |

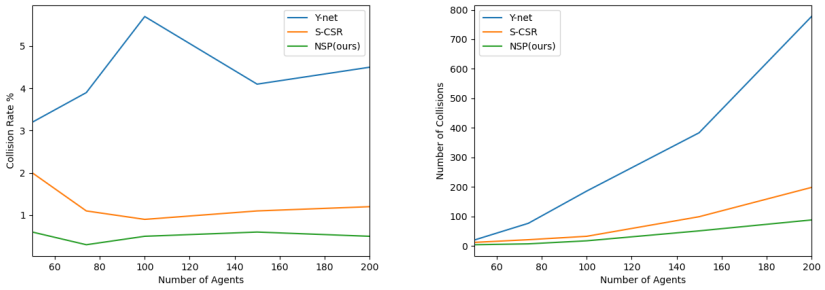
(e) 200 Agents

| Methods | (1) | (2) | (3) | avg |
|---|---|---|---|---|
| Y-net | 5.9% | 4.0% | 3.5% | 4.5% |
| S-CSR | 0.6% | 0.9% | 2.0% | 1.2% |
| NSP(ours) | 0.2% | 0.5% | 0.8% | 0.5% |

is always the best in the collision rate under different settings. Although its collision rate increases with the growth of the number of agents, our method is still the best compared with the baselines and our predictions are more plausible. In addition, we also plot the relation between the collision rate (and the number of collisions) and the agent number ranging from 50 to 200 in Figure 1. Y-net is worse than S-CSR and NSP. In addition, although the trend of NSP and S-CSR are similar, the number of collisions of S-CSR increases faster than NSP. Finally, some visualization results can be found in Figure 2. Here, every green disc has a radius of 7.5 pixels. When two green discs intersect, they collide with each other. Figure 2 demonstrates that our method (NSP) has better performance in avoiding collisions than Y-net and S-CSR.

### 1.2 Interpretability of Prediction

More examples of interpretability are shown in Figure 3. In Figure 3 (1)-(2), we show the influence of different three forces, $F_{goal}$, $F_{col}$ and $F_{env}$, on the whole trajectory of an agent. In Figure 3 (3)-(4), we choose two consecutive moments of one agent for analysis. In Figure 3 (1), instead of directly aiming for the goal, the agent suddenly turns (at the intersection between red and green dots) due to the incoming agents (the three blue dots under the green dots). The result is a result of major influence from $F_{goal}$ and $F_{col}$. Similarly, the agent in Figure 3 (2) did not need to avoid other agents but still did not directly walk towards the goal, because of $F_{env}$ from the grass. In Figure 3 (3)-(4), we show the detailed analysis




(a) Collision Rate (b) Number of Collisions

**Fig. 1.** The collision rate and the number of collisions against the number of agents are shown in (a) and (b) respectively. Both of horizontal axes represent the number of agents from 50 to 200. The vertical axes in (a) and (b) represent the collision rate and the number of collisions respectively.

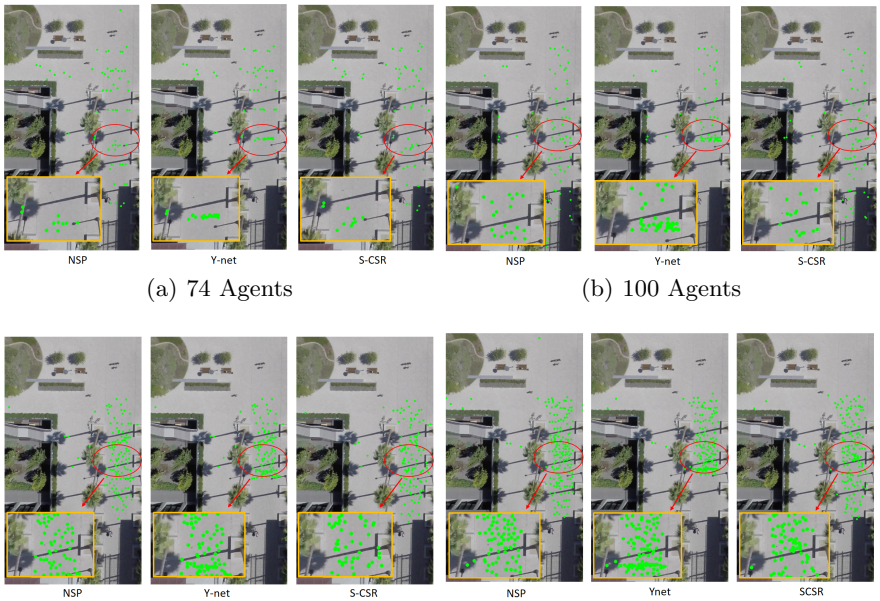
**Table 2.** Ablation experiments on network architecture. Goal-Network and the Collision-Network possess the same architecture under each experimental setup.

| ADE | Two layers MLP | Full MLP |
|---|---|---|
| w/o LSTM | 6.83 | 6.61 |
| with LSTM | 6.66 | **6.52** |

of forces at two consecutive time steps of the same agent, where $F_{env}$ is from the lawn which is a 'weakly repulsive area'. More examples where randomness is captured by our model are shown in Figure 4.

### 1.3 Ablation Experiments

We conduct more ablation experiments to further validate our design decisions and explore the effect of components of our model. The ablation studies on the network architectures focuses on the Goal-Network and the Collision-Network. The main variants are with/without LSTM to show the importance of the temporal modeling for learning $\tau$ and $k_{nj}$, and replacing the MLPs with simple two-layer MLPs. Table 2 shows the results on SDD. We can see that the temporal modeling and the original MLPs make our model achieve the best performance. To understand the role of each component in our model, we take social force model (SFM) as the baseline and incrementally add components from our model. The results are shown in Table 3. We tried our best to manually find good parameter values: $\tau = 0.5$, $k_{nj}$=25/50 and $k_{env}$=65. We adopted the same way with our model to sample destinations for SFM. Then we only learn $\tau$ and $k_{nj}$. At last, the result of the full model without CVAE is given. The performance is better when more components are added.




(a) 74 Agents (b) 100 Agents

(c) 150 Agents (d) 200 Agents

**Fig. 2.** The visualization results of generalization to 74, 100, 150 and 200 agents on coupa0 are shown in (a), (b), (c) and (d) respectively. For each experimental setting, visualization results of NSP, Y-net and S-CSR are at the same frame. We amplify the area of red ellipse to boxes with yellow borders for better visualization performance.

## 2 Details of the Neural Social Physics Model

In this section, we elaborate the details of the Goal Sampling Network (GSN) and the conditional Variational Autoencoder (CVAE) in our model.

### 2.1 Goal Sampling Network

The main components of the GSN are two U-nets [3] as illustrated in Figure 5. We first feed the scene image $\boldsymbol{I}$ to a U-net, $U_{seg}$, to get its corresponding environment pixel-wise segmentation with dimension of $H * W * K_c$. $H$ and $W$ are the height and width of $\boldsymbol{I}$, and $K_c$ is the number of classes for segmentation. The segmantation maps are byproducts of the GSN from [1]. NSP can use manually annotated or automatically segmented environment maps to calculate $F_{env}$, but using segmentation maps from the GSN is more efficient. Then the past trajectories $\{p^t\}_{t=0}^{M}$ are converted into M+1 trajectory heatmaps by:

$$Hm(t,i,j) = 2\frac{\|(i,j) - p^t\|}{\max\limits_{(x,y)\in \boldsymbol{I}} \|(x,y) - p^t\|} \tag{2}$$

**Table 3.** Ablation experiments on SDD. Different components from our model are added incrementally

| $k_{nj}$=25 | hand-tuned | learned $\tau$ and $k_{nj}$ | NSP |
|---|---|---|---|
| ADE | 8.32 | 6.53 | 6.52 |
| FDE | 10.97 | 10.61 | 10.61 |
| $k_{nj}$=50 | hand-tuned | learned $\tau$ and $k_{nj}$ | NSP |
| ADE | 7.54 | 6.53 | 6.52 |
| FDE | 10.81 | 10.61 | 10.61 |

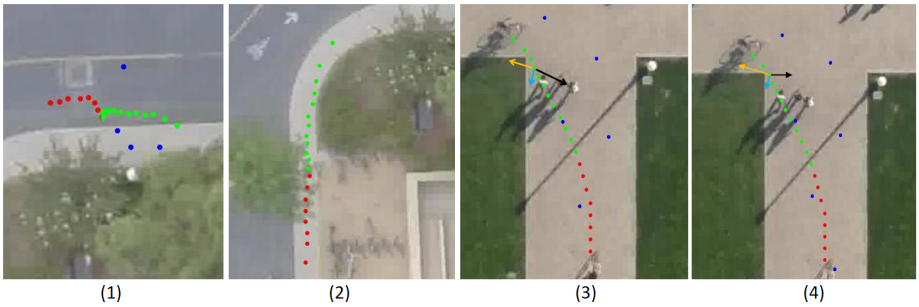



**Fig. 3.** Examples of interpretability. Red dots are observed, green dots are our prediction. Bule dots in (1), (3) and (4) are other pedestrians at time step 7, 16 and 17 respectively. We show the influence of all forces, $F_{goal}$, $F_{col}$ and $F_{env}$, on the whole trajecroty in (1) and (2). We display detailed analysis of three forces at two consecutive time steps of the same agent, where $F_{goal}$, $F_{col}$ and $F_{env}$ are shown as yellow, light blue and black arrows respectively.

where $(i, j)$ is the pixel coordinate on the heatmap and $(x, y)$ is the pixel coordinate on the scene image $\boldsymbol{I}$. Then, we concatenate these trajectory heatmaps and the segmentation map to get the input with dimension of $H*W*(K_c+M+1)$ for the network $U_{goal}$. $U_{goal}$ will output a non-parametric probability distribution map, $\tilde{D}_{goal}$, with dimensions $H * W$. Every pixel in $\tilde{D}_{goal}$ has a corresponding probability value between 0 and 1, and their sum is equal to 1. Details of these two U-nets can be found in [1]. We train the GSN by minimizing the Kullback–Leibler divergence between predicted $\tilde{D}_{goal}$ and its ground truth $D_{goal}$. We assume that $D_{goal}$ is a discrete gaussian distribution with a mean at the position of the ground-truth goal and a hyper-parameter variance $\sigma_{goal}$. During testing, instead of picking the position with highest probability, we adopt the test-time sampling trick (TTST) introduced by [1] to sample goals for better performance.

### 2.2 Conditional Variational Autoencoder.

We model the dynamics stochasticity for each agent individually by using a CVAE as illustrated in Figure 6. Red connections are only used in the training

**Fig. 4.** Motion randomness is captured by our model. Red dots are observed, green dots are our prediction and black dots are the ground-truth.

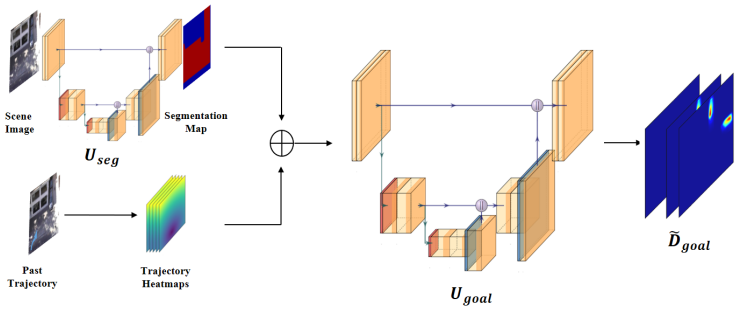



**Fig. 5.** Model Architecture of Goal Sampling Network. The detailed network architecture of two U-nets, $U_{seg}$ and $U_{goal}$, can be found in [1].

phase. Given an agent $p^t$ and his/her destination, a deterministic prediction $\bar{p}^{t+1}$ without dynamics stochasticity is first calculated from $F_{goal}$, $F_{col}$ and $F_{env}$ and a semi-implicit scheme. During training time, we use the corresponding ground truth $p^{t+1}$ to calculate the error $\alpha^{t+1} = p^{t+1} - \bar{p}^{t+1}$, and feed $\alpha^{t+1}$ into an encoder $E_{bias}$ to get the feature $f_{bias}$. The brief history $(p^{t-7}, \ldots, p^{t-1}, p^t)$ is encoded as $f_{past}$ by using an encoder $E_{past}$. We concatenate $f_{bias}$ with $f_{past}$ and encode it using a latent encoder to yield the parameters $(\mu, \sigma)$ of the gaussian distribution of the latent variable Z. We sample Z, concatenate it with $f_{past}$ for history information, and decode using the decoder $D_{latent}$ to acquire our guess for stochasticity $\tilde{\alpha}^{t+1}$. Finally, the estimated stochasticity will be added to the deterministic prediction $\bar{p}^{t+1}$ to get our final prediction $\tilde{p}^{t+1}$. During testing time, the ground truth $p^{t+1}$ is unavailable. Therefore, we sample the latent variable Z from a gaussian distribution $N(0, \sigma_{latent} I)$ where $\sigma_{latent}$ is a hyperparameter. We concatenate the sampled Z and $f_{past}$ to decode directly using the learned decoder $D_{latent}$ to get the estimate of stochasticity $\tilde{\alpha}^{t+1}$. We can produce final prediction $\tilde{p}^{t+1}$ using the same way as the training phase. Encoders $E_{bias}$, $E_{past}$, $E_{latent}$ and the decoder $D_{latent}$ are all multi-layer perceptrons (MLP) with dimensions indicated in the square brackets in Figure 6.

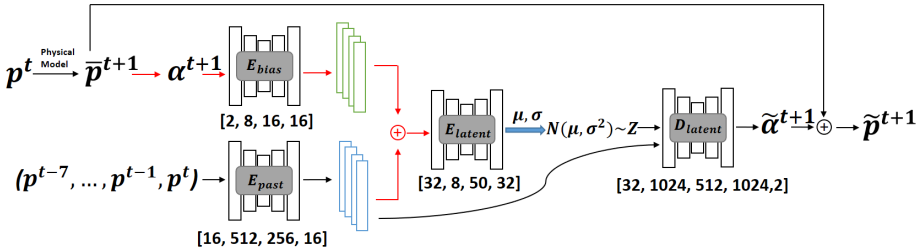



**Fig. 6.** The architecture of the CVAE, where $\bar{p}^{t+1}$ is the intermediate prediction out of our force model and $\alpha^{t+1} = p^{t+1} - \bar{p}^{t+1}$. Encoder $E_{bias}$, $E_{past}$, $E_{latent}$ and decoder $D_{latent}$ are all MLP networks with dimensions indicated in the square brackets. Red connections are only used in the training phase.

## 3 Implementation Details

We use ADAM as the optimizer to train the Goal-Network, Collision-Network and $F_{env}$ with a learning rate between $3 \times 10^{-5}$ and $3 \times 10^{-4}$, and to train the CVAE with a learning rate between $3 \times 10^{-6}$ and $3 \times 10^{-5}$. When we train the CVAE of our model, the training data is scaled by 0.005 to balance reconstruction error and KL-divergence in $l_{cvae}$. The hyper-parameter $\lambda$ in $l_{cvae}$ is set to 1. Concrete structures of all sub-network are shown in Figure 6.

For the Goal-Network, instead of learning parameter $\tau$ directly, we set $\tau = a * sigmoind(NN_{\phi_1}(q^t, p^T)) + b$ where $a$ and $b$ are hyper-parameters. We list all hyper-parameters of our model in Table 4. We segment scene images into two classes and three classes on ETH/UCY and SDD, respectively. The two classes on ETH/UCY are 'walkable area' and 'unwalkable area'. Three classes on SDD include 'walkable area', 'unwalkable area' and 'weakly repulsive area' that some people tend to avoid such as lawns. The calculation of $F_{env}$ on ETH/UCY has been introduced in our main paper. On SDD, we calculate the position of the obstacle $p_{obs}$ and the position of the weak obstacle $p_{w-obs}$ (i.e. in the weakly repulsive area) by averaging pixels that are classified as 'unwalkable area' and 'weak repulsive area' respectively. Then, the $F_{env}$ consists of two repulsive forces from $p_{obs}$ and $p_{w-obs}$ as shown in Equation 3, where the parameter $k_{env}$ is shared and an additional hyper-parameter $\lambda_{weak}$ is introduced for $p_{w-obs}$:

$$F_{env} = \frac{k_{env}}{\|p_n^t - p_{obs}\|}\left(\frac{p_n^t - p_{obs}}{\|p_n^t - p_{obs}\|}\right) + \frac{\lambda_{weak} k_{env}}{\|p_n^t - p_{w-obs}\|}\left(\frac{p_n^t - p_{w-obs}}{\|p_n^t - p_{w-obs}\|}\right) \quad (3)$$

## References

1. Mangalam, K., An, Y., Girase, H., Malik, J.: From goals, waypoints & paths to long term human trajectory forecasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15233–15242 (2021)

**Table 4.** Hyper-parameters for all six datasets.

| Hyper-Para | ETH | Hotel | UNIV | ZARA1 | ZARA2 | SDD |
|---|---|---|---|---|---|---|
| a($\tau$) | 1 | 1 | 1 | 1 | 1 | 1 |
| b($\tau$) | 0.1 | 0.1 | 2.2 | 1.6 | 1.4 | 0.4 |
| a($k_{nj}$) | 50 | 50 | 50 | 50 | 50 | 100 |
| b($k_{nj}$) | 0 | 0 | 0 | 0 | 0 | 0 |
| $\omega$ | $\pi/3$ | $\pi/3$ | $\pi/3$ | $\pi/3$ | $\pi/3$ | $\pi/3$ |
| $r_{col}$ | 75 | 75 | 75 | 75 | 75 | 100 |
| $r_{env}$ | 50 | 50 | 50 | 75 | 75 | 50 |
| $\sigma_{goal}$ | 4 | 4 | 4 | 4 | 4 | 4 |
| $\sigma_{latent}$ | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 |
| $\lambda_{weak}$ | N/A | N/A | N/A | N/A | N/A | 0.2 |

2. 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. pp. 759–776. Springer (2020)
3. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
4. Zhou, H., Ren, D., Yang, X., Fan, M., Huang, H.: Sliding sequential cvae with time variant socially-aware rethinking for trajectory prediction. arXiv preprint arXiv:2110.15016 (2021)