Social LSTM:
Human Trajectory Prediction in Crowded Spaces

Alexandre Alahi*, Kratarth Goel*, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, Silvio Savarese
Stanford University
{alahi,kratarth,vigneshr,arobicqu,feifeili,ssilvio}@cs.stanford.edu

Abstract

Pedestrians follow different trajectories to avoid obstacles and accommodate fellow pedestrians. Any autonomous vehicle navigating such a scene should be able to foresee the future positions of pedestrians and accordingly adjust its path to avoid collisions. This problem of trajectory prediction can be viewed as a sequence generation task, where we are interested in predicting the future trajectory of people based on their past positions. Following the recent success of Recurrent Neural Network (RNN) models for sequence prediction tasks, we propose an LSTM model which can learn general human movement and predict their future trajectories. This is in contrast to traditional approaches which use hand-crafted functions such as Social forces. We demonstrate the performance of our method on several public datasets. Our model outperforms state-of-the-art methods on some of these datasets. We also analyze the trajectories predicted by our model to demonstrate the motion behaviour learned by our model.

1. Introduction

Humans have the innate ability to “read” one another. When people walk in a crowded public space such as a sidewalk, an airport terminal, or a shopping mall, they obey a large number of (unwritten) common sense rules and comply with social conventions. For instance, as they consider where to move next, they respect personal space and yield right-of-way. The ability to model these rules and use them to understand and predict human motion in complex real world environments is extremely valuable for a wide range of applications - from the deployment of socially-aware robots [41] to the design of intelligent tracking systems [43] in smart environments.

Predicting the motion of human targets while taking into account such common sense behavior, however, is an extremely challenging problem. This requires understanding the complex and often subtle interactions that take place between people in crowded spaces. Recent research in computer vision has successfully addressed some of these challenges. Kitani et al. [32] have demonstrated that the inferred knowledge about the semantics of the static environment (e.g., location of sidewalks, extension of grass areas, etc) helps predict the trajectory of pedestrians in future instants more accurately than a model which ignores the scene information. Pioneering works by [24, 50, 35] have also proposed ways to model human-human interactions (often called “social forces”) to increase robustness and accuracy in multi-target tracking problems.

However, most of these works are limited by the following two assumptions. i) They use hand-crafted functions to model “interactions” for specific settings rather than inferring them in a data-driven fashion. This results in fa-
voring models that capture simple interactions (e.g. repulsion/attractons) and might fail to generalize for more complex crowded settings. ii) They focus on modeling interactions among people in close proximity to each other (to avoid immediate collisions). However, they do not anticipate interactions that could occur in the more distant future.

In this work, we propose an approach that can address both challenges through a novel data-driven architecture for predicting human trajectories in future instants. Inspired by the recent success of Long-Short Term Memory networks (LSTM) for different sequence prediction tasks such as handwriting [20] and speech [21] generation, we extend them for human trajectory prediction as well. While LSTMs have the ability to learn and reproduce long sequences, they do not capture dependencies between multiple correlated sequences.

We address this issue through a novel architecture which connects the LSTMs corresponding to nearby sequences. In particular, we introduce a “Social” pooling layer which allows the LSTMs of spatially proximal sequences to share their hidden-states with each other. This architecture, which we refer to as the “Social-LSTM”, can automatically learn typical interactions that take place among trajectories which coincide in time. This model leverages existing human trajectory datasets without the need for any additional annotations to learn common sense rules and conventions that humans observe in social spaces.

Finally, we demonstrate that our Social-LSTM is capable of predicting trajectories of pedestrians much more accurately than state-of-the-art methods on two publicly available datasets: ETH [49], and UCY [39]. We also analyze the trajectory patterns generated by our model to understand the social constraints learned from the trajectory datasets.

2. Related work

Human-human interactions Pioneering work from Helbing and Molnar [24] presented a pedestrian motion model with attractive and repulsive forces referred to as the Social Force model. This has been shown to achieve competitive results even on modern pedestrian datasets [39, 49]. This method was later extended to robotics [41] and activity understanding [43, 73, 50, 38, 37, 9, 10].

Similar approaches have been used to model human-human interactions with strong priors for the model. Treuille et al. [62] use continuum dynamics, Antonini et. al. [2] propose a Discrete Choice framework and Wang et. al. [69], Tay et. al. [59] use Gaussian processes. Such functions have also been used to study stationary groups [74, 48]. These works target smooth motion paths and do not handle the problems associated with discretization.

Another line of work uses well-engineered features and attributes to improve tracking and forecasting. Alahi et. al. [1] presented a social affinity feature by learning from human trajectories in crowd their relative positions, while Yu et. al. [74] proposed the use of human-attributes to improve forecasting in dense crowds. They also use an agent-based model similar to [6]. Rodriguez et al. [54] analyze videos with high-density crowds to track and count people.

Most of these models provide hand-crafted energy potentials based on relative distances and rules for specific scenes. In contrast, we propose a method to learn human-human interactions in a more generic data-driven fashion.

Activity forecasting Activity forecasting models try to predict the motion and/or action to be carried out by people in a video. A large body of work learns motion patterns through clustering trajectories [26, 30, 46, 77]. More approaches can be found in [45, 52, 34, 3, 16, 33]. Kitani et. al. in [32] use Inverse Reinforcement Learning to predict human paths in static scenes. They infer walkable paths in a scene by modeling human-space interactions. Walker et al. in [68] predict the behavior of generic agents (e.g., a vehicle) in a visual scene given a large collection of videos. Ziebart et al. [78, 23] presented a planning based approach.

Turek et al. [63, 40] used a similar idea to identify the functional map of a scene. Other approaches like [27, 19, 42, 36] showed the use of scene semantics to predict goals and paths for human navigation. Scene semantics has also been used to predict multiple object dynamics [17, 36, 34, 28]. These works are mostly restricted to the use of static scene information to predict human motion or activity. In our work, we focus on modeling dynamic crowd interactions for path prediction.

More recent works have also attempted to predict future human actions. In particular, Ryoo et. al. [55, 8, 71, 67, 44, 58] forecast actions in streaming videos. More relevant to our work, is the idea of using a RNN model to predict future events in videos [53, 57, 66, 56, 31]. Along similar lines, we predict future trajectories in scenes.

RNN models for sequence prediction Recently Recurrent Neural Networks (RNN) and their variants including Long Short Term Memory (LSTM) [25] and Gated Recurrent Units [12] have proven to be very successful for sequence prediction tasks: speech recognition [21, 11, 13], caption generation [64, 29, 75, 15, 72], machine translation [4], image/video classification [7, 22, 70, 47], human dynamics [18] to name a few. RNN models have also proven to be effective for tasks with densely connected data such as semantic segmentation [76], scene parsing [51] and even as an alternative to Convolutional Neural Networks [65]. These works show that RNN models are capable of learning the dependencies between spatially correlated data such as image pixels. This motivates us to extend the sequence generation model from Graves et al. [20] to our setting. In particular, Graves et al. [20] predict isolated handwriting
sequences; while in our work we jointly predict multiple correlated sequences corresponding to human trajectories.

3. Our model

Humans moving in crowded scenes adapt their motion based on the behaviour of other people in their vicinity. For instance, a person could completely alter his/her path or stop momentarily to accommodate a group of people moving towards him. Such deviation in trajectory cannot be predicted by observing the person in isolation. Neither, can it be predicted with simple “repulsion” or “attraction” functions (the traditional social forces models [24, 43, 73, 50])

This motivates us to build a model which can account for the behavior of other people within a large neighborhood, while predicting a person’s path. In this section, we describe our pooling based LSTM model (Fig. 2) which jointly predicts the trajectories of all the people in a scene. We refer to this as the “Social” LSTM model.

Problem formulation We assume that each scene is first preprocessed to obtain the spatial coordinates of all people at different time-instants. Previous work follow this convention as well [41, 1]. At any time-instant $t$, the $i$th person in the scene is represented by his/her xy-coordinates $(x_i^t, y_i^t)$. We observe the positions of all the people from time 1 to $T_{obs}$, and predict their positions for time instants $T_{obs}+1$ to $T_{pred}$. This task can also be viewed as a sequence generation problem [20], where the input sequence corresponds to the observed positions of a person and we are interested in generating an output sequence denoting his/her future positions at different time-instants.

3.1. Social LSTM

Every person has a different motion pattern: they move with different velocities, acceleration and have different gaits. We need a model which can understand and learn such person-specific motion properties from a limited set of initial observations corresponding to the person.

Long Short-Term Memory (LSTM) networks have been shown to successfully learn and generalize the properties of isolated sequences like handwriting [20] and speech [21]. Inspired by this, we develop a LSTM based model for our trajectory prediction problem as well. In particular, we have one LSTM for each person in a scene. This LSTM learns the state of the person and predicts their future positions as shown in Fig. 2. The LSTM weights are shared across all the sequences.

However, the naive use of one LSTM model per person does not capture the interaction of people in a neighborhood. The vanilla LSTM is agnostic to the behaviour of other sequences. We address this limitation by connecting neighboring LSTMs through a new pooling strategy visualized in Fig. 3.2.

Social pooling of hidden states Individuals adjust their paths by implicitly reasoning about the motion of neighboring people. These neighbors in-turn are influenced by others in their immediate surroundings and could alter their behaviour over time. We expect the hidden states of an LSTM to capture these time varying motion-properties. In order to jointly reason across multiple people, we share the states between neighboring LSTMs. This introduces a new challenge: every person has a different number of neighbors and in very dense crowds [1], this number could be prohibitively high.

Hence, we need a compact representation which combines the information from all neighboring states. We handle this by introducing “Social” pooling layers as shown in Fig. 2. At every time-step, the LSTM cell receives pooled hidden-state information from the LSTM cells of neighbors.
While pooling the information, we try to preserve the spatial information through grid based pooling as explained below. The hidden state $h^t_i$ of the LSTM at time $t$ captures the latent representation of the $i^{th}$ person in the scene at that instant. We share this representation with neighbors by building a “Social” hidden-state tensor $H^t_i$. Given a hidden-state dimension $D$, and neighborhood size $N_o$, we construct a $N_o \times N_o \times D$ tensor $H^t_i$ for the $i^{th}$ trajectory:

$$H^t_i(m,n,:) = \sum_{j \in N_i} 1_{mn}[x^*_j - x^*_i, y^*_j - y^*_i]h^t_{i-1}, \quad (1)$$

where $h^t_{i-1}$ is the hidden state of the LSTM corresponding to the $i^{th}$ person at $t-1$, $1_{mn}[x, y]$ is an indicator function to check if $(x, y)$ is in the $(m, n)$ cell of the grid, and $N_i$ is the set of neighbors corresponding to person $i$. This pooling operation is visualized in Fig. 3.

We embed the pooled Social hidden-state tensor into a vector $a^t_i$ and the co-ordinates into $e^t_i$. These embeddings are concatenated and used as the input to the LSTM cell of the corresponding trajectory at time $t$. This introduces the following recurrence:

$$e^t_i = \phi(x^*_i, y^*_i; W_e) \quad (2)$$

$$a^t_i = \phi(H^t_i; W_a),$$

$$h^t_i = \text{LSTM} (h^t_{i-1}, e^t_i, a^t_i; W_l)$$

where $\phi(.)$ is an embedding function with ReLU non-linearity, $W_e$ and $W_a$ are embedding weights. The LSTM weights are denoted by $W_l$.

**Position estimation** The hidden-state at time $t$ is used to predict the distribution of the trajectory position $(\hat{x}, \hat{y})^{t+1}_i$ at the next time-step $t+1$. Similar to Graves et al. [20], we assume a bivariate Gaussian distribution parametrized by the mean $\mu^t_{i+1} = (\mu_x, \mu_y)^{t+1}_i$, standard deviation $\sigma^t_{i+1} = (\sigma_x, \sigma_y)^{t+1}_i$ and correlation coefficient $\rho^t_{i+1}$. These parameters are predicted by a linear layer with a $5 \times D$ weight matrix $W_p$. The predicted coordinates $(\hat{x}^*_i, \hat{y}^*_i)$ at time $t$ are given by

$$(\hat{x}, \hat{y})^{t}_i \sim \mathcal{N}(\mu^t_{i+1}, \sigma^t_{i+1}, \rho^t_{i+1}) \quad (3)$$

The parameters of the LSTM model are learned by minimizing the negative log-Likelihood loss ($L^t$ for the $i^{th}$ trajectory):

$$[\mu^t_{i}, \sigma^t_{i}, \rho^t_{i}] = W_p h^t_{i-1} \quad (4)$$

$$L^t(W_e, W_l, W_p) = - \sum_{t=T_{obs}+1}^{T_{pred}} \log \left( \mathbb{P}(x^*_i, y^*_i | \sigma^t_{i}, \mu^t_{i}, \rho^t_{i}) \right),$$

where $1_{mn}[]$ is an indicator function as defined previously. This can also be viewed as a simplification of the social tensor in Eq. 1 where the hidden state vector is replaced by a constant value indicating the presence or absence of neighbors in the corresponding cell.

The vectorized occupancy map is used in place of $H^t_i$ in Eq. 2 while learning this simpler model.

**Inference for path prediction** During test time, we use the trained Social-LSTM models to predict the future position $(\hat{x}^*_i, \hat{y}^*_i)$ of the $i^{th}$ person. From time $T_{obs}+1$ to $T_{pred}$,
we use the predicted position \((\hat{x}_i^t, \hat{y}_i^t)\) from the previous Social-LSTM cell in place of the true coordinates \((x_i^t, y_i^t)\) in Eq. 2. The predicted positions are also used to replace the actual coordinates while constructing the Social hidden-state tensor \(H_i^t\) in Eq. 1 or the occupancy map \(O_i^t\) in Eq. 5.

3.2. Implementation details

We use an embedding dimension of 64 for the spatial coordinates before using them as input to the LSTM. We set the spatial pooling size \(N_o\), to be 32 and use a 8x8 sum pooling window size without overlaps. We used a fixed hidden state dimension of 128 for all the LSTM models. Additionally, we also use an embedding layer with ReLU (rectified Linear Units) non-linearity on top of the pooled hidden-state features, before using them for calculating the hidden state tensor \(H_i^t\). The hyper-parameters were chosen based on cross-validation on a synthetic dataset. This synthetic was generated using a simulation that implemented the social forces model. This synthetic data contained trajectories for hundreds of scenes with an average crowd density of 30 per frame. We used a learning rate of 0.003 and RMS-prop [14] for training the model. The Social-LSTM model was trained on a single GPU with a Theano [5] implementation.

4. Experiments

In this section, we present experiments on two publicly available human-trajectory datasets: ETH [49] and UCY [39]. The ETH dataset contains two scenes each with 750 different pedestrians and is split into two sets (ETH and Hotel). The UCY dataset contains two scenes with 786 people. This dataset has 3-components: ZARA-01, ZARA-02 and UCY. In total, we evaluate our model on 5 sets of data. These datasets represent real world crowded settings with thousands of non-linear trajectories. As shown in [49], these datasets also cover challenging group behaviours such as couples walking together, groups crossing each other and groups forming and dispersing in some scenes.

We report the prediction error with three different metrics. Similar to Pellegrini et al. [49] we use:

1. **Average displacement error** - The mean square error (MSE) over all estimated points of a trajectory and the true points. This was introduced in Pellegrini et al. [49].

2. **Final displacement error** - The distance between the predicted final destination and the true final destination at end of the prediction period \(T_{pred}\).

3. **Average non-linear displacement error** - The is the MSE at the non-linear regions of a trajectory. Since most errors in trajectory-prediction occur during non-linear turns arising from human-human interactions, we explicitly evaluate the errors around these regions. We set a heuristic threshold on the norm of the second derivative to identify non-linear regions.

In order to make full use of the datasets while training our models, we use a leave-one-out approach. We train and validate our model on 4 sets and test on the remaining set. We repeat this for all the 5 sets. We also use the same training and testing procedure for other baseline methods used for comparison.

During test time, we observe a trajectory for 3.2secs and predict their paths for the next 4.8secs. At a frame rate of 0.4, this corresponds to observing 8 frames and predicting for the next 12 frames. This is similar to the setting used by [49, 39]. In Tab. 4, we compare the performance of our model with state-of-the-art methods as well as multiple control settings:

- **Linear model (Lin.)** We use an off-the-shelf Kalman filter to extrapolate trajectories with assumption of linear acceleration.

- **Collision avoidance (LTA).** We report the results of a simplified version of the Social Force [73] model which only uses the collision avoidance energy, commonly referred to as linear trajectory avoidance.

- **Social force (SF).** We use the implementation of the Social Force model from [73] where several factors such as group affinity and predicted destinations have been modeled.

- **Iterative Gaussian Process (IGP).** We use the implementation of the IGP from [61]. Unlike the other baselines, IGP also uses additional information about the final destination of a person.

- **Our Vanilla LSTM (LSTM).** This is a simplified setting of our model where we remove the “Social” pooling layers and treat all the trajectories to be independent of each other.

- **Our LSTM with occupancy maps (O-LSTM).** We show the performance of a simplified version of our model (presented in Sec. 3.1). As a reminder, the model only pools the coordinates of the neighbors at every time-instance.

The naive linear model produces high prediction errors, which are more pronounced around non-linear regions as seen from the average non-linear displacement error. The vanilla LSTM outperforms this linear baseline since it can extrapolate non-linear curves as shown in Graves et al. [20]. However, this simple LSTM is noticeably worse than the Social Force and IGP models which explicitly model
human-human interactions. This shows the need to account for such interactions.

Our Social pooling based LSTM and O-LSTM outperform the heavily engineered Social Force and IGP models in almost all datasets. In particular, the error reduction is more significant in the case of the UCY datasets as compared to ETH. This can be explained by the different crowd densities in the two datasets: UCY contains more crowded regions with a total of $32K$ non-linearities as opposed to the more sparsely populated ETH scenes with only $15K$ non-linear regions.

In the more crowded UCY scenes, the deviation from linear paths is more dominated by human-human interactions. Hence, our model which captures neighborhood interactions achieves a higher gain in UCY datasets. The pedestrians’ intention to reach a certain destination plays a more dominant role in the ETH datasets. Consequently, the IGP model which knows the true final destination during testing achieves lower errors in parts of this dataset.

In the case of ETH, we also observe that the occupancy and Social LSTM errors are at par with each other and in general better than the Social force model. Again, our Social-LSTM outperforms O-LSTM in the more crowded UCY datasets. This shows the advantage of pooling the entire hidden state to capture complex interactions in dense crowds.

### 4.1. Analyzing the predicted paths

Our quantitative evaluation in the Sec. 4 shows that the learned Social-LSTM model outperforms state-of-the-art methods on standard datasets. In this section, we try to gain more insights on the actual behaviour of our model in different crowd settings. We qualitatively study the performance of our Social-LSTM model on social scenes where individuals interact with each others in a specific pattern.

We present an example scene occupied by four individuals in Figure 4. We visualize the distribution of the paths predicted by our model at different time-instants. The first and third rows in Figure 4 show the current position of each person as well as their true trajectory (solid line for the future path and dashed line for the past). The second and fourth rows show our Social-LSTM prediction for the next 12.4 secs. In these scenes, we observe three people(2,3,4) walking close to each other and a fourth person(1) walking farther away from them.

Our model predicts a linear path for person(1) at all times. The distribution for person (1) is similar across time indicating that the speed of the person is constant. We can observe more interesting patterns in the predicted trajectories for the 3-person group. In particular, our model makes intelligent route choices to yield for others and preempt future collisions. For instance, at time-steps 2, 4, and 5 our model predicts a deviation from the linear paths for person(3) and person(4), even before the start of the actual turn.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lin</td>
<td>0.80</td>
<td>0.39</td>
<td>0.47</td>
<td>0.45</td>
<td>0.57</td>
<td>0.53</td>
</tr>
<tr>
<td></td>
<td>LTA</td>
<td>0.54</td>
<td>0.38</td>
<td>0.37</td>
<td>0.40</td>
<td>0.51</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>SF [73]</td>
<td>0.41</td>
<td>0.25</td>
<td>0.40</td>
<td>0.40</td>
<td>0.48</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td>IGP* [60]</td>
<td><strong>0.20</strong></td>
<td><strong>0.24</strong></td>
<td><strong>0.43</strong></td>
<td><strong>0.41</strong></td>
<td><strong>0.61</strong></td>
<td><strong>0.37</strong></td>
</tr>
<tr>
<td></td>
<td>LSTM</td>
<td>0.60</td>
<td>0.15</td>
<td>0.43</td>
<td>0.51</td>
<td>0.52</td>
<td>0.44</td>
</tr>
<tr>
<td>Avg. disp. error</td>
<td>our O-LSTM</td>
<td>0.49</td>
<td><strong>0.09</strong></td>
<td><strong>0.22</strong></td>
<td><strong>0.28</strong></td>
<td>0.35</td>
<td><strong>0.28</strong></td>
</tr>
<tr>
<td></td>
<td>our Social-LSTM</td>
<td>0.50</td>
<td><strong>0.11</strong></td>
<td><strong>0.22</strong></td>
<td><strong>0.25</strong></td>
<td><strong>0.27</strong></td>
<td><strong>0.27</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ETH [49]</td>
<td>0.95</td>
<td>0.55</td>
<td>0.56</td>
<td>0.44</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td>Avg. non-linear disp. error</td>
<td>HOTEL [49]</td>
<td>0.70</td>
<td>0.49</td>
<td>0.39</td>
<td>0.41</td>
<td>0.57</td>
<td>0.51</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.49</td>
<td>0.38</td>
<td>0.41</td>
<td>0.39</td>
<td>0.44</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>ZARA 1 [39]</td>
<td>0.49</td>
<td>0.34</td>
<td>0.54</td>
<td>0.43</td>
<td>0.62</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>ZARA 2 [39]</td>
<td>0.41</td>
<td>0.34</td>
<td>0.54</td>
<td>0.43</td>
<td>0.62</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>UCY [39]</td>
<td>0.57</td>
<td>0.34</td>
<td>0.41</td>
<td>0.39</td>
<td>0.41</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>0.53</td>
<td>0.34</td>
<td>0.41</td>
<td>0.39</td>
<td>0.41</td>
<td>0.34</td>
</tr>
<tr>
<td>Average</td>
<td>ETH [49]</td>
<td>1.31</td>
<td>0.55</td>
<td>0.89</td>
<td>0.91</td>
<td>1.14</td>
<td>0.97</td>
</tr>
<tr>
<td>Final disp. error</td>
<td>HOTEL [49]</td>
<td>0.77</td>
<td>0.64</td>
<td>0.66</td>
<td>0.72</td>
<td>0.95</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>ZARA 1 [39]</td>
<td>0.59</td>
<td>0.37</td>
<td>0.39</td>
<td>0.72</td>
<td>0.95</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>ZARA 2 [39]</td>
<td>0.59</td>
<td>0.37</td>
<td>0.39</td>
<td>0.72</td>
<td>0.95</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>UCY [39]</td>
<td>0.43</td>
<td>0.37</td>
<td>0.39</td>
<td>0.41</td>
<td>0.72</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>1.31</td>
<td>0.37</td>
<td>0.39</td>
<td>0.41</td>
<td>0.72</td>
<td>0.74</td>
</tr>
</tbody>
</table>

Table 1. Quantitative results of all the methods on all the datasets. We present the performance metrics as follows: First 6 rows are the Average displacement error, row 7 to 12 are the Average displacement error for non-linear regions, and the final 6 rows are the Final displacement error. All methods forecast trajectories for a fixed period of 4.8 seconds. (*) Note that IGP uses the intended ground truth destination of a person during test time unlike other methods.
At time-step 3 and 4, we notice that the Social-LSTM predicts a “halt” for person(3) in order to yield for person(1). Interestingly at time-step 4, the location of the haling point is updated to match the true turning-point in the path. At the next time-step, with more observations, the model is able to correctly predict the full turn anchored at that point.

In Figure 5, we illustrate the prediction results of our Social-LSTM, the SF model [49] and the linear baseline on one of the ETH datasets. When people walk in a group or as e.g. a couple, our model is able to jointly predict their trajectories. It is interesting to note that unlike Social Forces [73] we do not explicitly model group behavior. However, our model is better at predicting grouped trajectories in a holistic fashion. In the last row of Figure 5, we show some failure cases, i.e., when our predictions are worse than previous works. We either predict a a linear path (2nd column) or de-
Figure 5. Illustration of our Social-LSTM method predicting trajectories. On the first 3 rows, we show examples where our model successfully predicts the trajectories with small errors (in terms of position and speed). We also show other methods such as Social Forces [73] and linear method. The last row represents failure cases, e.g., person slowed down or took a linear path. Nevertheless, our Social-LSTM method predicts a plausible path. The results are shown on ETH dataset [49].

We have presented a LSTM-based model that can jointly reason across multiple individuals to predict human trajectories in a scene. We use one LSTM for each trajectory and share the information between the LSTMs through the introduction of a new Social pooling layer. We refer to the resulting model as the “Social” LSTM. Our proposed method outperforms state-of-the-art methods on two publicly available datasets. In addition, we qualitatively show that our Social-LSTM successfully predicts various non-linear behaviors arising from social interactions, such as a group of individuals moving together. Future work will extend our model to multi-class settings where several objects such as bicycles, skateboards, carts, and pedestrians share the same space. Each object will have its own label in the occupancy map. In addition, human-space interaction can be modeled in our framework by including the local static-scene image as an additional input to the LSTM. This could allow jointly modeling of human-human and human-space interactions in the same framework.

The research reported in this publication was supported by funding from the Stanford AI Lab-Toyota Center for Artificial Intelligence Research and the ONR sparse grant (N00014-13-1-0761 and N00014-15-1-2615).
References


[48] H. S. Park and J. Shi. Social saliency prediction. 2


