Laying the Foundations of Deep Long-Term Crowd Flow Prediction

Samuel S. Sohn[1], Honglu Zhou[1], Seonghyeon Moon[1], Sejong Yoon[2], Vladimir Pavlovic[1], and Mubbasir Kapadia[1]

[1] Rutgers University, Piscataway, USA
{sss286,hz289,sm2062,vladimir,mk1353}@cs.rutgers.edu
[2] The College of New Jersey, Ewing, USA
yoons@tcnj.edu

Abstract. Predicting the crowd behavior in complex environments is a key requirement for crowd and disaster management, architectural design, and urban planning. Given a crowd's immediate state, current approaches must be successively repeated over multiple time-steps for long-term predictions, leading to compute expensive and error-prone results. However, most applications require the ability to accurately predict hundreds of possible simulation outcomes (e.g., under different environment and crowd situations) at real-time rates, for which these approaches are prohibitively expensive. We propose the first deep framework to instantly predict the *long-term flow* of crowds in arbitrarily large, realistic environments. Central to our approach are a novel representation `CAGE` , which efficiently encodes crowd scenarios into compact, fixed-size representations that losslessly represent the environment, and a modified SegNet architecture for instant long-term crowd flow prediction. We conduct comprehensive experiments on novel synthetic and real datasets. Our results indicate that our approach is able to capture the essence of real crowd movement over very long time periods, while generalizing to never-before-seen environments and crowd contexts.[3]

Keywords: Vision Applications and Systems · Datasets and Evaluation

1 Introduction

Predicting the behavior of human crowds in complex environments is a key requirement in a multitude of application areas, including crowd and disaster management, architectural design, and urban planning [38]. In recent years, crowds research has been receiving increasing attention from the computer vision community [33,53,6,13,3,46,30,40,55,49]. These works have largely focused on the estimation of a crowd's size, while some others [43,2,1,56,14] have tackled the prediction of "crowd flow" at a fixed future time-step. Crowd flow is the cumulative distribution of crowd dynamics over an environment for a certain time

[3] The associated Supplementary Material, models, and datasets are available at github.com/SSSohn/LTCF.

interval. This notion has been well-studied because it is critical for understanding and managing the movement of large crowds in confined spaces [8]. In contrast to these tasks, we formulate the novel task of *long-term crowd flow prediction*. In this context, the term "long-term" indicates that crowd flow is predicted over the full simulation duration (i.e., until all agents have reached their goals), instead of over a small, fixed duration. The long-term task requires repetitions of the comparatively short-term task, which will naturally lead to error propagation over long prediction durations and become prohibitively expensive for real-time applications. Some current approaches are unable to predict long-term crowd flow because only one snapshot of the initial crowd configuration is given [1,2]. Others present a tradeoff between computational efficiency, prediction accuracy, and scenario complexity for predictions on arbitrary large, complex environments with high-density crowds (in terms of the environment and crowd size) [43,56], each of which is critical for real-time decision-making in real-world applications, such as evacuation planning. In order to escape this continuum, we tackle this new problem as a whole (not as a repeating subproblem like others have) and present the first accurate and instantaneous prediction of long-term crowd movement characteristics in complex environmental and crowd scenarios [17].

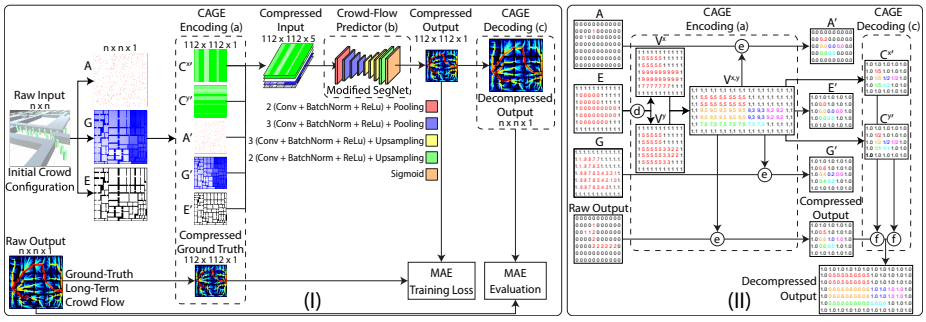

Fig. 1: Proposed Approach. (I) Deep Long-term Crowd Flow Prediction Framework. (II) Process for `CAGE` encoding and decoding.

We propose a unified deep framework (Fig. 1) to perform long-term crowd flow prediction in realistic scenarios where environments can be arbitrarily large and complex, and crowd densities can be both low and high. This is made possible through the following main contributions: (1) We propose a novel `CAGE` representation consisting of Capacity, Agent, Goal, and Environment-oriented information, which can compress environments of arbitrary size and high complexity, into a fixed-size image-like representation while preserving the key environmental characteristics, which are critical for predicting the crowd movement. (2) In order to overcome the challenge of obtaining real-world data, we develop a synthetic dataset of procedurally generated environments, dynamically simulated crowd flows, and statically derived "proxy" crowd flows (which have more error but are more efficient to compute), for model training and evaluation. (3)

While the framework is model-agnostic, we have specifically chosen the efficient SegNet architecture [4], modified to accept as input the `CAGE` representation (a 3D tensor) and predict the crowd flow. We demonstrate the efficacy of `CAGE` in encoding environments of any size and the evaluate prediction accuracy across environmental conditions, amounts of compression, crowd densities (from sparse to extremely dense crowds), and contexts (from crowds moving toward a single goal to multiple goals) using models trained on proxy and simulated crowd flow data. We showcase the model's ability to capture the essence of real human crowd behavior while permitting extrapolation to new and unseen situations. Our research lays a strong foundation towards the application of deep long-term crowd-flow prediction in applications including disaster and security management, computer-aided architectural design, and urban planning.

2 Related Work

Crowds research has been gaining attention from within the computer vision community in tackling problems such as pedestrian detection [49,3,53], crowd counting [33,6,30,40,55], crowd density estimation [13], and crowd flow prediction. Crowd flow prediction, our focus, is a spatio-temporal problem, which bears a similarity to crowd density estimation. The estimation of a crowd's density is normally performed on a camera image or video of a crowd, and the task is to determine the crowd's distribution over a space [52,25,54,42,21]. Meanwhile, the prediction of crowd flow attempts to determine the aggregation of crowd densities across the temporal axis, which encodes the crowd's movement [1,2,16,56,14,43,50]. The problem can take two forms: predictions in the short-term or long-term. Regardless of whether by rule-based simulators or deep-learning-based models, tackling long-term crowd flow prediction with short-term approaches relies on successive, expensive predictions of the crowd's future state over short time intervals (i.e, predicting the crowd motion at the next time step, then using the predicted crowd motion to continue predicting until termination). Computational time and resources are wasted in using short-term predictors for the cases where only the aggregated long-term analysis is necessary (e.g., identifying locations in the environment that will cause congestion). In contrast to the previous works, we predict crowd behaviors in the long term.

Convolutional Neural Networks (CNN) have been prevalently deployed in the domain of crowds [48,36,39,19,24], with applications in crowd counting [41,12,27] [23,45,44] and density estimation [22,15,51,20]. CNNs have also shown promise for predicting crowd dynamics [18,50]. Behavior-CNN [47] models pedestrian behaviors in crowded scenes with applications to walking path prediction, destination prediction, and tracking. ST-ResNet [50] and STRCNs [16] forecast the inflow and outflow of crowds in every region of a city. CSRNet [19] can understand and recognize highly congested scenes. Variants of CNNs (with recurrence) are also used to predict short-term crowd dynamics [14,56,16]. Given the complexity of the scenarios that we propose to tackle, we cannot rely on the same datasets as prior works, since these are not representative of concerted crowd

flows at different densities. This compels us to generate our own dataset. We employ a deep convolutional encoder-decoder architecture for the instant prediction of long-term crowd flow. Our task is similar to crowd density estimation in terms of the visual format of the output. However, we do not generate a density map for a static crowd configuration as aforementioned works have. Instead, we predict the aggregated change in a crowd's configuration over a long, future time interval.

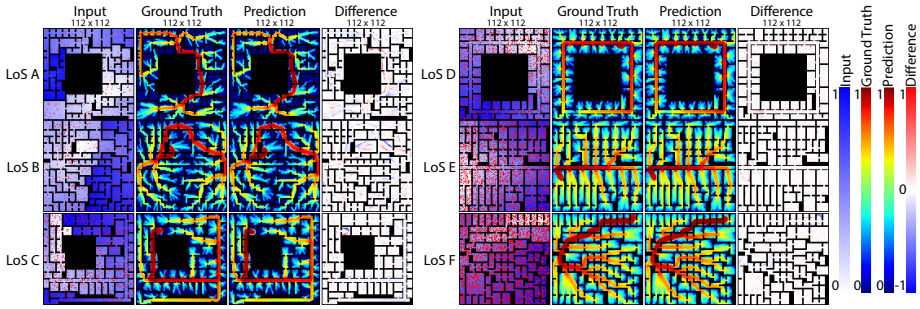

Fig. 2: Model performance on different density levels in small environments (no compression involved). See Section 4.3 for color-coding.

3 Method

We formulate the problem of *long-term crowd flow prediction* as follows:

Definition 1. *(Long-Term Crowd Flow Prediction): Given an initial crowd scenario, consisting of environment, agent, and goal information, at time t_0 and time t_l at which the last agent will reach its goal,* ***Long-Term Crowd Flow Prediction*** *aims to predict the aggregated crowd flow, i.e., the log-frequency with which agents appear in each location of the environment from t_0 to t_l.*

3.1 Preliminaries

In predicting long-term crowd flow, we do not limit ourselves with a fixed-size environment or simplistic crowd configurations. We aim specifically to make these predictions for an arbitrary crowd configuration in a built environment of any size that moves toward arbitrary goal locations. As raw input, we are given a matrix $\mathbf{E} \in \{0,1\}^{n \times n}$, which stores a 2D discretization of a built environment where each cell's width and height represent the size of one person. The diameter of the agents is 0.6×0.6 meters^2 in the simulated data, which makes the environment size between $0.6n \times 0.6n$ m^2. In matrix $\mathbf{E} \in \{0,1\}^{n \times n}$, navigable cells have value 0 and non-navigable cells have value 1. We add paddings with value 1 for environments that are not in a square shape. Another matrix $\mathbf{A} \in \{0,1\}^{n \times n}$ is used to represent the unique initial agent locations, where the cells occupied

by an agent have value 1, and all others are 0. The matrix $\mathbb{G} \in \{0,1\}^{n\times n}$ stores in each cell whether the cell contains one of the agents' goals. We use index pair (i,j) to denote a location in the matrices, where $i = 1..n$ and $j = 1..n$. The crowd flow prediction output $\mathbf{Y} \in [0,1]^{n\times n}$ is calculated for each navigable cell in environment $\mathbf{E}$ as the normalized logarithm of the number of times any agent appears in the cell (i,j). As defined earlier, long-term crowd flow aggregates the crowd states across all time steps in the interval. In this work's datasets, we have configured the time steps to be 0.02 seconds.

We further represent $\mathbb{G} \in \{0,1\}^{n\times n}$ with the matrix $\mathbf{G} \in [0,1]^{n\times n}$, which stores in each cell the normalized path distance to its closest goal location. Agents can have multiple goals, which are encoded into $\mathbf{G}$ by the following procedure. We transform each goal location $g := (i,j)$ into a distance matrix $\mathbf{D}^{(g)} \in [0,\infty)^{n\times n}$, where each cell is the non-normalized shortest path length from that cell to the closest goal using Theta* [29]. The resulting distance information $\mathbf{D}^{(g)}$ for each goal is then aggregated into $\mathbf{G}$ by taking the minimum value across the distance matrices for each cell, $\mathbf{G} = \min_g \mathbf{D}^{(g)}$, which is subsequently normalized. The resulting $\mathbf{G}$ encodes not only where goals are located (i.e., wherever $\mathbf{G}_{i,j} = 0$), but also how far an agent is from its goal. Fig. 4 visualizes multiple goals in its second row. The implication of $\mathbf{G}$ is that agents can perform pathfinding to find the shortest path to its goal, which is ubiquitous among crowd simulators.

The resulting input representation is $\mathbf{X} = [\mathbf{A}, \mathbf{G}, \mathbf{E}]$, for which the scale of the environment is dependent on the diameter of agents (which is equal to the width of a cell in $\mathbf{E}$). This means that a cell can represent an exact capacity of one agent per cell. However, there is a certain inflexibility to this representation for predicting crowd flow $\mathbf{Y}$ from $\mathbf{X}$. With a fixed number n, $\mathbf{E}$ cannot capture the size of an arbitrary input environment. We resolve this problem by converting the variable-size agent-centric representation $\mathbf{X}$ into a fixed-size environment-centric one $\mathbf{X}'$ through our novel `CAGE` encoding and decoding techniques. The proposed framework bridges these two representations (Fig. 1). The pipeline consists of three major stages: (a) the `CAGE`-encoding of raw data, (b) the prediction of crowd flow, and (c) the `CAGE`-decoding of the prediction. The predictor takes as input the `CAGE` representation $\mathbf{X}'$, which is computed by compressing the original input $\mathbf{X}$ to a fixed-size representation compatible with the predictive model's input (Sec. 3.3). If compression has occurred, then the decompression stage is needed in order to view the predicted crowd flow $\hat{\mathbf{Y}}'$ on the original environment $\hat{\mathbf{Y}}$. Next, we describe `CAGE` and propose a variant to the CNN architecture for instant long-term crowd flow prediction.

3.2 `CAGE` Representation

Fig. 1 (II) illustrates the `CAGE` encoding and decoding process, described below.

Encoding. Given an axis-aligned real environment in its discretized form $\mathbf{E}^{n\times n}$, we apply a compression method that preserves environmental information and

reduces the dimensions of $\mathbf{E}$ to $n' \times n'$, where $n' <= n$[4]. This is achieved by maintaining the local navigability between cells, while warping their capacities (i.e., the maximum number of agents that can occupy them). For this, we require two additional "channels" of information $\mathbf{C^x} \in [0,1]^{n' \times n'}$ and $\mathbf{C^y} \in [0,1]^{n' \times n'}$ for storing capacities: one for capacity along the x-axis and the other for capacity along the y-axis. Before compression, both $\mathbf{C^x}$ and $\mathbf{C^y}$ consists of ones, meaning that each cell has a capacity of 1 agent per cell. $\mathbf{C^x}$ and $\mathbf{C^y}$ for compressed environments are calculated as follows. First, based on $\mathbf{E}$, we introduce another two $n \times n$ matrices, $\mathbf{V^x}$ and $\mathbf{V^y}$, which encode the visibility of $\mathbf{E}$, i.e., in each navigable cell (i, j), $\mathbf{V^x}_{i,j}$ stores the number of visible cells along the x-axis (the number of successive navigable cells without being blocked by any non-navigable cells along the x-axis). $\mathbf{V^y}$ is computed in a similar way but along the y-axis (Fig. 1 II.d). Combining $\mathbf{V^x}$ and $\mathbf{V^y}$, we form matrix $\mathbf{V^{x,y}}$ of dimension $n \times n$, in which cell (i, j) stores values from $\mathbf{V^x}$ and $\mathbf{V^y}$ in the form of $(\mathbf{V^x}_{i,j}, \mathbf{V^y}_{i,j})$. The $p \times q$ successive cells (forming a region of size $p \times q$) in $\mathbf{V^{x,y}}$ that share the same value are the cells that can be merged into a single cell in the compressed environment. A proof of the above procedure's optimality has been provided in the Supplementary Material. As the capacity of that single cell (newly compressed) along the x-axis corresponds to q agents per cell and along the y-axis corresponds to p agents per cell, the value of that single cell becomes $1/q$ in $\mathbf{C^x}$ and $1/p$ in $\mathbf{C^y}$. The total capacity of that single cell (newly compressed) is $p \times q$. By manipulating $\mathbf{C^x}$ and $\mathbf{C^y}$, regions in $\mathbf{E}$ are able to be compressed. After the above procedure, navigable regions in the original environment are compressed and become smaller, but the compressed environment remains the same size as the original environment with dimensions $n \times n$. However, there is an increase in the amount of non-navigable cells in the borders. The outer rows and columns that are entirely non-navigable can be removed, thus decreasing the dimensions of the compressed environment by $(n - n')$ along both the x- and y-axes. In this way, $\mathbf{E}$ is now transformed into $\mathbf{E'}$. For our framework, we have used $n' = 112$ and an agent radius of 0.3, which allows us to encode environments from size 67×67 m^2 to at least 336×336 m^2 in our dataset, but this size can exceed beyond $17,136 \times 17,136$ m^2. By explicitly encoding the amount of compression, our predictive model does not need to infer the scale of the input. Other works such as Switch-CNN [35] rely on giving different scales to different regressors.

The `CAGE` encoding of $\mathbf{E}$ affects channels $\mathbf{A}$ and $\mathbf{G}$ as well. For the compressed environment, $\mathbf{A'} \in [0,1]^{n \times n}$ represents agent density, instead of binary agent presence. The compressed cell's value in $\mathbf{A'}$ is the total number of agents among the corresponding original cells in the uncompressed environment divided by the total capacity of that compressed cell. For the $\mathbf{G'} \in [0,1]^{n \times n}$ of a compressed environment, each of its cells takes on the minimum $\mathbf{G}$-value of the original cells that are being compressed into this single cell, as this is consistent with the aforementioned use of $\mathbf{G}$, i.e., in finding the shortest path. Although some in-

[4] If $n < n'$, $\mathbf{A}$, $\mathbf{E}$, and $\mathbf{G}$ are padded with row and columns to become $n' \times n'$. In this case, $\mathbf{A'}$, $\mathbf{E'}$, and $\mathbf{G'}$ are equal to $\mathbf{A}$, $\mathbf{E}$, and $\mathbf{G}$. $\mathbf{C^x}$ and $\mathbf{C^y}$ are matrix of dimension $n' \times n'$ with all entry values as 1.

formation from $\mathbf{A}$ and $\mathbf{G}$ is lost during compression, erroneous compression of $\mathbf{E}$ can more adversely affect the crowd flow prediction (e.g., a thin wall disappearing), so we primarily seek to encode $\mathbf{E}$ losslessly, while maintaining a fixed bound of the imprecision in $\mathbf{A}'$ and $\mathbf{G}'$, limited to the decompressed region. The resulting CAGE representation constitutes of 5 channels, $\mathbf{X}' = \left[\mathbf{C}^{\mathbf{x}}, \mathbf{C}^{\mathbf{y}}, \mathbf{A}', \mathbf{G}', \mathbf{E}'\right]$, where the prime symbol indicates that a matrix has been compressed to $n' \times n'$.

When training our predictive model, CAGE is also used to encode the ground truth crowd flow $\mathbf{Y}$ into compressed crowd flow $\mathbf{Y}'$ (Fig. 1 II.e), for the cases of model input being compressed. As a region of cells in $\mathbf{Y}$ is compressed into a single cell in $\mathbf{Y}'$, we sum up the crowd flow values in the original cells and divide by the total capacity of the compressed cell. This CAGE encoding technique is now equipped to represent a nearly infinite number of compressed environments and crowd scenarios within $n' \times n'$. Other image resampling techniques cannot do the same without losing some environment information. We have evaluated CAGE with other techniques such as nearest-neighbor, area, bilinear, bicubic, and Lanczos resampling for environments from our dataset, and we find that all other techniques consistently produce artifacts such as blurring [31], while CAGE is pixel-perfect (provided in the Supplementary Material). While CAGE is not guaranteed to compress a *random* environment, typical real-world environments are *structured* and, therefore, more likely to be compressed to $n' \times n'$. Furthermore, we provide a variant of the compression procedure for non-axis-aligned environments (Supplementary Material Fig. 3) that yields the same representation with some loss of environmental information.

Decoding. After the crowd flow is predicted on the compressed representation $\mathbf{X}'$ (denoted by $\hat{\mathbf{Y}}'$), decompression can be achieved by using $\mathbf{C}^{\mathbf{x}}$ and $\mathbf{C}^{\mathbf{y}}$ (Fig. 1 II.f). The value of a compressed cell in $\hat{\mathbf{Y}}'$ is duplicated back to the original cells that formed this compressed cell. In this way, we can turn $\hat{\mathbf{Y}}'$ with dimension $n' \times n'$ into $\hat{\mathbf{Y}}$ of dimension $n \times n$. CAGE considers each cell value in $\hat{\mathbf{Y}}'$ as the *average* crowd flow in a cell, and when CAGE decodes $\hat{\mathbf{Y}}'$, it recovers the original amount of crowd flow that was in the same compressed region. If prediction is perfectly accurate, the sum of values in $\hat{\mathbf{Y}}$ is equal to the sum of values in $\mathbf{Y}$. This can produce some aliasing artifacts, but since crowd flow is smooth, it is less affected by this.

3.3 Crowd-Flow Prediction

CAGE-encoded representations are effectively 5-channel images, which motivates our use of Convolutional Neural Networks. For the predictor (b) in Fig. 1, we modified SegNet [4] to facilitate the deep prediction of the complex crowd flows. We favor ourselves in modifying SegNet out of the following reasons: (1) SegNet is a symmetrical architecture with the ability to encode image-like representations to low dimensions, and map the low-resolution representation back to its full input resolution, perfectly serving our purposes. The mapping also has superiority in delineating objects of small sizes and in retaining boundary information for structured prediction without having as many trainable parameters as models such as U-Net [34]. (2) SegNet has the capability to extract, preserve, and

understand appearance, shape, and the spatial-relationship information from the input image representation, which is beneficial for us to differentiate agents, obstacles, and navigable areas, and to capture the spatial-temporal correlation behind the scene. (3) SegNet is smaller and easier to train in an end-to-end fashion than many other CNN architectures such as [28,26]. It is designed to be efficient in memory and computational time especially in inference, which is crucial for our application scenarios (e.g. predicting long-term crowd flow to aid timely plan-making for emergency evacuations).

Network Architecture. The modified SegNet still has a symmetrical convolutional encoder-decoder architecture like the original SegNet. We made the following 2 changes. First, we removed the 3 outermost layers from both the encoder and decoder to allow inputs and outputs with dimensions of 112×112, instead of the original 224×224. This is because 224×224 is unnecessarily large for our dataset. The architecture of the modified SegNet can be seen in Fig. 1 (I), in which all convolutional layers utilize 3×3 convolution kernels and 1 padding. Second, the original SegNet is designed for image segmentation. Therefore, its last layer is a *softmax* for pixel-wise labeling. The prediction of long-term crowd flow is a regression task, so we have substituted the last layer with a *sigmoid* layer to predict the compressed output.

Training Details and Loss Function. For the modified SegNet, the encoder and decoder weights are all initialized following [10]. Stochastic gradient descent (SGD) with a fixed learning rate of 0.01 and momentum of 0.9 is used as the optimizer [5]. The batch size is set to 64, and the training set is shuffled before each epoch. We train the model with 200 epochs and use the same set of hyper-parameters for each of the following model variants. We select model hyper-parameters according to the validation performance. The batch size, learning rate and the momentum value are chosen empirically to obtain the best validation result from sets $\{32, 64, 128\}$, $\{0.1, 0.01, 0.001\}$, and $\{0.8, 0.9\}$ respectively. Adaptive learning rates have been considered but we find that the fixed one remains to be the most effective. We use *mean absolute error* as the objective loss function. The size of the training set (10% for validation) and the size of the testing set vary for each model variant, which we will describe in detail in Section 5.

4 Experimental Preliminaries

4.1 Synthetic Datasets

We have compiled a large dataset of synthetic environments and crowd movement for model training and evaluation.

Synthetic Environments. The work in [7] establishes a comprehensive taxonomy of floorplans based on observations of the exterior shapes and interior organizations of modern architecture. We have re-purposed this taxonomy in order to generate floorplans instead of classifying them. For this study, a dataset of over 84,000 floorplans was procedurally generated. We describe the particulars of our generator in the Supplementary Material.

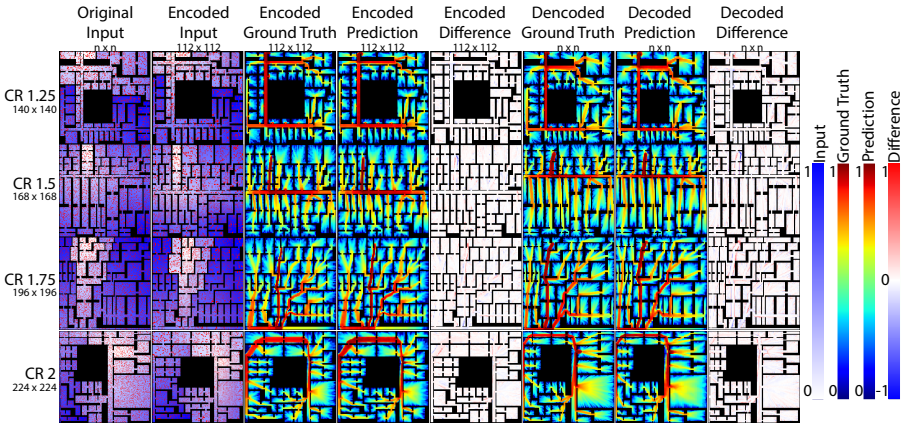

Fig. 3: Model performance on different compression rates (CR).

Simulated Crowd Flow. The crowd flow data that we use as ground truth must either be taken from the real world or simulated. It is quite difficult to acquire real-world crowd flows in varied environments and scenarios, so we have instead opted to simulate crowd flow using the Social Force model (SF) [11]. Our choice of the SF model is based on the model's computational efficiency and widespread use to this day in fields such as architecture, urban planning, and transportation engineering.

The simulated crowd flow has been generated under two crowd scenarios: scenarios with a single goal where agents are uniformly distributed across the environment, and multi-goal and non-uniform scenarios (MGNU), where agents share multiple goals and are distributed in clusters. Using an open-source simulator [37], it takes over 167 hours to produce 7,800 crowd flow images in a 112×112 environment, making it prohibitively expensive to produce larger dataset with larger environments. Therefore, it is desirable to have a proxy that has key characteristics of simulated crowd flow, but is less expensive to generate.

Proxy Crowd Flow. Using proxy crowd flow, we aim to capture two major features of SF crowd flow: that (1) agents rely on the shortest path, and (2) as agents move closer to their goal, they become clustered with other agents, which causes congestion at bottlenecks and increases crowd flow closer to goals. The average mean absolute error between proxy and SF crowd flow across 600 varied goal, environment, and agent scenarios is 0.073 ± 0.025. In the context of the visualization, each primary/secondary color occupies $60°$ of the hue spectrum, and this error translates to an expected hue error of $26.28° \pm 9°$ for each pixel, which is very small. A visual comparison between proxy and SF crowd flows can be found in the Supplementary Material.

Given raw input $\mathbf{X} = [\mathbf{A}, \mathbf{G}, \mathbf{E}]$, the proxy crowd flow is generated using Alg. 1. Each agent in $\mathbf{A}$ has its shortest path of cells planned to its goal (line 3), and has its path overlaid onto a frequency matrix $\mathbf{F}$ (line 5). The frequency matrix is then used to produce the latter feature of SF crowd flow by clustering

Algorithm 1 Proxy crowd flow generation procedure.

1: $\mathbf{F} \leftarrow [0]^{n \times n}$
2: **for** $(i, j) \in \mathbf{A} \mid \mathbf{A}_{i,j} = 1$ **do**
3: $\quad path \leftarrow ShortestPathToClosestGoal(i, j)$
4: $\quad$ **for** $(a, b) \in path$ **do**
5: $\quad\quad \mathbf{F}_{a,b} \leftarrow \mathbf{F}_{a,b} + 1$
6: $\mathbf{Y} \leftarrow [0]^{n \times n}$
7: **for** $(i, j) \in \mathbf{F}$ **do**
8: $\quad groupSize \leftarrow \log(1 + \mathbf{F}_{i,j})$
9: $\quad group \leftarrow GetClosestCells(i, j, groupSize)$
10: $\quad$ **for** $(a, b) \in group$ **do**
11: $\quad\quad \mathbf{Y}_{a,b} \leftarrow \mathbf{Y}_{a,b} + \mathbf{F}_{i,j}$

agents (line 9) and increasing crowd flow closer to goals (line 11). Function *GetClosestCells* returns a set of *groupSize*-many cells with closest path distance to cell (i, j). The proxy crowd flow is generated under the crowd scenario where there is a single goal in the environment and agents are uniformly distributed. It takes little over 2 hours to generate the same 7,800 scenarios as the SF crowd flow (167 hours).

4.2 Real-World Datasets

NARF Dataset. We evaluate our model on real-world environments with simulated crowd flows. These environments include King's Cross, Prentice Women's Hospital, University of Economics Vienna's Library and Learning Centre, and Tate Britain. The environments are non-axis-aligned real-world floorplans (NARF), and serve to evaluate the ability for our models, which were trained only on axis-aligned environments, to generalize.

Stanford Crowd Flow. The Stanford crowd trajectory dataset [1] provides a large set of real pedestrian trajectories collected at a train station of size $25m \times 100m$ for 12×2 hours (totally 12 days, for each day, 7:00 a.m. to 8:00 a.m. and 5:00 p.m. to 6:00 p.m. are recorded) by a set of distributed cameras. We follow the same data processing strategy introduced in [32] to retrieve the pedestrians' trajectories and the environment layout. We select the data of the first day and choose 5 minutes as the time interval to bin the two-hour data by.

4.3 Evaluation Protocols

Qualitative Evaluation. We use a visualization scheme for comparing two types of crowd flows (typically ground truth and predictions). For example, in Fig. 3, the first two columns show the `CAGE` input, where the white-blue gradient corresponds to $\mathbf{G}$, the transparent-to-red gradient corresponds to $\mathbf{A}$, and the black pixels show where $\mathbf{E}$ is non-navigable. The next two columns show crowd flow using the jet gradient, and the following column computes the signed difference by subtracting the third column from the fourth column,

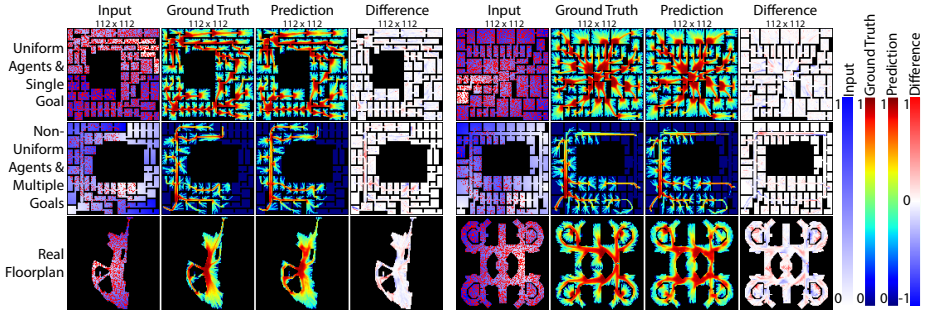

Fig. 4: Model performance on SF crowd flows for synthetic environment (first two rows) and real-world floorplan (last row).

	$PanL^{PX}$			$PanL^{SF}$					$PanL^{SF+}$			
CR-Train	1	1	1	1	1	1	1	1	1	1	1	1
Flow-Train	Proxy	Proxy	Proxy	SF	SF	SF	SF	SF	SF + MGNU	SF + MGNU	SF + MGNU	SF + MGNU
Flow-Test	Proxy	SF	Stanford	Proxy	SF	MGNU	NARF	Stanford	SF	MGNU	NARF	Stanford
LoS A	**0.0186**	0.0707	0.0160	0.0665	0.0250	0.0336	0.0672	**0.0141**	**0.0200**	**0.0084**	**0.0514**	0.0142
LoS B	**0.0179**	0.0777	0.0151	0.0720	0.0261	0.0341	0.0649	0.0149	**0.0216**	**0.0088**	**0.0537**	**0.0144**
LoS C	**0.0167**	0.0752	0.0162	0.0700	0.0256	0.0346	0.0635	0.0152	**0.0215**	**0.0099**	**0.0544**	**0.0145**
LoS D	**0.0161**	0.0738	0.0172	0.0700	0.0254	0.0345	0.0607	0.0158	**0.0222**	**0.0095**	**0.0546**	**0.0148**
LoS E	**0.0147**	0.0732	N/A	0.0711	0.0250	0.0326	0.0594	N/A	**0.0221**	**0.0121**	**0.0546**	N/A
LoS F	**0.0143**	0.0722	N/A	0.0732	0.0241	0.0346	0.0570	N/A	**0.0220**	**0.0095**	**0.0542**	N/A

Table 1: Average Mean Absolute Error of models trained without compression.

where negative values are false negatives (FN) that underpredict crowd flow, and positive values are false positives (FP) that produce phantom crowd flow where there should be either less or none. The signed difference is visualized using a blue-white-red gradient, where FN is blue, perfect accuracy is white, and FP is red.

Quantitative Evaluation. We use the widely-adopted Mean Absolute Error (MAE) and Mean Squared Error (MSE) for quantitative evaluation [51,24,44,27] [15,42,45,23]. For each test scenario, we calculate the MAE and MSE of each test data point and then average over all test data points in that test set. We choose MAE to report instead of MSE because MSE is often being criticized for throwing the scale off and penalizing more for outliers, it is also biased towards higher deviations. In general MAE is more robust compared to MSE.

5 Experiments and Evaluation

5.1 Varied Density and No Compression

Level of Service (LoS) [9] describes flow relationships with respect to volume or density in a pedestrian environment. Six Levels of Service are defined (A to F) from sparse to dense crowds. We use the following LoS density conditions for evaluating our models: A $\leq$ 0.27 agents/m^2, 0.31 $< B \leq$ 0.43, C $\leq$ 0.72, D $\leq$ 1.08, E $\leq$ 2.17, F $>$ 2.17. We have trained a model $PanL^{PX}$ using proxy crowd flow data from LoS A to F (train-test split of $12k$: $3k$ with equal parts

per LoS). The qualitative and quantitative results are presented in Fig. 2 and Tab. 1 respectively. Model $PanL^{PX}$ performs well across all density levels.

5.2 Varied Compression Rate with Varied Density

Given an environment $\mathbf{E}^{n\times n}$ that has been CAGE-encoded into $\mathbf{E}^{n'\times n'}$, Compression Rate (CR) is defined as n/n'. We evaluate CAGE and the effect of compression rate on proxy crowd flow prediction for crowds of varied density levels (LoS A to F). We first train independent models on data with each of the following compression rates $\{1.0, 1.25, 1.5, 1.75, 2.0\}$. The models are $PanL^{PX}$, $CR^{1.25}$, $CR^{1.5}$, $CR^{1.75}$, and CR^{2} respectively (train-test split is $12k$: $3k$ with equal parts per LoS). We test the 5 models on all compression rates. The results shown in Tab. 2 indicate that models trained with samples from CR 1.25 and CR 1.75 perform better across compression rates, alluding to the benefits of a model trained across different compression rates.

	$PanL^{PX}$	$CR^{1.25}$	$CR^{1.5}$	$CR^{1.75}$	CR^{2}	$PanCR$
CR-Train	1	1.25	1.5	1.75	2	1∼2
Flow-Train	Proxy	Proxy	Proxy	Proxy	Proxy	Proxy
Flow-Test	Proxy	Proxy	Proxy	Proxy	Proxy	Proxy
LoS A	0.0728	0.0346	0.0358	0.0357	0.1270	**0.0322**
LoS B	0.0582	0.0312	0.0320	0.0326	0.1147	**0.0289**
LoS C	0.0509	0.0296	0.0300	0.0308	0.1248	**0.0269**
LoS D	0.0475	0.0274	0.0279	0.0285	0.1345	**0.0248**
LoS E	0.0402	0.0232	0.0237	0.0241	0.0997	**0.0210**
LoS F	0.0389	0.0212	0.0224	0.0211	0.0795	**0.0211**
CR 1.0	**0.0164**	0.0190	0.0201	0.0261	0.1011	0.0167
CR 1.25	0.0433	0.0247	0.0255	0.0280	0.1057	**0.0236**
CR 1.5	0.0577	0.0288	0.0293	0.0291	0.1097	**0.0273**
CR 1.75	0.0667	0.0321	0.0326	0.0301	0.1128	**0.0298**
CR 2.0	0.0728	0.0348	0.0355	**0.0313**	0.1415	0.0317

Table 2: Average Mean Absolute Error of models trained with CAGE under different compression rate (CR).

Following these observations, we train a model on a dataset of all compression rates and all density levels (train-test split is $18k$: $3k$ with equal parts per CR and LoS). The qualitative results of this multi-density and multi-compression-rate model ($PanCR$) can be found in Fig. 3. We also compare $PanCR$ with the models trained on a single compression rate quantitatively in Tab. 2. Model $PanCR$ is able to successfully predict proxy crowd flow across different environment sizes and crowd densities.

5.3 Results on Simulated Crowd Flow

Substitute Proxy with Simulated. Using model $PanL^{SF}$, we evaluate the performance of our approach on Social Force simulated crowd flows for all density levels on environments with a compression rate of 1 (train-test split is $7.2k$: $0.6k$ with equal parts per LoS). Qualitative results are shown in the first row of Fig. 4, and a quantitative comparison between $PanL^{PX}$ and $PanL^{SF}$ can be seen in Tab. 1. Our model achieves high performance on the SF crowd flow prediction task, showing results that we theorized to be more difficult to achieve than for proxy crowd flow. This finding indicates that our approach is applicable to realistic and complex simulated crowd flows.

Multi-Goal and Non-Uniform Scenario (MGNU). We further explore the model's capability for more complex crowd scenarios, where agents or people in the crowd are targeting different goals simultaneously and are initially clustered

instead of being uniformly distributed (MGNU scenarios). We test $PanL^{SF}$, which was trained with only SF flows for single goals and uniform agent distributions, on 600 cases of SF simulated MGNU scenarios. We also trained a model $PanL^{SF+}$with the same amount of total data but 50% being SF flows for single goals and uniform agents, and the other 50% being SF flows for multiples goals and clustered agents. The quantitative comparison between $PanL^{SF}$ and $PanL^{SF+}$ can be seen in Tab. 1. $PanL^{SF+}$ performs better than $PanL^{SF}$ on the more challenging MGNU scenarios, while still performing well on the single-goal and uniform distribution scenarios. The qualitative results are shown in the second row of Fig. 4.

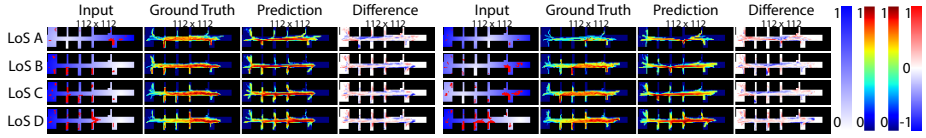

Fig. 5: Model $PanL^{SF+}$ on real-world crowd flows (Stanford Crowd Trajectory dataset).

5.4 Results on Real-World Data

Tab. 1 and Fig. 4 illustrate the prediction performance of our models on the NARF dataset. Though trained with only axis-aligned crowd configurations, our model generalizes well to non-axis-aligned unseen real environments. We test $PanL^{PX}$, $PanL^{SF}$ and $PanL^{SF+}$ on Stanford Crowd Trajectory dataset. The quantitative results can be found in Tab. 1 and the qualitative results of $PanL^{SF+}$ can be seen in Fig. 5. $PanL^{SF+}$ reproduces the crowd movement patterns exhibited in real crowds at different density levels. $PanL^{PX}$ (trained with proxy crowd flows) also achieved comparable performance, indicating that proxy crowd flows are adequate for gaining a general understanding of real crowd flows. More results on Stanford Crowd Trajectory dataset can be found in the Supplementary Material.

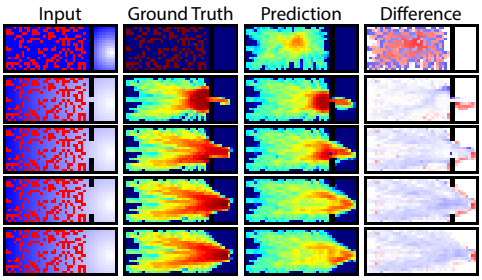

Fig. 6: Crowd flow prediction for different bottleneck widths.

5.5 Case Study

We demonstrate the potential of our prediction platform to be used as part of computer-aided design pipelines. In this simple study, we use $PanL^{SF+}$ to in-

vestigate the change in predicted crowd flow in a simple egress scenario through a bottleneck that gradually widens, where the goal is centered on the right side of the bottleneck (Fig. 6). The results are consistent with our expectation that widening the bottleneck would decrease congestion, and the result for an inaccessible goal (top) is quite surprising, since no training data has been provided for this scenario.

6 Conclusion

We have formulated the novel task of long-term crowd flow prediction and proposed the first deep framework for instantly and accurately predicting long-term crowd movement patterns (crowd flow) in scenarios involving complex environments of any size and crowds varying in both density and configuration. These include sparse crowds which require the prediction of microscopic features of individuals, extremely dense crowds which are prohibitively challenging for current methods, and heterogeneous crowds with different destinations. Central to our approach are a novel `CAGE` representation which facilitates the efficient, lossless encoding of environments, and a modified SegNet architecture for crowd flow prediction. We have developed a novel dataset of synthetic environments and two forms of crowd flows (dynamically simulated crowd flows and statically derived proxy crowd flows) for model training and evaluation, offsetting the challenges of obtaining sufficient data from real human crowds. This dataset acts as a gateway into a new and exciting crowds research area within computer vision and establishes the baseline for others to compare their advancements against. Our models are able to accurately and efficiently predict proxy and simulated crowd flow across a large variety of environments and crowd contexts, while also generalizing to new situations (i.e., real crowd flows and real environments). Two valuable insights from our experiments are that (1) training on proxy crowd flows yields similar predictive accuracy to training on simulated crowd flows for a real-world dataset and (2) training our model on axis-aligned, rectangular environments provides a surprisingly good transfer to testing on never-before-seen, non-axis-aligned environments (Tab. 1).

Limitations and Future Work. `CAGE` can handle arbitrarily large axis-aligned environments, but makes no guarantee to successfully compress any axis-aligned environment into a pre-defined size. Although it facilitates the accurate prediction of potential congestion areas in an environment and is thus adequate for applications (e.g. crowd and disaster management), `CAGE`-decoding may cause aliasing artifacts on the crowd flow.

Acknowledgements

The project was funded in part by NSF IIS-1703883 and NSF S&AS-1723869.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 961–971 (2016) 1, 2, 3, 10
2. Amirian, J., Hayet, J.B., Pettre, J.: Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (June 2019) 1, 2, 3
3. Assari, S.M., Idrees, H., Shah, M.: Human re-identification in crowd videos using personal, social and environmental constraints. In: European Conference on Computer Vision. pp. 119–136. Springer (2016) 1, 3
4. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. CoRR **abs/1511.00561** (2015), http://arxiv.org/abs/1511.00561 3, 7
5. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp. 177–186. Springer (2010) 8
6. Cao, X., Wang, Z., Zhao, Y., Su, F.: Scale aggregation network for accurate and efficient crowd counting. In: The European Conference on Computer Vision (ECCV) (September 2018) 1, 3
7. Dogan, T., Saratsis, E., Reinhart, C.: The optimization potential of floor-plan typologies in early design energy modeling (12 2015) 8
8. Fang, Z., Lo, S., Lu, J.: On the relationship between crowd density and movement velocity. Fire Safety Journal **38**(3), 271–283 (2003) 2
9. Fruin, J.J.: Pedestrian planning and design. Tech. rep. (1971) 11
10. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE international conference on computer vision. pp. 1026–1034 (2015) 8
11. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Phys. Rev. E **51**, 4282–4286 (May 1995). https://doi.org/10.1103/PhysRevE.51.4282, https://link.aps.org/doi/10.1103/PhysRevE.51.4282 9
12. Huang, S., Li, X., Zhang, Z., Wu, F., Gao, S., Ji, R., Han, J.: Body structure aware deep crowd counting. IEEE Transactions on Image Processing **27**(3), 1049–1059 (2017) 3
13. Idrees, H., Tayyab, M., Athrey, K., Zhang, D., Al-Maadeed, S., Rajpoot, N., Shah, M.: Composition loss for counting, density map estimation and localization in dense crowds. In: The European Conference on Computer Vision (ECCV) (September 2018) 1, 3
14. Jiang, R., Song, X., Huang, D., Song, X., Xia, T., Cai, Z., Wang, Z., Kim, K.S., Shibasaki, R.: Deepurbanevent: A system for predicting citywide crowd dynamics at big events. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2114–2122. ACM (2019) 1, 3
15. Jiang, X., Xiao, Z., Zhang, B., Zhen, X., Cao, X., Doermann, D., Shao, L.: Crowd counting and density estimation by trellis encoder-decoder networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6133–6142 (2019) 3, 11
16. Jin, W., Lin, Y., Wu, Z., Wan, H.: Spatio-temporal recurrent convolutional networks for citywide short-term crowd flows prediction. In: Proceedings of the 2nd International Conference on Compute and Data Analysis. pp. 28–35. ACM (2018) 3

17. Kapadia, M., Pelechano, N., Allbeck, J., Badler, N.: Virtual crowds: Steps toward behavioral realism. Synthesis lectures on visual computing: computer graphics, animation, computational photography, and imaging **7**(4), 1–270 (2015) 2
18. Li, C., Zhang, Z., Sun Lee, W., Hee Lee, G.: Convolutional sequence to sequence model for human dynamics. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5226–5234 (2018) 3
19. Li, Y., Zhang, X., Chen, D.: Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1091–1100 (2018) 3
20. Lian, D., Li, J., Zheng, J., Luo, W., Gao, S.: Density map regression guided detection network for rgb-d crowd counting and localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1821–1830 (2019) 3
21. Liu, C., Weng, X., Mu, Y.: Recurrent attentive zooming for joint crowd counting and precise localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1217–1226 (2019) 3
22. Liu, J., Gao, C., Meng, D., Hauptmann, A.G.: Decidenet: Counting varying density crowds through attention guided detection and density estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5197–5206 (2018) 3
23. Liu, L., Qiu, Z., Li, G., Liu, S., Ouyang, W., Lin, L.: Crowd counting with deep structured scale integration network. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1774–1783 (2019) 3, 11
24. Liu, N., Long, Y., Zou, C., Niu, Q., Pan, L., Wu, H.: Adcrowdnet: An attention-injective deformable convolutional network for crowd understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3225–3234 (2019) 3, 11
25. Liu, S., Huang, D., Wang, Y.: Adaptive nms: Refining pedestrian detection in a crowd. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6459–6468 (2019) 3
26. Liu, W., Rabinovich, A., Berg, A.C.: Parsenet: Looking wider to see better. arXiv preprint arXiv:1506.04579 (2015) 8
27. Liu, W., Salzmann, M., Fua, P.: Context-aware crowd counting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5099–5108 (2019) 3, 11
28. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431–3440 (2015) 8
29. Nash, A., Daniel, K., Koenig, S., Felner, A.: Theta^*: Any-angle path planning on grids. In: AAAI. vol. 7, pp. 1177–1183 (2007) 5
30. Oñoro-Rubio, D., López-Sastre, R.J.: Towards perspective-free object counting with deep learning. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 615–629. Springer International Publishing, Cham (2016) 1, 3
31. Parker, J.A., Kenyon, R.V., Troxel, D.E.: Comparison of interpolating methods for image resampling. IEEE Transactions on medical imaging **2**(1), 31–39 (1983) 7
32. Qiao, G., Zhou, H., Kapadia, M., Yoon, S., Pavlovic, V.: Scenario generalization of data-driven imitation models in crowd simulation. In: Motion, Interaction and Games. p. 36. ACM (2019) 10
33. Ranjan, V., Le, H., Hoai, M.: Iterative crowd counting. In: The European Conference on Computer Vision (ECCV) (September 2018) 1, 3

34. 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) 7
35. Sam, D.B., Surya, S., Babu, R.V.: Switching convolutional neural network for crowd counting. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4031–4039. IEEE (2017) 6
36. Sindagi, V., Patel, V.M.: A survey of recent advances in cnn-based single image crowd counting and density estimation. CoRR **abs/1707.01202** (2017), http://arxiv.org/abs/1707.01202 3
37. Singh, S., Kapadia, M., Faloutsos, P., Reinman, G.: An open framework for developing, evaluating, and sharing steering algorithms. In: International Workshop on Motion in Games. pp. 158–169. Springer (2009) 9
38. Thalmann, D., Musse, S.R.: Crowd Simulation, Second Edition. Springer (2013) 1
39. Tripathi, G., Singh, K., Vishwakarma, D.K.: Convolutional neural networks for crowd behaviour analysis: a survey. The Visual Computer **35**(5), 753–776 (2019) 3
40. Walach, E., Wolf, L.: Learning to count with cnn boosting. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 660–676. Springer International Publishing, Cham (2016) 1, 3
41. Wan, J., Luo, W., Wu, B., Chan, A.B., Liu, W.: Residual regression with semantic prior for crowd counting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4036–4045 (2019) 3
42. Wang, Q., Gao, J., Lin, W., Yuan, Y.: Learning from synthetic data for crowd counting in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8198–8207 (2019) 3, 11
43. Wolinski, D., Lin, M.C., Pettré, J.: Warpdriver: Context-aware probabilistic motion prediction for crowd simulation. ACM Trans. Graph. **35**(6), 164:1–164:11 (Nov 2016). https://doi.org/10.1145/2980179.2982442, http://doi.acm.org/10.1145/2980179.2982442 1, 2, 3
44. Xu, C., Qiu, K., Fu, J., Bai, S., Xu, Y., Bai, X.: Learn to scale: Generating multipolar normalized density maps for crowd counting. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 8382–8390 (2019) 3, 11
45. Yan, Z., Yuan, Y., Zuo, W., Tan, X., Wang, Y., Wen, S., Ding, E.: Perspective-guided convolution networks for crowd counting. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 952–961 (2019) 3, 11
46. Yi, S., Li, H., Wang, X.: Pedestrian behavior understanding and prediction with deep neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 263–279. Springer International Publishing, Cham (2016) 1
47. Yi, S., Li, H., Wang, X.: Pedestrian behavior understanding and prediction with deep neural networks. In: European Conference on Computer Vision. pp. 263–279. Springer (2016) 3
48. Yogameena, B., Nagananthini, C.: Computer vision based crowd disaster avoidance system: A survey. International journal of disaster risk reduction **22**, 95–129 (2017) 3
49. Yun, S., Yun, K., Choi, J., Choi, J.Y.: Density-aware pedestrian proposal networks for robust people detection in crowded scenes. In: Hua, G., Jégou, H. (eds.) Computer Vision – ECCV 2016 Workshops. pp. 643–654. Springer International Publishing, Cham (2016) 1, 3

50. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. In: Thirty-First AAAI Conference on Artificial Intelligence (2017) 3
51. Zhang, Q., Chan, A.B.: Wide-area crowd counting via ground-plane density maps and multi-view fusion cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8297–8306 (2019) 3, 11
52. Zhang, S., Wu, G., Costeira, J.P., Moura, J.M.: Understanding traffic density from large-scale web camera data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5898–5907 (2017) 3
53. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Occlusion-aware r-cnn: Detecting pedestrians in a crowd. In: The European Conference on Computer Vision (ECCV) (September 2018) 1, 3
54. Zhao, M., Zhang, J., Zhang, C., Zhang, W.: Leveraging heterogeneous auxiliary tasks to assist crowd counting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12736–12745 (2019) 3
55. Zhao, Z., Li, H., Zhao, R., Wang, X.: Crossing-line crowd counting with two-phase deep neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 712–726. Springer International Publishing, Cham (2016) 1, 3
56. Zonoozi, A., jae Kim, J., li Li, X., Cong, G.: Periodic-crn: A convolutional recurrent model for crowd density prediction with recurring periodic patterns. In: IJCAI (2018) 1, 2, 3