Computers & Graphics 88 (2020) 83–96

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Predicting Crowd Egress and Environment Relationships to Support Building Design Optimization ☆

Kaidong Hu [a], Sejong Yoon [b,*], Vladimir Pavlovic [a], Petros Faloutsos [c,d], Mubbasir Kapadia [a]

[a] Rutgers University, Piscataway, NJ, 08854, USA
[b] The College of New Jersey, Ewing, NJ, 08628, USA
[c] York University, Toronto, Ontario, M3J 1P3, Canada
[d] University Health Network-Toronto Rehabilitation Institute, Toronto, Ontario, M5G 2C4, Canada

ARTICLE INFO

Article history:
Received 15 September 2019
Revised 25 March 2020
Accepted 25 March 2020
Available online 12 April 2020

Keywords:
Crowd Simulation
Computer-Aided Design Optimization
Neural Network

ABSTRACT

Evaluating and optimizing the design of built and yet-to-be-built environments, with respect to human occupancy and behavior is both greatly beneficial and challenging. Crowd simulation can provide the computational means to analyze a design through the movement of virtual occupants (agents). A range of analytic information (metrics) can be computed from the simulated movement of the agents that offer insights on the design. Crowd simulation and the related analysis can be part of interactive or offline design optimization pipelines. Unfortunately, large scale crowd simulations are prohibitively expensive, especially when used within iterative design and optimization loops, where hundreds of simulations often need to be computed at interactive rates. We propose a machine learning framework that aims to solve this problem by learning the relationship between a building design and the evaluation metrics extracted from expensive simulations. We train an offline regression neural network using a synthetic training set that we generate for this purpose. Once the network is trained it can evaluate new designs efficiently, and approximate the corresponding analytic information with high accuracy. The proposed framework can also be used to find an optimized layout. We demonstrate the effectiveness of the framework on a variety of real world case studies.

1. Introduction

In architectural design, it is of primary importance to predict the relationship between an environment and the movement of its occupants at the time a building is designed rather than after it is built and occupied. Crowd simulations have been developed for such a purpose – to inform the decision-making process of architects and engineers so that they can test the implications of architectural design options before committing to their realization.

The application of crowd simulation for architectural design, however, is mostly limited to the analysis of a reduced number of design options generated by the architect [1]. Recent advancements in Computer-Aided Design (CAD) have facilitated the development of dynamic optimization tools that help architects explore a vast range of design solutions and find the one(s) that best satisfy different kinds of performance criteria. This is an iterative process whereby design solutions are automatically synthesized by the computer, and then progressively tested and refined to maximize a user-defined utility function. While this approach has been successfully applied to optimize floor plans [2–5], HVAC systems [6], thermal and lighting performance [7,8], acoustics [9] and building energy consumption [8,10], fewer efforts have been directed towards the optimization of architectural layouts for crowd behavior, mostly because of the dynamic nature of human movement.

Some prior works [11,12] have utilized static models of human movement for the optimization. However, these approaches rely on mathematically-inclined analysis of geometric aspects of a building's layouts, without considering crowd-oriented features such as egress times, movement speeds, and distance traveled. A different approach is thus needed, which incorporates dynamic aspects of human crowd movement, while still supporting efficient calculations and optimization.

To address this issue, we propose a machine learning framework to learn environment-crowd relationships from synthetic (simulated) training data, which is then used as the basis for crowd-aware building design optimization. Our approach involves modeling the aggregate dynamics of a virtual crowd and their relationship to the environment by training a neural-network on simulated crowd movement data. Specifically, we focus on crowd egress

☆ This article was recommended for publication by R Boulic.

* Corresponding author.
E-mail address: yoons@tcnj.edu (S. Yoon).

https://doi.org/10.1016/j.cag.2020.03.005
0097-8493/

behavior, and how it is impacted by the environment. This data is generated using agent-based crowd simulation techniques (e.g., Social Force [13] and ORCA [14]). In particular, we explore different neural network (NN) architectures to systematically study their ability to fit the training data, while generalizing to new situations.

Our experimental results demonstrate the potential utility of neural networks to improve the prediction performance over the baseline linear regression models. We also utilized this trained prediction neural network to compute loss function for the building design optimization framework. When applied for building design optimization, it shows significant improvements compared to existing methods as demonstrated in the empirical results. We also show how our optimization framework can help designers improve their design solutions as far as crowd movement is concerned, in a real environment.

Our contribution can be summarized as follows: (a) a novel neural network-based framework to learn the relationship between crowd movement and a building design in evacuation scenarios, (b) an optimization algorithm that efficiently iterates a vast number of design solutions to generate crowd-aware environmental layout design, and (c) a case study that involves optimizing a complex, real-world built environment (Metropolitan museum) to minimize evacuation times and reduce congestion.

2. Related Work

Computer-aided design (CAD) methods have garnered increasing attention in recent years, since they allow designers to efficiently optimize a building layout with respect to a wide range of design criteria, including crowd behavior considerations. In the following paragraphs, we report recent advances in the field.

2.1. Agent-based Crowd Simulation

There are three different categories of crowd simulation models: macroscopic (flow), mesoscopic (blob), and microscopic (individual) approaches. Macroscopic approaches [15,16] model crowds as a continuum in order to meet efficiency considerations, but are unable to model the underlying characteristics of each individual. While our work can certainly rely on macroscopic techniques, the focus of this study is to use agent-based (microscopic) techniques as the underlying simulator to support predictions.

Different approaches have been proposed for simulating microscopic crowd behaviors (for a comprehensive summary of current approaches please refer to [17]). Rule-based systems determine steering behaviors of agents represented as particles [18,19]. Such particle approaches have been further refined using social force models [13,20]. Geometric algorithms are used [14,21] to determine collision-free paths by accounting for the predicted velocities of neighbour agents. Agents have also used affordance fields [22] to identify a path to a goal. Cognitive-based approaches were used utility functions and an attentional system to define agents' desires and perception of the environment [20]. Different steering algorithm have been proposed to better represent agents' movements [23]. Path-planning approaches have been employed to calculate collision-free trajectories in complex environments [24,25]. Some approach can solve path planning in dynamic environments [26–28]. Parallelized approaches [29,30] have been used to accelerate the path search.

Data-driven techniques use local-space samples generated from real or simulated data to create steering policies. In [31] video samples were compiled into a database based on which the agents steer. The work of [32] focused more on recreating group dynamics than individual steering. The work of [33] used a more constrained state space of discretized slices around an agent.

There has been prior works using machine learning algorithms to understand or learn crowd motion [34], including those using data-driven techniques, e.g., [35–40], or evaluation of these approaches across different data and measures [41]. More recent works propose a new semantic metric learned from data [42] or visualize the latent manifold relating crowd simulation instances and environmental complexity [43]. However, most of these approaches are focused on human movement without much consideration on the relationship between the environment and the crowd motion. In our work we tightly couple crowd movement with the environment in which it takes place.

Our work is complementary to the large body of work in developing computational models of crowd behavior, and takes advantage of these simulation models to generate synthetic training data to learn environment-crowd relationships. Specifically, we have employed the social force (SF) [13] and the optimal reciprocal collision avoidance (ORCA) [44] to generate simulation data in this work. Other simulation techniques can easily be incorporated into our framework using the same general principles proposed here.

2.2. Building design optimization tools

There is a growing interest in using optimization techniques to explore architectural design options for near-optimal solutions with respect to a given set of performance criteria [45–47]. Cassol, et al. [48] proposed a framework to choose evacuation plans based on quantitatively validated metric that captures time, speed, and density of crowd. Galle [49] focused on exhaustively searching possible space layout configurations for small-scale environments. Evolutionary approaches [2], [50] have been used to overcome the infeasibility of brute-force methods for larger design spaces. Liu et al. [51] introduced functional design and fabrication constraints to guide the optimization process. Data-driven approaches [52] learn layout configurations from existing databases. The results are thus used to automatically generate new layouts for computer graphics applications. Design objectives have been modelled as physical forces to generate layout designs automatically [53]. A sophisticated optimization scheme accounts for the visibility, accessibility, and other spatial relationships between objects to produce interior design configurations [54].

Very few studies have incorporated crowd movement characteristics when optimizing environments. For example, in a related work, Feng, et al. [55] concentrated on synthesizing layout, optimizing among different designs (with only outer boundary given) to improve subjective crowd availability. A discrete random forest method with annealing strategy was used in their approach in order to explore different design options. However, this approach is limited to small [56] and mid-scale [55] layouts, due to computational efficiency considerations.

A related family of works concentrate on outdoor layout design. For example, Matthew, et al. [57] proposed a parameterized representation of outdoor environments, where instead of altering the behavioral parameters of the crowd, they tuned the environment that will yield the desired crowd behavior. They provided useful metrics to gauge the layout, with a specific focus on environments with larger scale. Moreover, they considered situations when the crowd's effort to find exits may be disturbed by obstacles and other factors, e.g., lack of prior information about the environment. While they relied on simulations to analyze environment configurations, their work is complementary to our study, which seeks to make predictions of simulation measures.

In a closely related recent work, Testa, et al. [58] proposed a modular framework for architectural design using a neural network-based evacuation time prediction. Their method thus enjoys similar strength as ours, where one does not require ex-

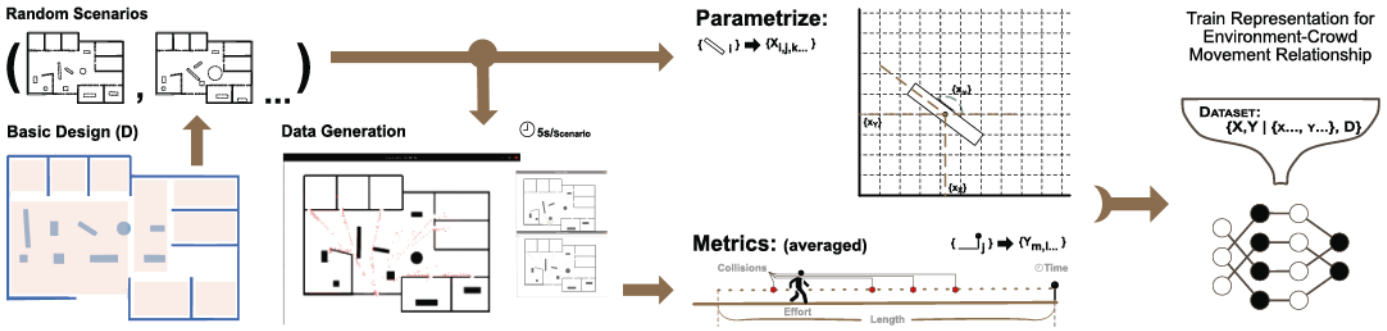


Fig. 1. Overview of the proposed framework for representation learning of the environment-crowd movement relationship. Please refer to the text for details of each step.

haustive computation to obtain evacuation time for a given environment. Moreover, this method can aggregate smaller prediction models to larger structure in a procedural way, formulating it as a flow calculation problem.

The work in [59] presented an interactive tool for floorplan design optimization that considers static metrics such as accessibility, visibility, and organization of space. Due to the practicality considerations described above, it is unable to integrate human behavior metrics extracted from simulations, as part of the design optimization loop, while meeting the interactivity constraints.

2.3. Comparison to Prior Work

Our work strives to generate an efficient algorithm to accurately simulate crowd behavior, and applying these results to optimize a complex building layout in terms of evacuation time and congestion patterns. Our work focuses on tuning basic layout with respect to objective metrics to reduce crowd evacuation time in emergency. To accomplish this, we employed a neural network for speeding up online queries. We also propose an environment-constrained back propagation-based optimization method for fine layout tuning without changing functions of the building while improving the evacuation time with a high accuracy competitive to the full-blown simulations.

Our work is complementary to [59]. Specifically, the user interface and diversity optimization approach can be combined with machine-learning based prediction models proposed in this paper to develop an interactive design tool that considers metrics related to human crowd movement.

Our work shares conceptual similarities with Testa al. [58], but we are tackling the problem from different perspectives, each with their own strengths. We envision that future explorations and practical deployments of such systems will stand to benefit from the ideas presented in both papers. We summarize the main differences below:

1. The work in [58] trains neural networks to predict crowd metrics at the room level, and uses heuristic approximations to aggregate room-level metrics for an entire environment. Our work trains a neural network for an entire environment, supporting globally accurate predictions at an environment-scale.
2. The work in [58] predicts crowd metrics for axis-aligned rooms which are parameterized using three factors (width, height, door width). Our work supports predictions for arbitrarily complex room structures within an environment, including non-axis aligned walls, presence of pillars/obstacles, as well as different crowd configurations. This significantly increases the complexity of the parameter space for learning.
3. Our work proposes a general-purpose optimization framework for automatically reconfiguring environments, using neural-network predictions within the optimization loop.

The work in [58] supports the ability to make globally-approximate predictions for different environment types, composed of axis-aligned rooms. Our approach supports globally-accurate predictions for arbitrarily complex room structures and obstacle configurations, and different crowd types, for a given environment layout, and proposes a method to optimize environments using these learnt metrics as part of the objective formulation.

This paper is a significantly extended version of Liu, et al. [60]. We extend Liu, al. [60] along three major thrusts: (a) we propose a new, integrated framework to optimize given built environment layout, (b) we propose a new measure to gauge the variance of the key metrics, and (c) we report experimental results on real, complex built environment design (Metropolitan Museum of Art) to demonstrate the utility of the proposed method.

3. Proposed Framework

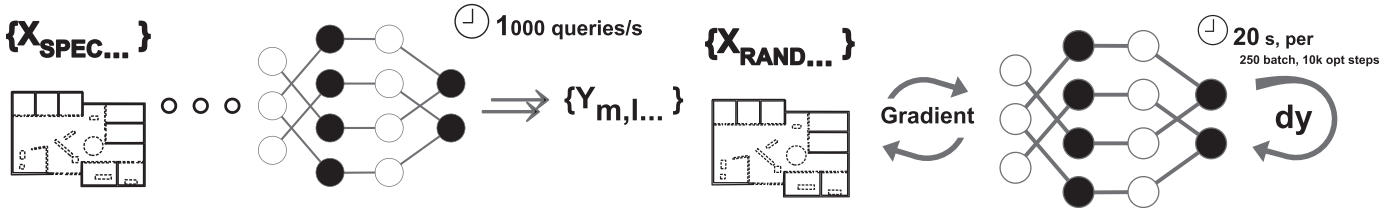
In this section, we introduce our crowd-environment relationship representation learning framework, and its two application examples: (a) crowd evacuation time prediction and (b) automated building design layout optimization. We first provide an overview of the proposed framework and applications, followed by detailed introduction of components consisting the framework.

3.1. Overview

Fig. 1 depicts the overall training procedure of our relationship representation learning framework. Our framework learns the environment-crowd relationship based on dataset generated from simulations. To generate the data, we procedurally generate variations of a parameterized environment with permissible bounds of environmental elements, and run crowd simulators to obtain performance metric values (e.g. time to evacuate the building). Then, we learn the relationship representation connecting the layout and the performance metric values using the deep neural network.

Environment Parameterization. To train the deep neural network, we parameterize the layout as a mixture of characteristic, representative objects. For each scenario, a concrete outer frame is placed in order to define the shape of the room. Several obstacles, with difference setup of absolute or relative position, orientation, and size in each instance, were carefully placed and bounded separately. For example, for the synthetic dataset we used in this work, our defined scenario layout parameter sets consist of the following:

- Circle: With moving area restricted, generate three parameters (x, z, r) control shape of circle, where (x, z) is the origin and r is the radius of the environment object.
- FinHorizon: Two obstacles with "|–" shape, generate two parameters (x, α) controls x position of horizontal shape & rotate angle of verticle shape.



(a) Evacuation Time Prediction (b) Environment Optimization

Fig. 2. Two application of the trained representations.

- FinVerticle: Same As FinHorizon except moving along z axis.
- Horizontal Door: A gap between two horizontal aligned obstacles (as open door), generate x coordinate of the middle point of the door.
- Verticle Door: Same As Horizontal Door except moving along z axis
- Rotate Box: With length of both side specified, generate (x, z) controlling obstacle's center, and (α) controlling obstacle's orientation.
- Joint Walls (Misc A, B, C): In different set several amount of walls connected heads-to-tails. With length of walls specified, the first wall anchored to the outer frame with position specified in x, and position of rest walls specified by their orientation α_i where i denotes the index of obstacles that consist the joint wall.

In the case study described in Section 5, we used slightly different set of parameters because of the nature of the real world data (Metropolitan Museum).

Crowd Motion Model. Within each scenario, we define partitioned areas, where different numbers of agents are randomly placed at initial locations. All agents are instructed to evacuate from the room to a single exit. To account for different types of steering behaviors, we consider two commonly used approaches in our simulations, a social-force based model (**SF**) and an optimal reciprocal collision avoidance method (**ORCA**) to generate simulated data of crowd movement. However, our training procedure is agnostic to the specific simulation technique, and can use other crowd simulation models.

Crowd Behavior Evaluation Metrics. Lots of metrics evaluating the crowd behavior have been investigated in the past. In this work, we study four metrics: (a) average time agent used (referred as "time"), (b) average length agent need to travel (referred as "length"), (c) average amount of each agent collided (referred as "collision"), and (d) average estimated effort agent used (principle of least effort [61,62], referred as "PLE") in order to complete the evacuation. Throughout the paper, we use y to denote this metric. Our approach is capable of predicting *any* metric which can be estimated from crowd movement trajectories.

Representation Learning and Applications. We designed several scenario-wise neural network models which used to predict one or multiple crowd behavior evaluation metric values from layout parameters. Models were trained with a batch of data from simulations. Different types of performance metrics can be used to train the model.

Fig. 2 depicts two applications of the proposed learning framework. After learning the environment-crowd motion relationship representation, we first apply this learned network to predict the crowd behavior evaluation metrics from a new environment as shown in Fig. 2a. We also embedded the trained network to build a meta-optimization framework, that can be used to find the optimal configuration of the obstacles in the room, as shown in Fig. 2b. Here, the environmental parameters that govern the location and orientation of obstacles will be converged to the optimal location and orientation that minimizes the difference between the predicted and optimal crowd evaluation metrics given the current configuration of the obstacles.

3.2. Environment-Crowd Relationship Parameters

We consider three kinds of parameters to describe an environment layout and its associated crowd behavior. The first part of the parameters encodes "environments." Each value in this part represents an x, z coordinates, or the orientation α of an obstacle. The second part is for "crowd", which indicates the number of agents placed in each pre-defined area. The third part describes a set of "agent" configuration parameters used for modeling agents' steering strategy in the simulator. All these parameters will be stacked together to construct the final input vector $\mathbf{p}$ to the neural network. We also define $\mathcal{P}$ as the set of all possible scenario instances (i.e., varying obstacle location/orientation, agent density, and steering configuration), for the given layout definition (i.e., fixed obstacle type/count and agent steering model). We assume that all sampled cases $\mathbf{p} \in \mathcal{P}$ can be represented as data points which are uniformly distributed within the space.

3.3. Extended artificial neural network

In a prior study [60], Liu et al. examined the possibility of utilizing machine learning methods to evaluate environment evacuation scenarios. In this work, we applied deeper neural network architectures and Maxout activation function [63] to obtain better performance by model averaging. This deep network structure has six layers, with ten Maxout cells in each layer, formally defined as:

$$\mathbf{h}_k = f_k(\mathbf{h}_{k-1}), \quad k \in 1..6 \tag{1}$$

where $\mathbf{h}_k$ denotes the output of k-th layer, and $\mathbf{h}_0 \in \mathbb{R}^d$ denotes the input of the network with d features, i.e., an instance of $\mathbf{p}$. Each k-th layer $f_k(\cdot)$ consists of a linear weight matrix $\mathbf{W}_k \in \mathbb{R}^{l \times m}$ followed by a Maxout function g_k as

$$f_k(\mathbf{h}_{k-1}) = g_k(\mathbf{W}_k\mathbf{h}_{k-1}) = \begin{bmatrix} g_{k,1}(\mathbf{W}_k\mathbf{h}_{k-1}) \\ g_{k,2}(\mathbf{W}_k\mathbf{h}_{k-1}) \\ \vdots \\ g_{k,10}(\mathbf{W}_k\mathbf{h}_{k-1}) \end{bmatrix}_{10\times 1} \tag{2}$$

where $g_{k,j}(\cdot)$ for $j \in 1..10$ is a max pooling for the j-th portion of ten evenly divided partitions of the input vector $\mathbf{W}_k\mathbf{h}_{k-1}$. Thus, m is d for $\mathbf{W}_1$ and 10 for the others ($\mathbf{W}_2 \cdots \mathbf{W}_6$). We used $l = 1,000$ in this work. Output of the last layer is linearly combined to regress against the performance metric y given input vector $\mathbf{h}_0$. This network structure allows us to accelerate the training process by applying the batch gradient descent.

To evaluate the trained model, we computed the root-mean-squared error (RMSE) on the test split of the dataset. By comparing the predicted value y with the simulated value $\hat{y}$, the predictive deviations can be expressed as

$$RMSE(y, \hat{y}) = \sqrt{\mathbb{E}\left[(y - \hat{y})^2\right]}. \tag{3}$$

Note that $y, \hat{y}$ can be vectors $\mathbf{y}, \hat{\mathbf{y}}$ when predicting multiple evaluation metrics simultaneously. R_{test} values in our results denote the RMSE on the test split. Note that we are computing the empirical mean of the sample deviations, not the full expected value in the experiments.

3.4. Evaluating prediction robustness

Our crowd simulations include parameters that are randomly chosen such as the agents initial conditions, or internal decision by the agents such as the choice of steering strategy. These may introduce source-based variance in the value of the predicted metrics. In other words, predicting the mean of such metrics may not be enough to ensure the robustness of our trained model. To see whether our model's metric predictions are within the deviation originated from the data source specified by the parameter $\mathbf{p}$ as described in Sections 3.1 and 3.2, we propose a relative measure that gauges this property.

First, we make an assumption that our data source (either crowd simulator or real-world capturing device) has an independent additive noise with a fixed, unknown variance σ^2 that varies the metric we measure using the source. We model the distribution of the metric random variable Y given certain parameter setting $\mathbf{p}$ as a Gaussian distribution with a fixed, unknown standard deviation σ independent of $\mathbf{p}$, i.e.

$$p(Y|\mathbf{p}) = \mathcal{N}(\mu_{\mathbf{p}}, \sigma^2), \tag{4}$$

where $\mu_{\mathbf{p}}$ denotes the mean of Y given $\mathbf{p}$. The simplest way to estimate σ is through empirical simulation trials that use the same parameter configuration. Next, we consider another random variable Y' given the same parameter $\mathbf{p}$ as

$$p(Y'|\mathbf{p}) = \mathcal{N}(\mu_{\mathbf{p}}, \sigma^2). \tag{5}$$

Then, one can show that the difference between the two random variables, $\tilde{Y} = Y - Y'$, will follow another Gaussian distribution with variance $2\sigma^2$ as

$$p(\tilde{Y}|\mathbf{p}) \sim \mathcal{N}(0, 2\sigma^2), \tag{6}$$

because the means of Y and Y' are the same. Let $\sigma^2_{\tilde{Y}}$ be the variance of $\tilde{Y}$. Naturally, $\sigma^2 = \sigma^2_{\tilde{Y}}/2$. To this end, our idea is to estimate $\sigma^2_{\tilde{Y}}$ instead of directly estimating σ^2. This can be done by collecting $\tilde{Y}$ over various parameter configurations $\mathbf{p} \in \mathcal{P}$, by conducting two simulation instances per each $\mathbf{p}$. Once the empirical estimate for $\sigma^2_{\tilde{Y}}$ is calculated, one can take a half of it to obtain the estimate of σ^2.

To measure the robustness of the metric prediction relative to the estimated data source variance σ^2, we propose the following measure:

$$R_{rel} = 2 \cdot \log_2 \frac{RMSE(y, \hat{y})}{\sigma}. \tag{7}$$

where $\hat{y}$ is a simulated metric value since

$$RMSE(y, \hat{y}) = \sqrt{\mathbb{E}\left[(y - \hat{y})^2\right]} \geq \sqrt{\mathbb{E}\left[(\mu - \hat{y})^2\right]} = \sigma, \tag{8}$$

we can see that R_{rel} will have the best value of zero when our model prediction is perfect ($RMSE(y, \hat{y}) = RMSE(\mu, \hat{y})$), and will become one when the $RMSE(y, \hat{y}) = \sigma_{\hat{Y}} = \sqrt{2}\sigma$. Essentially, the smaller R_{rel} is, the better the performance. In the supplementary material, we provide additional test results on the dataset we generated to show that the Gaussianity assumption on the data source noise is reasonable.

3.5. Built Environment Layout Optimization Algorithm

Our building layout optimization framework is summarized in Algorithm 1 . Given a trained model $\mathcal{M}(\cdot)$ and initial obstacle overlap counter $\mathcal{C}(\cdot)$ for the initial layout parameter $\mathbf{p}$, our goal is to find the optimal $\hat{\mathbf{p}}$ that minimizes both the desired performance measure (e.g. evacuation time) and the number of collisions among the obstacles in the given layout. The parameter $\mathbf{p}$ can be represented in one of the two formats of a set of obstacle parameters $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_o\}$. One is the *rolled up* format, where $\mathbf{p}_i$ is the parameter vector describing object i, and o is the total number of obstacles in the environment. In this format, we optimize parameters for each obstacle in the layout, regardless of objects' type. The other is the *group-by-obstacle* format where each $\mathbf{p}_i$ denotes parameters that controls all instances of object type i. The obstacle overlap counter $C(\cdot)$ will return obstacle collision/overlap information for individual obstacle basis (roll up) or obstacle type (group-by-obstacle), and O_i in algorithm denotes the overlap counter for the obstacle instance or type i.

Algorithm 1: Batch gradient optimization algorithm for building layout optimization.

Data: Sample batch size $b = 250$;Total number of iterations T; Trained model $\mathcal{M}$;Obstacle overlap counter $\mathcal{C}$;Layout parameter $\mathbf{p}$;
Result: Parameters for the optimized layout parameter $\hat{\mathbf{p}}$

1 Uniformly sample batch of size b, $\mathbf{p}^{(0)}$ from the parameter space;
2 **for** t **in** *0 to (T-1)*:
3 $O = \mathcal{C}(\mathbf{p}^{(t)})$;
4 $y = \mathcal{M}.forward\ (\mathbf{p}^{(t)})$;
5 **let** loss of y be y itself;
6 $\tilde{\mathbf{p}}^{(t)} = \mathcal{M}.backward\ (loss)$;
7 $\tilde{O} = \mathcal{C}(\tilde{\mathbf{p}}^{(t)})$;
8 **for** i **in** *(all obstacles instances* ***or*** *all obstacle types)*:
9 **if** $\tilde{O}_i \leq O_i$:
10 $\mathbf{p}_i^{(t+1)} = \tilde{\mathbf{p}}_i^{(t)}$;
11 **else:**
12 $\mathbf{p}_i^{(t+1)} = \mathbf{p}_i^{(t)}$;
13 **return** $\mathbf{p}^{(T)}$ as the found $\hat{\mathbf{p}}$;

With these input and subroutine definitions, the algorithm works as follows: (Line 2) Iterate lines 2–12 predefined maximum number of iterations. (Lines 3) Count obstacle collisions/overlaps based on the current parameter description. (Line 4) Using the pretrained metric prediction network, obtain current metric (e.g. evacuation time). (Lines 5 and 6) Compute the loss. Note that the loss should be carefully determined for the Algorithm 1 to work properly. If the desirable layout is to minimize the evacuation time, the loss should be the prediction output y itself because its lower limit is zero. (Line 6) Backpropagate the loss to obtain new parameter values. To compute the gradient in each layer, one can use the standard back propagation algorithm [64]. (Line 7) Counts obstacle collision/overlaps based on the new parameter description. (Line 8–12) For each obstacle instance (or obstacle type, depending on the input parameter format), choose the parameter that yielded less collisions or overlaps. (Line 13) Return the last parameter vector.

3.6. Sensitivity on Network Model Hyperparameters

To find the best combination of model hyperparameters (number of layers), we conducted extensive experiments with the varying number of layers on all 64 configurations, shown in Table 5. We ran for number of layers in the range of 1..10, and found that with an exception of the 1-layer case, the number of layers does not impact our framework's performance significantly. It is reasonable to see that the 1-layer case performs poorly since the model cannot learn ade-

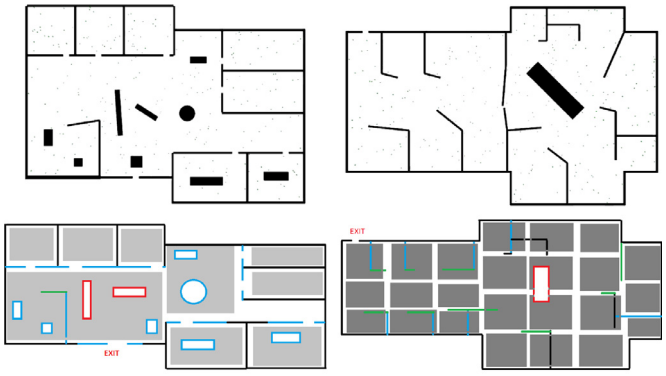


Fig. 3. Two synthetic layouts designed for training. Please refer the text for details. These designs were also used in [60].

quate representations with such a shallow network. Detailed results of these experiments are included in the supplementary materials.

4. Experiments

In this section, we present our experimental results on the synthetic layout dataset, for crowd behavior evaluation metric prediction and layout optimization. We start our discussion with description on our synthetic dataset generation procedure.

4.1. Dataset Generation

Generating appropriate, and realistic dataset is both essential and critical part of the framework. As we are targeting at evaluations to the crowd evacuation cases, we first prepared two simplified, synthetic environment layout designs which meet our requirements. In each layout design, there is a fixed boundary around the room. Several movable obstacles are placed within the boundary, of which movements are restricted by the range of location, or pinned in a joint with another obstacle. Agents are randomly placed in several pre-defined areas before a simulation instance starts. Gray areas shown in Fig. 3 are pre-defined agent areas where agents are placed at the beginning of the simulation and the number of agents within each region is counted separately; blue obstacles (lines and shapes) can only perform translation; red obstacles are allowed to be moved or rotated; green obstacles are anchored on fixed or translatable obstacles, and are only allowed to rotate along with their joint points. There are in total of 10 separate areas in the Map A and 27 areas in the Map B. The default number of generated agent in total are 250 in Map A and 239 in Map B. For **Cwd** dataset (where we can have random number of agents in the scene), in all cases, agents between 0 and double of the default number are generated, following a uniform distribution. Train, validation, and test splits were: 65%, 20%, 15%, respectively.

Gathering sufficient amount of large scale data is critical for the data-driven neural network model training. However, obtaining a large set of records from real human experiments is very challenging. Thus, we utilize a crowd simulator [65] to produce more systematic result for the research. The simulator uses A* algorithm to find initial global path guidance for agents using SF and ORCA. We generated 40,000 different scenarios for each combinations of Map, steering algorithm, and factors, totaling 960,000 instances. After eliminating scenarios which agents failed to evacuated from the map, we collected dataset as their size summarized in Table 1.

In this work, several synthetic designs and one real-world design are used for the experiments and demonstrations. We used following notation to distinguish designs throughout the rest of paper: In synthetic case, two maps were created either room-oriented (Map A) or screen-partitioned (Map B) as depicted in Fig. 3. Different factors are considered to link the synthetic design to the practical applications. Environment (**Env**) factors are about different obstacle placement. Crowd (**Cwd**) factors respond to different cases of agent density in building, Agent (**Agt**) factors considered that people may have different characteristic in their moving ability, and abstracted those abilities into model specified parameters. When all factors are considered, we referred it to **All**. These are the same conditions used to generate dataset in [60].

4.2. Results on Metric Prediction

We used Tensorflow [66] to implement our training framework. The neural network model consists of five layers of Maxout 10-unit output with 100 nodes in each unit, followed by a fully-connected layer linked to the output. We used 256 for the random mini-batch for the training, with the ADAM stochastic gradient algorithm [67] for the optimization. L2 regularization is used in order to suppress overfit, as well as to improve performance on validation and test splits of the dataset. Regularization parameter λ is set to be linearly increased with a very small step ($0.0001 \approx 0.001$ per epoch), and we determined this by conducting experiments on validation set. We stopped the iterative optimization when the cross-validation error does not decrease any further. This practically happens around after 20,000 iterations. We used the following loss function during training, where N is the batch size:

$$\mathcal{L} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2} + \lambda||W|| \tag{9}$$

To obtain $\sigma_{\tilde{Y}}$ for each configuration, we sampled another 5000 points and run simulation twice for each point. After all failed sim-

Table 1
Number of scenarios and dimensions of parameters in our synthetic dataset. We also report amount of computation time to run these simulations. Numbers in the every other row shows mean and standard deviation of the amount of hours needed for simulation, with 100 batch jobs running simultaneously. Each job will simulate 400 scenarios. Overall, we ran 960,000 simulations in total and it took a little more than 3 hours using HPC.

Map	Environment (**Env**)		Crowd (**Cwd**)		Agent (**Agt**)		All	
	Scenarios	Parameters	Scenarios	Parameters	Scenarios	Parameters	Scenarios	Parameters
A - SF	39,891	31	39,999	10	39,539	12	37,589	53
Time		1.26 ± 0.05		1.84 ± 0.37		2.22 ± 0.04		1.59 ± 0.04
B - ORCA	34,450	22	38,711	27	27,961	4	25,261	53
Time		1.30 ± 0.32		0.79 ± 0.01		0.85 ± 0.01		0.86 ± 0.02
A - ORCA	25,265	31	39,989	10	38,652	4	23,766	45
Time		0.58 ± 0.01		0.54 ± 0.01		0.62 ± 0.01		0.67 ± 0.02
B - SF	36,043	22	38,535	27	28,469	12	27,030	61
Time		1.79 ± 0.10		1.50 ± 0.09		1.50 ± 0.12		1.83 ± 0.09

Table 2
Training time (in hours) for the prediction network of particular metric. Note that this time includes both the training and cross validation time needed to find the best parameter value λ in Eq. 9, thus it is the total amount of time needed for training the network.

		SF				ORCA			
		Time	Collision	Length	PLE	Time	Collision	Length	PLE
Map A	**Env**	2.75	3.06	3.01	3.02	1.95	0.03	0.10	1.79
	Cwd	2.71	0.01	2.90	0.01	2.95	2.86	2.97	0.02
	Agt	0.51	0.22	0.30	0.32	1.31	0.84	0.34	1.18
	All	0.89	2.07	0.11	0.24	1.75	0.02	1.82	1.82
Map B	**Env**	3.12	1.96	3.35	3.44	3.24	3.29	2.81	0.05
	Cwd	3.72	1.63	3.65	3.71	3.66	3.70	3.66	2.75
	Agt	0.60	0.54	0.32	0.44	0.50	0.08	1.03	0.53
	All	1.02	1.75	1.46	2.79	2.34	0.01	2.67	2.33

Table 3
Dataset generated for repeated test; in this type of dataset, each layout were simulated exactly twice. These are selected based on two-step pre-processing on the generated simulations: (Step 1) For every simulation instance, mark if all agent within the instance successfully reached the target within set max number of frames. (Step 2) For each pair of two-trial pairs, take all simulation pairs if both instances are marked success in Step 1.

Map	Env	Cwd	Agt	All
A - SF	9 955	9 999	9 861	9 359
B - ORCA	8 249	9 468	6 704	6 053
A - ORCA	5 807	9 996	9 563	5 531
B - SF	8 968	9 317	6 638	6 646

Table 4
Comparison of results on A-SF-All configuration using datasets with different sizes. Numbers in parenthesis indicate the size of the dataset. Unit for each measure is indicated within the parenthesis.

Dataset	Time (s)	Collision (#)	Length (m)	PLE (J/Kg/s)
R_{test} (30k)	3.23	11.57	3.56	13.18
R_{test} (68k)	2.77	9.43	3.23	10.75
R_{rel} (30k)	1.6	3.5	2.0	1.9
R_{rel} (68k)	1.1	2.9	1.7	1.3

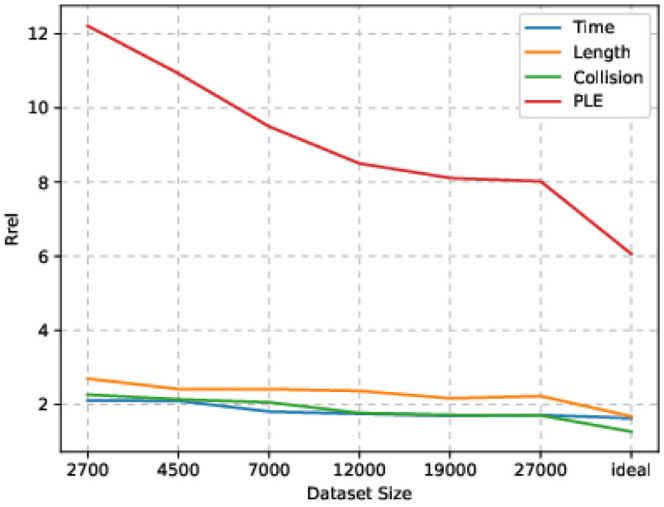


Fig. 4. R_{rel} performance as a function of increasing the dataset size on **A-SF-Env** configuration, using different metrics.

ulations filtered out, the size of each simulation outputs are summarized in Table. 3. W is the weights of the network that we impose regularization constraint.

Training Results. We present our evaluation results in Table 5. In the case of single configuration learning, cases in **Env** have a moderate to well performance based on our evaluation on R_{rel}. Values near to 1 suggests that our framework's prediction variance is as good as one instance of simulation from the simulator would yield. In **Cwd** cases, the proposed method performance well based on our evaluation on R_{rel}. Values near to 0 suggests that our predictions are close to the mean of the simulations. In **Agt** configuration, results indicate that our framework struggled on learning patterns from SF model using the parameter configuration, but indicates a good performance in ORCA model. In this case, the R_{rel} went down to negatives values. We believe that the neural network found the statistical means from parameter-dependent variance which breaks our parameter-independent variance assumption, due to the extremely small parameter space of ORCA-Cwd configuration which has only four dimensions. Refer Table 1 for numbers of parameters in all configurations.

The accuracy of predictions is relatively lower when using the PLE metric. This may mean that the network did not learn the agent speeds correctly which has a significant influence on the results. While the incorrect velocity estimate can affect the time and length metrics as well, given the definition of PLE [61], large speed variation will introduce more PLE error than the other metrics.

In the cases of **A-SF-All** and **B-SF-All** cases, the results were not as good as the individual configurations. We posit that this can be the case both due to the difficulty to learn social force model and the lack of data due to its relatively high number of parameters. To prove this hypothesis, we conducted additional experiments with the expanded dataset of size 68K instead of 30K, and observed a significant performance improvement as shown in Table 4. We also conducted experiments using increasing data set size for an easy case (**A-SF-Env**), in order to see the performance changes relative to the amount of input data fed into the neural network. Table 4 and Fig 4 summarize this result. It shows that, with an exception of PLE, our model is insensitive to the amount of training data.

Multi-output prediction. In addition to the application for a specific evaluation metric prediction, we also investigated potential of the proposed framework in making multiple predictions simultaneously. The neural network structure is mostly the same as in the single output value setup, but now we have a structured output of four values stacked as a vector **y**. We tested two types of optimization strategies with respect to use of the gradient descent algorithm. The first one is the same gradient descent algorithm we used before. The other one scaled backward gradient flow from four targets with the inverse of their own $\sigma_{\tilde{Y}}$. This is to let gradient descent algorithm equally weigh contributions of the losses among all prediction outcomes. We present additional discussions in the supplementary material.

Table 5
Prediction results of combined configuration set.

		Map A-ORCA			Map A-SF			Map B-ORCA			Map B-SF		
	Metric	$\sigma_{\bar{Y}}$	R_{test}	R_{rel}	$\sigma_{\bar{Y}}$	R_{test}	R_{rel}	$\sigma_{\bar{Y}}$	R_{test}	R_{rel}	$\sigma_{\bar{Y}}$	R_{test}	R_{rel}
Env	Time	2.61	2.20	0.50	2.40	2.06	0.55	1.38	1.41	1.06	0.91	1.61	2.65
	Collision	3.57	3.27	0.74	1.81	2.36	1.76	1.45	1.45	1.01	0.36	1.21	4.53
	Length	2.68	2.20	0.43	2.46	1.96	0.34	1.28	1.40	1.26	1.03	1.47	2.02
	PLE	9.94	8.49	0.54	8.91	7.45	0.48	5.67	5.99	1.16	3.60	5.65	2.30
Cwd	Time	2.32	1.68	0.07	2.34	1.72	0.12	1.33	0.97	0.10	0.84	0.65	0.25
	Collision	3.01	2.17	0.05	1.81	1.35	0.15	1.54	1.11	0.05	0.11	0.08	0.11
	Length	2.39	1.74	0.08	2.41	1.78	0.12	1.25	0.91	0.09	1.00	0.76	0.23
	PLE	9.05	6.52	0.09	8.71	6.57	0.19	5.45	4.23	0.26	3.43	2.92	0.53
Agt	Time	2.43	1.73	0.02	2.78	2.14	0.25	1.23	0.89	0.07	0.84	0.71	0.53
	Collision	2.98	2.09	-0.02	3.86	4.03	1.12	1.42	1.05	0.13	0.28	0.54	2.94
	Length	2.34	1.67	0.02	2.57	1.91	0.14	1.24	0.90	0.07	0.99	0.74	0.16
	PLE	9.25	6.57	0.01	9.67	7.14	0.12	5.33	4.25	0.35	3.44	2.63	0.22
All	Time	2.67	2.74	1.07	2.65	2.91	1.26	1.24	1.87	2.19	0.91	2.01	3.29
	Collision	3.74	4.26	1.38	4.88	8.53	2.61	1.43	2.40	2.49	0.76	2.64	4.57
	Length	2.65	2.59	0.93	2.59	2.77	1.20	1.28	2.17	2.52	1.05	1.77	2.51
	PLE	10.36	10.37	1.00	9.59	10.50	1.26	5.41	9.28	2.56	3.66	7.55	3.09

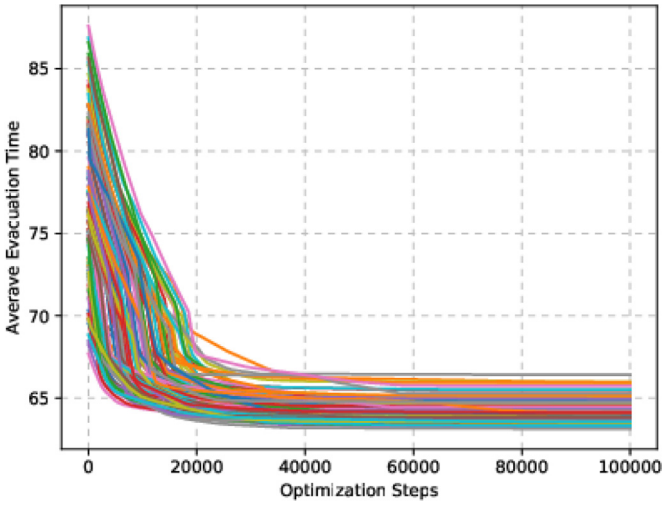


Fig. 5. Evacuation time-based Optimization of **A-SF-ENV** environment layout. Each line indicates an initial configuration of obstacles. Average evacuation time of agents is the loss used to optimize the environment because the smaller the time, the better the output.

Table 6
Results on constrained layout optimization. Numbers are statistics of predicted evacuation time over different initial crowd configurations. It is clear that the layout optimization made the prediction robust.

	Mean	Std. Dev.	Min	Max
Before	78.68	4.34	68.82	86.87
After	63.85	0.46	63.16	65.34

4.3. Results on Layout Optimization

For the layout optimization experiments, we selected the model for **A-SF-Env** as the test model for this application. Other configurations should work equally well. The goal of this demonstration is to show that, our optimization will find the layout that will minimizes the evacuation time as well as improve the robustness of the evacuation time predictions. Fig. 5 shows the convergence trend of the predicted evacuation time (in seconds) averaged over all agents. The plot shows optimization trend for all 50 independent initial configurations of obstacles. After 20,000 or so iterations, the layout optimization converged. Table 6 summarizes the statistics of predicted evacuation time over different initial crowd configurations. In Fig. 6, some sample snapshots during the optimization experiments are provided as qualitative results.

Table 7
Evacuation time in seconds per agent. For each optimized layout, we ran evacuation time prediction as well as a simulation. For comparison, predicted and simulated evacuation time for the original expert layout as well as additional 100 random layouts are shown. The results indicate that the optimized trajectories have competitive evacuation time compared to that of expert layout, significantly better than random case. In addition, large variance in random cases indicate that the obtained result is not a coincidental outcome.

	#Layouts	Prediction	Simulation
A	2	66.49 ± 0.00	68.78 ± 0.41
B	22	66.45 ± 0.02	66.92 ± 0.71
C	40	66.25 ± 0.03	66.90 ± 0.92
D	14	66.24 ± 0.01	67.13 ± 0.80
Expert	1	71.07	70.91 ± 1.44
Random	100	72.50 ± 6.49	72.36 ± 7.75

5. Case Study: Metropolitan Museum

In this section, we show that our framework can be applied to a real world built environment design. We trained the proposed neural network with a new dataset generated from a more complex built environment layout design using a quarter of expert-designed blueprint (Metropolitan museum) as the base structure. We show that the artificial neural network framework is not only able to estimate crowd behavior evaluation metric for an input layout in milliseconds with high accuracy, but also can be a part of any optimization (and advising) system in lieu of a more computationally expensive crowd simulator.

5.1. Parameterization of Layout

First, we designed the parametric layout from the expert-designed layout of Metropolitan museum to generate the dataset. The Metropolitan layout is very large so applying our framework to the whole environment would be infeasible due to computational cost. Therefore, a quarter part of the Metropolitan was used in our study. The complete design of Metropolitan shown in Fig. 7.

Next, we made a relaxed design of a portion of the original layout. With several anchors set on different group of obstacles or walls, we can make a more complex design from this relaxed design. Here, in order to move around each obstacle as we desired, we set up several reference points. Moreover, the position of each obstacle or joint of walls are movable with respect to the motion

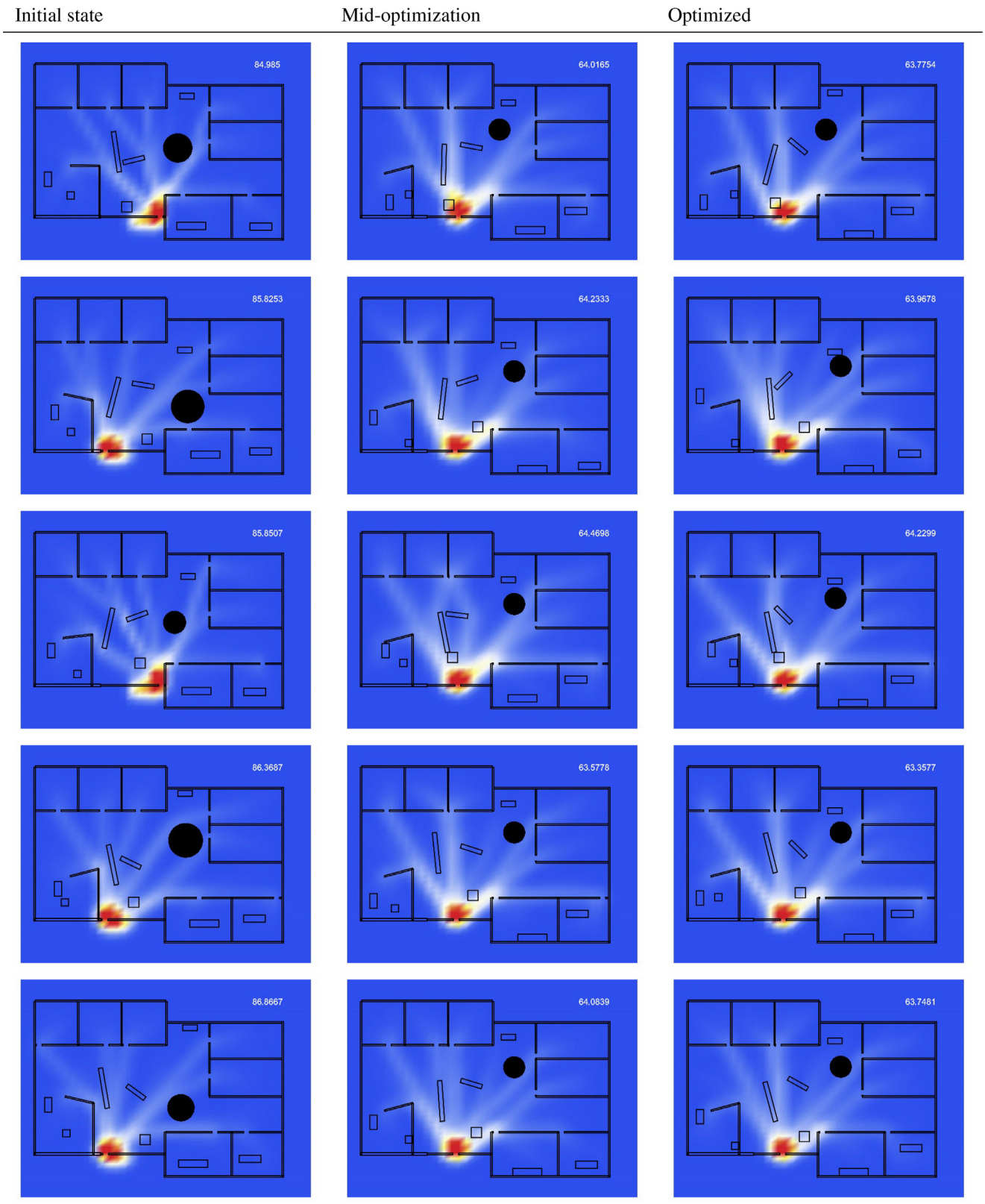


Fig. 6. Snapshots of the layout optimization and overall crowd movement heat map on the synthetic layout. Hot color indicates high density of the agents. Each row depicts a different initial configuration. Numbers in white indicate the predicted metric (evacuation time in seconds) at that state of obstacles. Note the changes of location and angle of obstacles and the metric as the optimization progresses. Black disc indicates a round-shaped table furniture. Note that the heatmaps were generated by separate simulations on the layouts, independent from optimization progress.

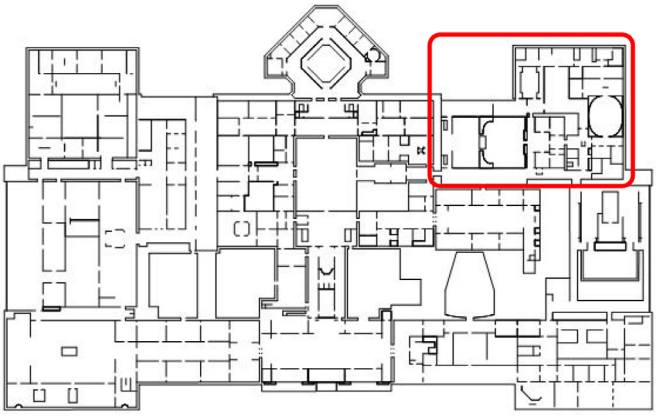


Fig. 7. Complete layout of Metropolitan Museum of Art. We focus on the top-right corner of the whole environment in this study.

of reference points. Thus, the parameter space becomes a span of constrained position of reference points. In addition, we used the notion of a set of obstacles as a group, which can be moved with respect to an invisible reference point. The X, Y positions of the reference point are recorded and aligned, shown as another input dimension to the simulator. The final layout we used is shown in Fig. 8 (Top).

For the evacuation simulation, we also redefined the areas where agents are placed initially. The whole building layout was divided into seven areas. In each area, the number of agents placed is proportionally determined relative to the area of region. Total number of agents was predefined as 240, equally, for each of the sample cases. Target of each agent is assigned so that to encourage the agent leave the building using the nearest gate. Fig. 8 (Bottom) depicts this agent and goal location setup.

5.2. Results on Metric Prediction

We now present our results on metric prediction experiments on the Metropolitan layout. First, we generated the dataset with randomly generated layout designs based on the parametric representation of the layout described earlier. About 122,000 random layout samples were generated and each layout was simulated to obtain the evacuation performance metric (in seconds) using an open source crowd simulator [68]. We split the 122,000 samples into train, validation, and test set following the same split distributions (65%, 20%, 15%, respectively) as we did in the synthetic experiments. Then, we trained our metric prediction neural network, using the proposed framework. After the training, we obtained results of $\sigma_{\tilde{Y}}$ is 2.76, R_{train} is 2.84, and R_{rel} is 1.1. The loss of the training, cross validation, and test dataset as a function of number of training iterations is shown in Fig. 9.

5.3. Results on Layout Optimization

Using the trained crowd-environment relationship representation network, we evaluated the proposed layout optimization algorithm on the Metropolitan museum layout. Our goal here is to see whether the proposed optimization algorithm can find an optimized layout that has similar property as the expert-designed original layout. As an example of such properties, congested region during evacuation is depicted as hot colored pixels in Fig. 11. As it can be seen, expert layout has distinctive heatmap pattern different from an arbitrary random layout. Thus, one can utilize this congested region heat map visualization as good qualitative

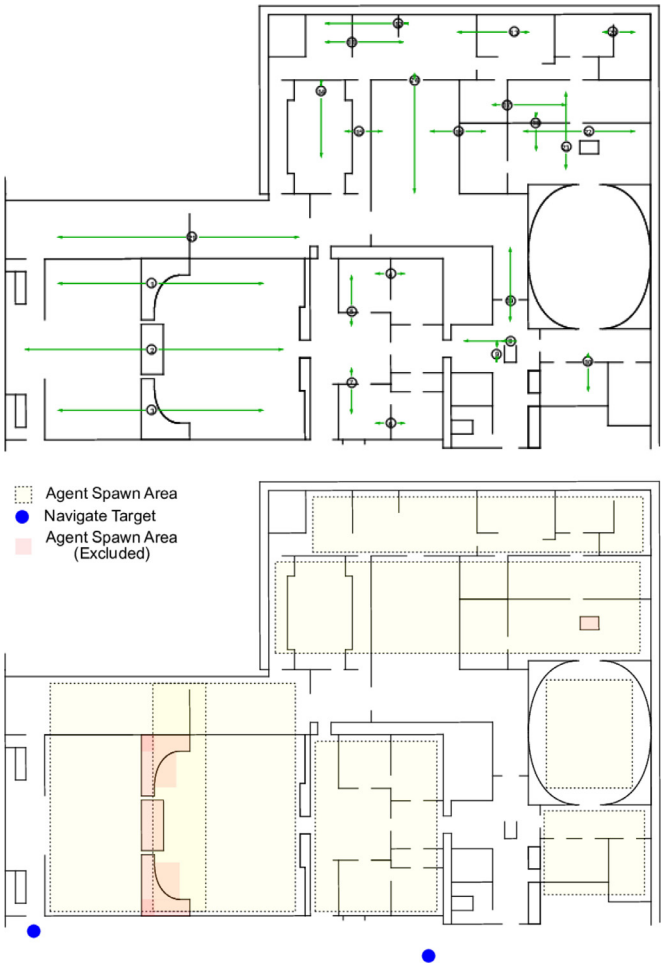


Fig. 8. (Top) Relaxed layout design of the selected area of the Metropolitan museum. Green arrows indicate the possible variations of the environmental structures. Circles indicate the location of invisible reference points. (Bottom) Definition of agent areas. Note that the map has two exits where agents can choose the best option for evacuation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

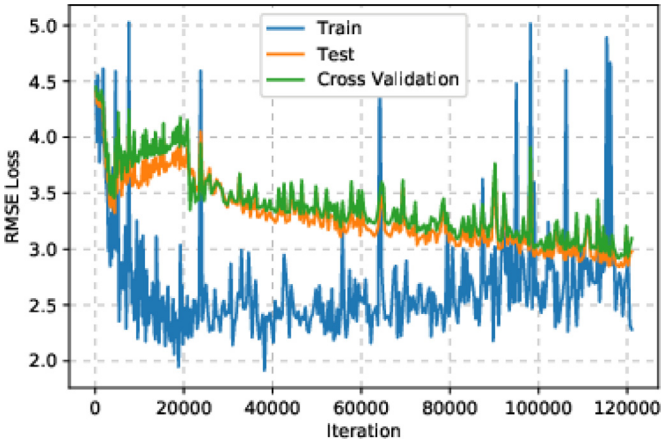


Fig. 9. Loss value (evacuation time difference compared to the simulation output, in seconds) change over iterations during the training process. We stop training when the cross validation performance does not improve further. Near the end of the plot, one can see that the cross validation error increases.

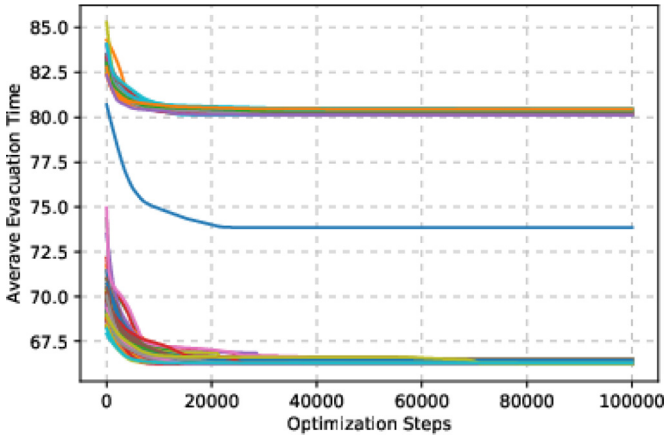


Fig. 10. Batch gradient optimization of Metropolitan layout. Each line represents one optimization instance of a random layout, thus 100 lines are plotted in the figure. The standalone blue plot indicates an outlier case when the particular obstacles are initialized in a way that can block the entire flow during the optimization, which is a very rare case. This can be easily avoided either by sampling or carefully chosen initialization. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

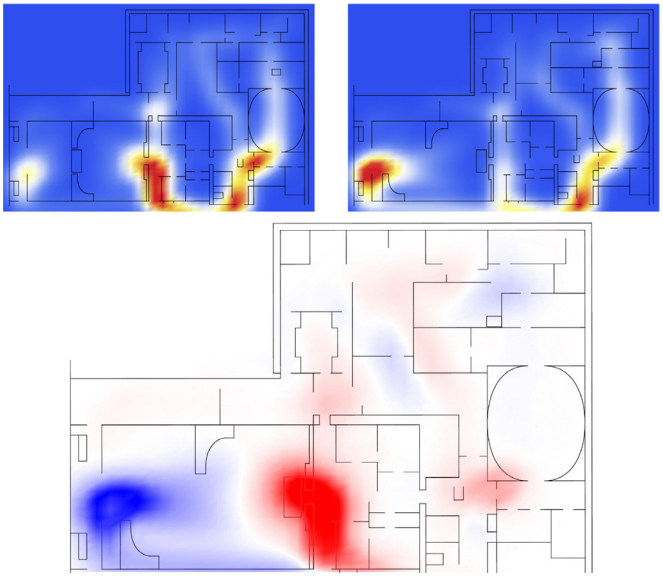


Fig. 11. Top-Left: Heatmap (crowd density) snapshot of expert design; Top-Right: Heatmap of an arbitrary random layout; Bottom: Comparison between the simulated densities between expert and random layout. Blue indicates false positive (exists in compared target but missing in expert layout) and red indicates false negative (exists in expert but missing in the compared target) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

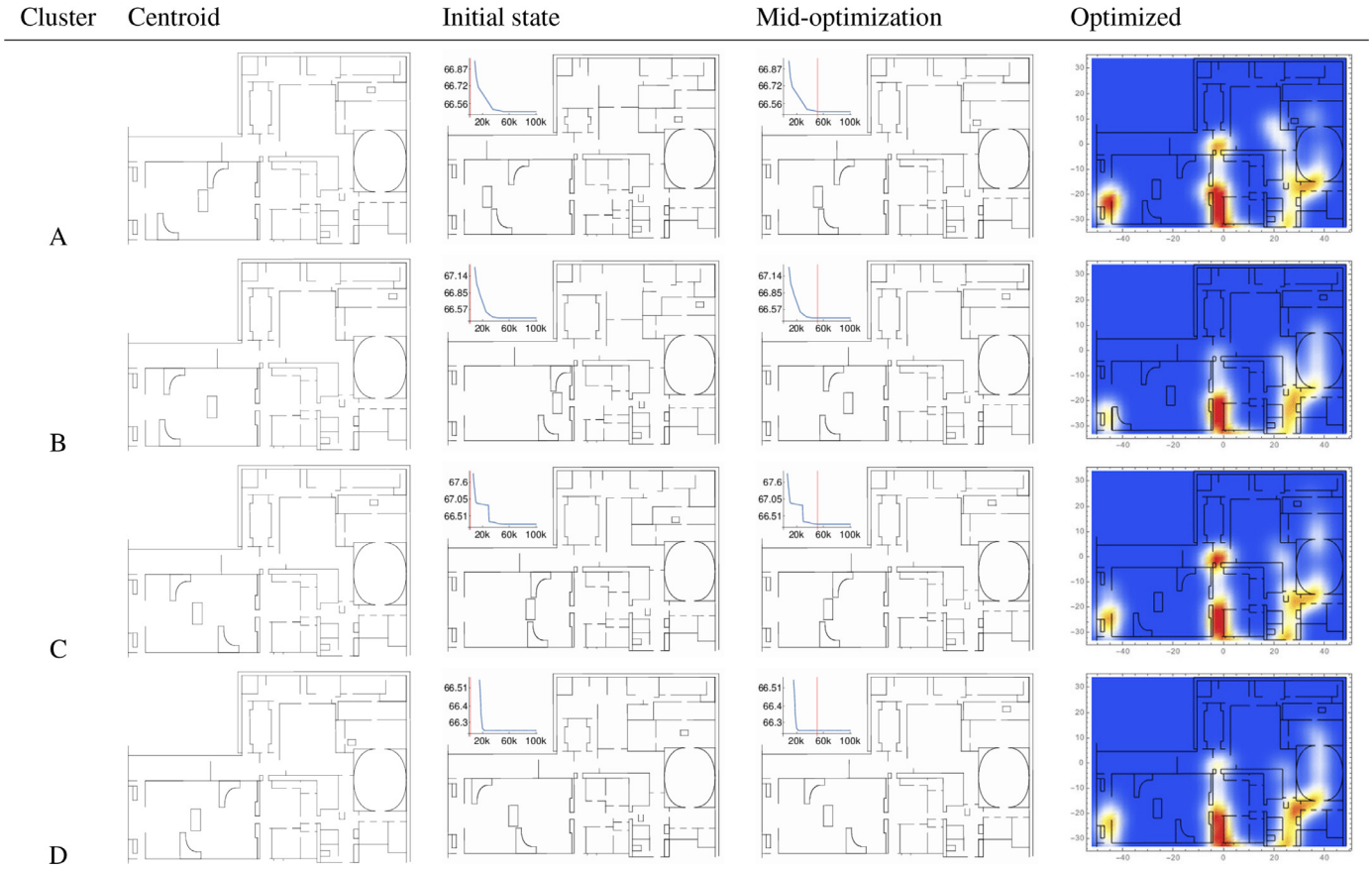


Fig. 12. Snapshots during the process of optimization and heat map of the final, optimized layouts. Plots of insets show neural network estimated evacuation time (in seconds) along with optimization steps (vertical axis), and the vertical red line indicates when the snapshot was taken throughout the iteration of the optimization (horizontal axis). Heat map shows that the final optimized layout makes reasonable design showing jammed area is limited around the exit (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

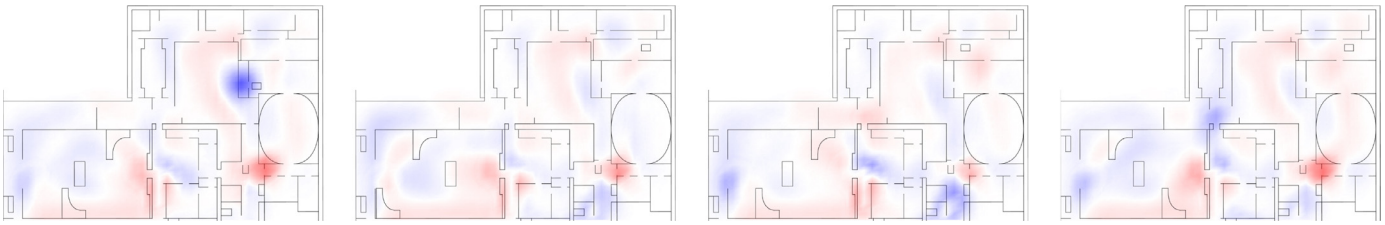


Fig. 13. Simulated density differences between expert and optimized layouts (A–D from left to right). It is clearly visible that the density difference has been reduced compared to the random situation shown in Fig. 11, i.e. optimization has found layouts that are closer to the expert layout. One can also observe that the crowd occupancy density pattern is distinct between the two optimized layouts. Each of these optimized layouts yielded better predicted average evacuation time than the predictions on corresponding initial states, as shown in Fig. 12 and Table 7.

evidence that our optimized layout has similar characteristic to that of the expert-designed layout.

To achieve this goal, we first generated 100 random layouts as sub-optimal layout samples. For each run, our optimization algorithm is initialized with one of the random layouts, and then start the layout optimization. When optimizing the layouts, we used the metric prediction network to predict the evacuation time for interim layout during the optimization. The network was pretrained using the training sample layouts described in the previous section.

Fig. 10 shows the average evacuation time trend for all 100 newly generated layouts as a function of optimization progress. Since the layout optimization task is non-convex, there are a few cases when the algorithm converged to sub-optimum solutions. To obtain further in-depth understanding of the optimized layouts, we first applied mean shift [69] algorithm on the layout parameters to identify clusters of layout samples that share the similar latent property. The mean shift algorithm applied on the 100 samples yielded four clusters.

Fig. 12 shows the progress of the optimization as well as the centroid within each cluster of the sample initial layouts we used to test our optimization framework. We randomly selected layouts with a random set of parameters, e.g. the one shown in the top-right of Fig. 11, which is obviously distinct from the expert-designed layout. The neural network is able to efficiently obtain layouts comparable in evacuation time and the flow patterns to the expert design in only 22 seconds. Four final, optimized layouts in clusters A-D all yielded better predicted average evacuation times than the predictions on corresponding initial states as shown in Fig. 12. Inset plots within layout images show the progress of evacuation time improvement as the optimization progresses. Fig. 13 shows the visual similarity between the expert layout and each of the four optimized layouts. It is clear that the extreme differences between the expert and the sample layout that exist in Fig. 11 have been removed.

We also verified the accuracy of the evacuation time predictions by running simulator-based simulations on each of these layouts. Table 7 summarizes the result comparing the predicted evacuation time versus the actual evacuation time obtained by the simulation using SF as the steering algorithm. We also report the predicted and simulated evacuation times of the original, expert-designed layout and the random, arbitrary layout to show that the optimized layout has either better or competitive property than the two. It is clear that our optimization framework can not only accurately predict the simulated evacuation time but also find optimal layouts that are either competitive or even better evacuation time performance without resorting to exhaustive simulations.

6. Conclusion

In this paper, we proposed (a) a novel neural network-based framework to learn the relationship between crowd movement and a building design in evacuation scenarios, (b) an optimization algorithm that efficiently iterates a vast number of design solutions to generate crowd-aware environmental layout design, and conducted (c) a case study that involves optimizing a complex, real-world built environment (Metropolitan museum) to minimize evacuation times and reduce congestion.

We showed that our framework can predict crowd-related factors from given building layout. Moreover, the proposed framework can evaluate the precision of predictions. The test cases result indicate predictive models have comparative precision in given metrics evaluation tasks against the tradition approach - simulations. We also showed that our framework can be used for performing building layout optimization from random initialization. A batch of optimization result can be obtained within only a half minute, and outcome layouts are significantly better performed compared to the origin non-optimized layout. In the case study using real world data, we showed that our model can be applied to a practical layout, tested utilization of our model out of synthetic experiments.

Limitations. As noted in experimental result section, the proposed measure R_{rel} may have negatives values when the dimension of the input parameters is extremely small (less than 10 in our experiments). We believe that such small dimension may cause the issue, because it might have been possible that the training set happens to encompass all possible variations of the small parameter space, making the neural network possible to memorize all instances, yielding smaller empirical variance. Moreover, if our key assumption on independence of the metric variance over environment/crowd/agent-configuration parameter configuration breaks, then applicability of the proposed method is limited. In such situations, additional consideration may need to be taken into account.

Future Works. Our work demonstrates a novel way of developing computer assisted layout design tools. Based on existing result up to this work, we can still improve its effectiveness by creating new models which can find optimal layouts based on the basic design. The models can also make comprehensive predictions on various performance measures.

Several aspects can be considered to further generalize the framework. First, the model can consider dynamic structural changes affected by problems that induce the evacuation (e.g., fire). Optimized layout may show how robust the estimated measures are, as a function of the changing environmental structures. Second, factors related to neuro-cognitive aspects (preference, stress level) or human-made objects (signage, lighting conditions) that may affect the agent steering can be factored into the model. Third, one can consider ways to lift the Gaussianity assumption on the noise model. Fourth, we can integrate our prediction-based optimization pipeline into interactive design optimization tools, e.g., IDOME [59] and use our approach to study larger environments at urban scales.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Kaidong Hu: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Sejong Yoon:** Supervision, Formal analysis, Investigation, Resources, Writing - review & editing. **Vladimir Pavlovic:** Formal analysis, Writing - review & editing. **Petros Faloutsos:** Formal analysis, Writing - review & editing, Funding acquisition. **Mubbasir Kapadia:** Supervision, Project administration, Funding acquisition, Writing - review & editing.

Acknowledgments

Authors thank anonymous reviewers for their constructive comments that greatly improved the quality of the paper. We have carefully revised the paper to account for reviewer feedback.

This work was supported in part by funds from ORF-RE RE08-054 (ISSUM) and TCNJ SOSA 2017-2019 grant. Kapadia has been funded in part by NSF IIS-1703883, and NSF S&AS-1723869. The authors acknowledge use of the ELSA HPC cluster at TCNJ for conducting the research reported in this paper. This cluster is funded by the NSF under grant number OAC-1828163.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cag.2020.03.005

References

[1] Schaumann D, Breslav S, Goldstein R, Khan A, Kalay YE. Simulating use scenarios in hospitals using multi-agent narratives. Journal of Building Performance Simulation 2017;10(5-6):636–52.
[2] Michalek J, Papalambros P. Interactive design optimization of architectural layouts. Engineering optimization 2002;34(5):485–501.
[3] Michalek J, Choudhary R, Papalambros P. Architectural layout design optimization. Engineering optimization 2002;34(5):461–84.
[4] Yeh I-C. Architectural layout optimization using annealed neural network. Automation in Construction 2006;15(4):531–9.
[5] Baušys R, Pankrašovaite I. Optimization of architectural layout by the improved genetic algorithm. Journal of Civil Engineering and Management 2005;11(1):13–21.
[6] Choudhary R. A hierarchical optimization framework for simulation-based architectural design.; 2004.
[7] Caldas LG, Norford LK. A design optimization tool based on a genetic algorithm. Automation in construction 2002;11(2):173–84.
[8] Shi X, Yang W. Performance-driven architectural design and optimization technique from a perspective of architects. Automation in Construction 2013;32:125–35.
[9] Bassuet A, Rife D, Dellatorre L. Computational and optimization design in geometric acoustics. Building Acoustics 2014;21(1):75–85.
[10] Lin S-H E, Gerber DJ. Designing-in performance: A framework for evolutionary energy performance feedback in early stage design. Automation in Construction 2014;38:59–73.
[11] Nagy D, Lau D, Locke J, Stoddart J, Villaggi L, Wang R, et al. Project Discover: An Application of Generative Design for Architectural Space Planning. In: Proc. of the Symposium on Simulation for Architecture and Urban Design; 2017.
[12] Haworth B, Usman M, Berseth G, Khayatkhoei M, Kapadia M, Faloutsos P. Code: Crowd-optimized design of environments. Computer Animation and Virtual Worlds 2017;28(6):e1749.
[13] Helbing D, Molnar P. Social force model for pedestrian dynamics. Physical review 1995;51(5):4282.
[14] van den Berg J, Ming L, Manocha D. Reciprocal Velocity Obstacles for real-time multi-agent navigation. In: IEEE International Conference on Robotics and Automation; 2008. p. 1928–35.
[15] Treuille A, Cooper S, Popoviundefined Z. Continuum crowds. ACM Trans Graph 2006;25(3):1160–8. doi:10.1145/1141911.1142008.
[16] Narain R, Golas A, Curtis S, Lin MC. Aggregate dynamics for dense crowd simulation. ACM SIGGRAPH Asia 2009 Papers; 2009. doi:10.1145/1661412.1618468.
[17] Kapadia M, Pelechano N, Allbeck J, Badler N. Virtual crowds: steps toward behavioral realism, 7. Synthesis Lectures on Visual Computing; 2015.
[18] Reynolds CW. A distributed behavioral model. ACM SIGGRAPH 1987;21(4):25–34.
[19] Reynolds CW. Steering behaviors for autonomous characters. In: Game Developers Conference; 1999. p. 763–82.
[20] Pelechano N, Allbeck JM, Badler NI. Controlling individual agents in high–density crowd simulation. In: ACM SIGGRAPH/Eurographics SCA; 2007. p. 99–108.
[21] Guy SJ, Chhugani J, Kim C, Satish N, Lin M, Manocha D, et al. ClearPath: Highly Parallel Collision Avoidance for Multi-Agent Simulation. In: ACM SIGGRAPH/Eurographics SCA; 2009. p. 177–87.
[22] Kapadia M, Singh S, Hewlett W, Faloutsos P. Egocentric affordance fields in pedestrian steering. In: Symp. on Interactive 3D Graphics, SI3D; 2009. p. 215–23. doi:10.1145/1507149.1507185.
[23] Singh S, Kapadia M, Reinman G, Faloutsos P. Footstep navigation for dynamic crowds. CAVW 2011;22(2-3):151–8.
[24] Kallmann M, Kapadia M. Geometric and discrete path planning for interactive virtual worlds. In: ACM SIGGRAPH Courses; 2016. p. 12:1–12:29. doi:10.1145/2897826.2927310.
[25] Huang T, Kapadia M, Badler NI, Kallmann M. Path planning for coherent and persistent groups. In: IEEE International Conference on Robotics and Automation, ICRA; 2014. p. 1652–9.
[26] Ninomiya K, Kapadia M, Shoulson A, Garcia FM, Badler NI. Planning approaches to constraint-aware navigation in dynamic environments. Journal of Visualization and Computer Animation 2015;26(2):119–39.
[27] Kallmann M, Kapadia M. Navigation meshes and real-time dynamic planning for virtual worlds. In: ACM SIGGRAPH; 2014. p. 3:1–3:81.
[28] Kapadia M, Beacco A, Garcia FM, Reddy V, Pelechano N, Badler NI. Multi-domain real-time planning in dynamic environments. In: ACM SIGGRAPH / Eurographics SCA; 2013. p. 115–24.
[29] Garcia FM, Kapadia M, Badler NI. Gpu-based dynamic search on adaptive resolution grids. In: IEEE International Conference on Robotics and Automation, ICRA; 2014. p. 1631–8.
[30] Kapadia M, Garcia FM, Boatright CD, Badler NI. Dynamic search on the GPU. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems, IROS; 2013. p. 3332–7.
[31] Lerner A, Chrysanthou Y, Lischinski D. Crowds by example. CGF 2007;26(3):655–64.
[32] Lee KH, Choi MG, Hong Q, Lee J. Group behavior from video: a data-driven approach to crowd simulation. ACM SIGGRAPH/Eurographics SCA 2007;1:109–18.
[33] Torrens P, Li X, Griffin WA. Building agent-based walking models by machine-learning on diverse databases of space-time trajectory samples. Transactions in GIS 2011(15):67–94.
[34] Scovanner P, Tappen MF. Learning pedestrian dynamics from the real world. In: 2009 IEEE 12th International Conference on Computer Vision; 2009. p. 381–8. doi:10.1109/ICCV.2009.5459224.
[35] Metoyer RA, Hodgins JK. Reactive pedestrian path following from examples. CASA 2003;20:149–56.
[36] Charalambous P, Karamouzas I, Guy SJ, Chrysanthou Y. A data-driven framework for visual crowd analysis. In: Computer Graphics Forum, 33. Wiley Online Library; 2014. p. 41–50.
[37] Kim S, Bera A, Manocha D. Interactive crowd content generation and analysis using trajectory-level behavior learning. In: Multimedia (ISM), 2015 IEEE International Symposium on. IEEE; 2015. p. 21–6.
[38] Bera A, Kim S, Manocha D. Interactive crowd-behavior learning for surveillance and training. IEEE Computer Graphics and Applications 2016;36(6):37–45. doi:10.1109/MCG.2016.113.
[39] Xu M, Li C, Lv P, Chen W, Manocha D, Deng Z, et al. Crowd simulation model integrating "physiology-psychology-physics" factors. CoRR 2018;abs/1801.00216.
[40] Motsch S, Moussaid M, Guillot E, Moreau M, Pettré J, Theraulaz G, et al. Modeling crowd dynamics through coarse-grained data analysis. Mathematical Biosciences and Engineering 2018;15:1271.
[41] Wolinski D, J Guy S, Olivier A-H, Lin M, Manocha D, Pettré J. Parameter estimation and comparative evaluation of crowd simulations. Comput Graph Forum 2014;33(2):303–12.
[42] Wang H, Ondřej J, O'Sullivan C. Trending paths: A new semantic-level metric for comparing simulated and real crowd data. IEEE Transactions on Visualization and Computer Graphics 2017;23(5):1454–64.
[43] Karamouzas I, Sohre N, Hu R, Guy SJ. Crowd Space: A Predictive Crowd Analysis Technique. ACM Trans Graph 2018;37(6).
[44] Snape J, Guy SJ, van den Berg J, Lin MC, Manocha D. Reciprocal collision avoidance and multi-agent navigation for video games. Multiagent Pathfinding, Papers from the 2012 AAAI Workshop, MAPF@AAAI 2012, Toronto, Ontario, Canada, July 22, 2012. AAAI Workshops, WS-12-10. AAAI Press; 2012.
[45] Block P, Knippers J, Mitra NJ, Wang W. Advances in Architectural Geometry 2014. Springer; 2015.
[46] Pottmann H, Eigensatz M, Vaxman A, Wallner J. Architectural geometry. Computers & graphics 2015;47:145–64.
[47] Peng C-H, Yang Y-L, Bao F, Fink D, Yan D-M, Wonka P, et al. Computational network design from functional specifications. ACM Transactions on Graphics (TOG) 2016;35(4):131.
[48] Cassol VJ, Smania Testa E, Rosito Jung C, Usman M, Faloutsos P, Berseth G, et al. Evaluating and optimizing evacuation plans for crowd egress. IEEE Computer Graphics and Applications 2017;37(4):60–71. doi:10.1109/MCG.2017.3271454.
[49] Galle P. An algorithm for exhaustive generation of building floor plans. Communications of the ACM 1981;24(12):813–25.

[50] Yi H, Yi YK. Performance based architectural design optimization: Automated 3d space layout using simulated annealing. In: Build. Simul. Conf; 2014. p. 292–9.
[51] Liu H, Yang Y-L, Alhalawani S, Mitra NJ. Constraint-aware interior layout exploration for pre-cast concrete-based buildings. The Visual Computer 2013;29(6-8):663–73.
[52] Merrell P, Schkufza E, Koltun V. Computer-generated residential building layouts. In: ACM Transactions on Graphics (TOG), 29. ACM; 2010. p. 181.
[53] Arvin SA, House DH. Modeling architectural design objectives in physically based space planning. Automation in Construction 2002;11(2):213–25.
[54] Yu L-F, Yeung S-K, Tang C-K, Terzopoulos D, Chan TF, Osher SJ. Make it home: Automatic optimization of furniture arrangement. ACM SIGGRAPH 2011 Papers; 2011. doi:10.1145/1964921.1964981.
[55] Feng T, Yu L-F, Yeung S-K, Yin K, Zhou K. Crowd-driven mid-scale layout design.. ACM Trans Graph 2016;35(4). 132–1
[56] Shukla PK. Genetically optimized architectural designs for control of pedestrian crowds. In: Australian Conference on Artificial Life. Springer; 2009. p. 22–31.
[57] Mathew CDT, Knob PR, Musse SR, Aliaga DG. Urban walkability design using virtual population simulation. Computer Graphics Forum 2019;38(1):455–69. doi:10.1111/cgf.13585.
[58] Testa E, Barros RC, Musse SR. CrowdEst: a method for estimating (and not simulating) crowd evacuation parameters in generic environments. The Visual Computer 2019;35(6):1119–30.
[59] Berseth G, Haworth B, Usman M, Schaumann D, Khayatkhoei M, Kapadia MT, et al. Interactive Architectural Design with Diverse Solution Exploration. IEEE Transactions on Visualization and Computer Graphics 2019. doi:10.1109/TVCG.2019.2938961.
[60] Liu W, Hu K, Yoon S, Pavlovic V, Faloutsos P, Kapadia M. Characterizing the relationship between environment layout and crowd movement using machine learning. In: Proceedings of the Tenth International Conference on Motion in Games. ACM; 2017. p. 2.
[61] Guy SJ, Chhugani J, Curtis S, Dubey P, Lin M, Manocha D. Pledestrians: A least-effort approach to crowd simulation. In: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. Goslar, DEU: Eurographics Association; 2010. p. 119–28.
[62] Whittle MW. Gait analysis: an introduction. Butterworth-Heinemann; 2007.
[63] Goodfellow IJ, Warde-Farley D, Mirza M, Courville AC, Bengio Y. Maxout networks. In: Proceedings of the 30th International Conference on Machine Learning, ICML; 2013.
[64] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323(6088):533–6.
[65] Singh S, Kapadia M, Faloutsos P, Reinman G. An open framework for developing, evaluating, and sharing steering algorithms. In: Proceedings of 2nd Intl. Workshop on Motion in Games MIG, 5884; 2009. p. 158–69.
[66] Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., et al. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org; URL https://www.tensorflow.org/.
[67] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Bengio Y, LeCun Y, editors. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings; 2015 http://arxiv.org/abs/1412.6980.
[68] 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. Springer; 2009. p. 158–69.
[69] Fukunaga K, Hostetler L. The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Transactions on Information Theory 1975;21(1):32–40.