SUPPLEMENTARY MATERIAL

This supplementary material is organized as follows. First, we provide the algorithmic reconstruction procedure of our proposed MSMC-Net. Next, we describe the details of datasets used for model training and testing. Then, we provide the implementation details for our method and all the baseline methods. Lastly, we describe the compared metrics for the crowd risk quantification.

A. Summary of reconstruction procedure

The reconstruction procedure of our MSMC-Net is shown in Algorithm 1.

Algorithm 1 Reconstruction procedure of our MSMC-Net

Input: MSMC graphs $\{G^s\}_{s=1}^{S}$

for $s = 1 \rightarrow S$ **do**
  $\mathbf{z}^s = \mathrm{GCN}^s\left(G^s\right)$ # GCN encoding
  $\tilde{\mathbf{z}}^s = \mathrm{Unpooling}^s\left(\mathbf{z}^s\right)$ # reshape into a unified scale
  $\{\tilde{\alpha}^s\}_{s=1}^{S}$, $\{\tilde{\beta}^s\}_{s=1}^{S}$ = $\{\tilde{\mathbf{z}}^s\}_{s=1}^{S}$
  Obtain multi-scale attention map $[a_{xy}^s]$ based on Eq. 5
  Obtain spatio-temporal attention map $[b_{xy}^s]$ based on Eq. 9-10
end for
Obtain normalized multi-scale attention maps $\mathbb{A}$ based on Eq. 6-7
Obtain normalized spatio-temporal attention maps $\mathbb{B}$ based on Eq. 11-13
Obtain multi-scale fusion vector $\mathbf{z}_{\mathrm{msFus}}$ using Eq. 8
Obtain spatio-temporal fusion vector set $\{\mathbf{z}_{\mathrm{stFus}}^s\}_{s=1}^{S}$ using Eq. 14
for $s = 1 \rightarrow S$ **do**
  $\tilde{\mathbf{A}}^s = \mathrm{Pooling}^s\left(\mathbf{A}^s\right)$ # reshape into original scales
  $\tilde{\mathbf{B}}^s = \mathrm{Pooling}^s\left(\mathbf{B}^s\right)$ # reshape into original scales
  $\tilde{\mathbf{z}}_{\mathrm{msFus}}^s \oplus \tilde{\mathbf{z}}_{\mathrm{stFus}}^s = \mathrm{Pooling}^s\left(\mathbf{z}_{\mathrm{msFus}} \oplus \mathbf{z}_{\mathrm{stFus}}^s\right)$
  $\hat{\mathbf{E}}_{\mathrm{Fus}}^s = \mathrm{Decode}\left(\tilde{\mathbf{z}}_{\mathrm{msFus}}^s \oplus \tilde{\mathbf{z}}_{\mathrm{stFus}}^s \oplus \tilde{\mathbf{z}}^s\right)$ # part (b) reconstruction
  $\hat{\mathbf{E}}_{\mathrm{Aux}}^s = \mathrm{Decode}\left(\mathbf{z}^s\right)$ # part (c) reconstruction
end for
return $\{\hat{\mathbf{E}}_{\mathrm{Fus}}^s\}_{s=1}^{S}, \{\hat{\mathbf{E}}_{\mathrm{Aux}}^s\}_{s=1}^{S}, \{\tilde{\mathbf{A}}^s\}_{s=1}^{S}$ and $\{\tilde{\mathbf{B}}^s\}_{s=1}^{S}$.

B. Datasets

UMN[1] is a video shot by CCTV at the University of Minnesota in 2006 (see Fig. 1a). The video contains three scenes. The video areas of the three scenes are 6 x 10 meters, 7 x 12 meters, and 10 x 9 meters, respectively. In the video, the walking pedestrians are considered normal, while the crowd escaping is abnormal. The original resolution of UMN is 320x240. As shown in Fig. 1a, a flashing text label appears whenever a frame is regarded as abnormal. To avoid the influence of these labels on the anomaly detection task, we trim off the text label area of all video frames and the trimmed video has a resolution of 320x213. The dataset is divided into non-overlapping training and testing parts. Our training set contains 4410 frames of normal behaviors and our testing set contains 3300 frames of both normal (1823 frames) and abnormal behaviors (1477 frames). For labeling the ground truth, we use the original label already provided in the dataset.

Normal crowd | Outdoor escape | Indoor escape

(a) Example screenshots in UMN dataset

Normal crowd | Indoor counterflow | Outdoor counterflow

(b) Example screenshots in Hajj dataset

Normal crowd | Counterflow | Crowd Turbulence

(c) Example screenshots in Love Parade dataset

Bidirectional flow | Reverse flow | Cross shaped flow

(d) Example screenshots in MOT20 dataset

Fig. 1. Example screenshots in our benchmark datasets. The arrows are added to indicate the optical flow in the frame. It is worth noting that during anomalies in the UMN dataset, there is a flashing text label in the upper left corner (see screenshots of outdoor and indoor escape), which is trimmed in our experiments to avoid its influence.

Hajj[2] comes from the surveillance video of Saudi Arabia's annual religious pilgrimage, including nine 45-second segments of dense crowds (see Fig. 1b). In these videos, abnormal behaviors include standing, sitting, sleeping, running, moving in the opposite or different direction of the crowd, and non-pedestrian movements, such as cars and wheelchairs. Among these abnormal behaviors, we select the crowd-level abnormal behavior, namely counter flow, for evaluation. The resolution of the videos is 720x576. The selected dataset is divided into non-overlapping training and testing parts. The training set contains 1500 frames without counter flow behavior and the testing set contains 1080 frames with both counter flow behavior (480 frames) and normal behavior (600 frames). For labeling the ground truth, we use the original label already provided in the dataset.

[1] http://mha.cs.umn.edu/Movies/Crowd-Activity-All.avi

[2] https://github.com/KAU-Smart-Crowd/Hajj_abnormal_behavior_detection

Love Parade[3] contains surveillance videos from 7 monitoring locations within 3 hours before the love parade accident in 2012 (see Fig. 1c). The total length of all videos is over 23 hours and the crowd density in the videos reaches 11 people per square meter. The videos contain various CABs, including counter flow and crowd turbulence. We chose the videos from two of the cameras (4 and 13) in which counter flow and turbulence occurred and selected the video frames that contain the anomalies and are before and after the occurrence of anomalies. The selected dataset is divided into non-overlapping training and testing parts. The training set contains 2767 frames of normal behaviors and the testing set contains 1830 frames in total, in which 810 frames contain anomalies. For labeling the ground truth of the counter flow, frames are labeled based on whether the groups of pedestrians move in a counter-direction from the majority of the crowd. To label the ground truth of the crowd turbulence, we refer to the post-disaster analysis of the Love Parade disaster and label the crowd turbulence events according to the provided timeline [1]. The first testing video containing the crowd turbulence is clipped from 14:50 to 15:10 of the Camera_13_1620 video, and the turbulence starts at 15:00. The second testing video containing the crowd turbulence is clipped from 13:50 to 14:11 of Camera_04_1440 video, and the turbulence starts at 14:04.

MOT20[4] contains 8 video sequences generated by monitoring of 5 different scenes, such as urban streets, indoor shopping malls, public transportation, etc., covering various challenging factors such as congestion, lighting changes, occlusion. These sequences contain different forms of movement, such as one-way flow, two-way flow, cross flow (see Fig. 1d). The average number of people in each video frame is about 160. To verify the performance of our model on complex crowd movement behavior using the MOT20 dataset, we used these videos as training data for the model, and then evaluated the model using test sets from other datasets.

C. Baseline Details

FramePred [2]: The algorithm adopts U-Net as a generator to predict the next frame. It adopts the constraints of appearance (intensity loss and gradient loss) to generate a high-quality image and motion (optical flow loss) and detects anomaly by predicting future frame. Experimental results show that the AUC of the algorithm on CUHK Avenue, UCSD ped1, UCSD ped2, and ShanghaiTech datasets are 0.851, 0.831, 0.954, and 0.728, respectively.

AMC [3]: This algorithm is a classical anomaly detection work based on reconstruction. Anomaly detection is realized by combining Conv-AE and CNN of U-Net to reconstruct the frame, then comparing the difference between the reconstructed frame and the actual frame based on the patch scheme. Experimental results show that the AUC of the algorithm on CUHK Avenue and UCSD ped2 datasets are 0.869 and 0.962, respectively.

MNAD (recons/pred) [4]: It exploits multiple prototypes to consider the various behaviors of normal data and uses a memory module to record the prototypical patterns of the items in the memory. At the same time, two modes of anomaly detection, prediction, and reconstruction, are proposed. Based on the memory module, novel feature compactness and separability loss are proposed to train memory, which ensures the diversity and discrimination of memory items. Experiments show that the AUC values of the reconstruction-based scheme on UCSD ped2, CUHK Avenue, and ShanghaiTech datasets are 0.902, 0.828, and 0.698, respectively; The AUC of the prediction-based scheme is 0.970, 0.885 and 0.705, respectively.

AMMC-Net [5]: Inspired by the rules of recognizing abnormal frames from multi-modal signals, the appearance motion memory consistency network (AMMC-Net) is proposed. The method fully uses the prior knowledge of appearance and motion signals and captures the corresponding relationship between them in the high-level feature space. Then, combined with multi-view features, a more fundamental and robust feature representation of conventional events is obtained, which can significantly increase the gap between abnormal and normal events. Experimental results show that the AUC of the algorithm on UCSD ped2, CUHK Avenue, and Shanghai tech datasets are 0.966, 0.866, and 0.737, respectively.

TABLE I
PARAMETER SETTINGS

Parameters	Value (UMN, Hajj, LP)
self-attention dimension (q, k, v)	32, 16, 32
graph embedding dimension(z)	32, 16, 32
encoder network size	(32, 16), (16,8), (32, 16)
learning rate	$3e^{-4}$, $1e^{-4}$, $3e^{-4}$
β_1 of Adam optimizer	0.9
β_2 of Adam optimizer	0.999
velocity direction categories (D)	8
length of sliding window (m)	20
sliding window step (τ)	1
multi-scale number (S)	3
fusion loss weight (λ_{Fus})	1
soft sharing loss weight (λ_{Sof})	1
auxiliary loss weight (λ_{Aux})	1
moving average weight (λ_{Mov})	0.2
random seed	42

D. Implementation Details

All our experiments are conducted on a desktop PC with Intel(R) Xeon(R) CPU E5-1650 V4@3.60GHz and 32 GB RAM operated with a 64-bit Windows 10 system.

MSMC-Net (Ours) We use a grid search to set hyper-parameters on the test split, and our hyper-parameters tuned for each dataset can be found in Table I. The decay rates (β_1,β_2) of the Adam optimizer [6] for network parameter tuning are set to 0.9 and 0.999, as suggested in the original paper. The number of velocity direction categories (D) is set to the typical number of 8, representing top, bottom, left, right, top left, bottom left, etc. Considering a moderate range of scales, our

[3]https://loveparade2010doku.wordpress.com/2010/08/30/lopavent-\\verof fentlicht-originalvideos-von-7-der-16-uberwachungskameras-der-loveparade-2010/

[4]https://motchallenge.net/data/MOT20/

multi-scale framework is tested using the scale range of {1x, 2x, 4x} scales. To train all parts in a balanced way, we set an equal weight for the fusion loss, auxiliary loss, and soft sharing loss. Considering crowd behaviors tend to last for a period, the weight of the moving average (λ_{Mov}) is set to 0.2. The region size $w \times h$ of the baseline $1x$ scale for each dataset is determined by the average pixel size of pedestrians in the dataset's videos, as described in the Methodology section. The random seed is set to 42 when initializing the network's parameters using the random seed function provided in Python. Farneback optical flow[5] is obtained using OpenCV[6]. All codes are implemented in Python, and our MSMC-Net is trained using PyTorch[7].

FramePred [2] The code is obtained from the link[8] and a grid search is used to set the hyper-parameters on the same test split as used for tuning our method. The learning rates are set to ($2e^{-4}$, $2e^{-5}$), ($1e^{-4}$, $1e^{-5}$) and ($2e^{-4}$, $2e^{-5}$) on the three datasets respectively. As stated in the original paper, λ_{int}, λ_{gd}, λ_{op} and λ_{adv} vary from different datasets very slightly and they are set to 1.0, 1.0, 2.0 and 0.05, respectively. For a fair comparison, the sliding window size is set to 20, the same as in our method.

AMC [3] The code is obtained from the link[9] and a grid search is used to set the hyper-parameters on the same test split as used for tuning our method. The learning rates are set to ($1e^{-4}$, $1e^{-5}$), ($5e^{-5}$, $5e^{-6}$) and ($1e^{-4}$, $1e^{-5}$) on the three datasets, respectively.

MNAD [4] The code is obtained from the link[10] and a grid search is used to set the hyper-parameters on the same test split as used for tuning our method. For the reconstruction version of MNAD, the learning rates are set as $2e^{-5}$, $1e^{-5}$ and $2e^{-5}$, and λ is set to 0.9, 0.7 and 0.6 for the three datasets, respectively. For the prediction version of MNAD, the learning rates are set as $2e^{-4}$, $1e^{-4}$ and $2e^{-4}$, and λ is set to 0.9, 0.5 and 0.5 for the three datasets, respectively. As for the height H and width W of the query feature map, and the number of feature channels C and memory items M hardly vary among different datasets. They are set to 32, 32, 512 and 10 as the same as in the original paper. The sliding window size is set to 20, the same as in our method for a fair comparison.

AMMC-Net [5] The code is obtained from the link[11] and a grid search is used to set the hyper-parameters on the same test split as used for tuning our method. The learning rates are set as $1e^{-3}$, $3e^{-4}$ and $1e^{-3}$ for the three datasets. According to the ablation study in the original paper, the memory item numbers K, memory size N and memory dimension D are set to 2, 64 and 256. For a fair comparison, the sliding window size is also set to 20, the same as in our method.

[5]Farnebäck, G. 2003. Two-frame motion estimation based on polynomial expansion. In SCIA, 363–370.

[6]https://docs.opencv.org/2.4/modules/video/doc/motion_analysis_and_object_tracking.html?highlight=calcopticalflowfarneback

[7]https://pytorch.org/

[8]https://github.com/StevenLiuWen/ano_pred_cvpr2018

[9]https://github.com/nguyetn89/Anomaly_detection_ICCV2019

[10]https://github.com/cvlabyonsei/MNAD

[11]https://github.com/NjuHaoZhang/AMMCNet_AAAI2021

E. Crowd Risk Metrics for Comparison

Particle entropy [7] measures the degree of disorder in the distribution of crowds by treating them as particles based on their position distribution and velocity direction distribution. The calculation of particle entropy depends on the location distribution information of a given crowd. The calculation of particle entropy first divides the video frame into grids and then counts the distribution of crowd particles in both horizontal and vertical directions. The particle entropy in the horizontal and vertical directions is then calculated separately and then combined to obtain the particle entropy of an entire frame. The effectiveness of particle entropy was validated on the Crowd PETS09 dataset, and the effectiveness of particle entropy in evaluating the degree of disorder in crowds with abnormal escape situations has been validated on the UMN dataset.

Collective consistency [8] reflects the degree of consistency among individuals within a group by measuring the distribution of individual velocity directions. The calculation of collective consistency depends on the detailed trajectory information of crowd movement. Collective consistency first calculates the velocity correlation between each individual and the neighboring crowd based on *K*-nearest neighbors and then calculates the crowd similarity along the path based on crowd trajectory information. Finally, average normalization is performed on all the paths in a frame to form collective consistency. Experimental analysis has been conducted to verify the effectiveness of collective consistency on crowd collective walking and lane formation.

Crowd pressure [9] measures the degree of crowding and pushing during crowd turbulence by calculating the variance of speed and combining it with crowd density. The calculation of crowd pressure depends on the location information of the crowd obtained by crowd tracking method, as well as the density and speed of the crowd measured based on tracking information. The calculation method for crowd pressure is the variance of velocity per unit area multiplied by crowd density. Experimental analysis has been conducted in Love Parade dataset demonstrating the ability of crowd pressure to quantify turbulence anomalies.

REFERENCES

[1] D. Helbing and P. Mukerji, "Crowd disasters as systemic failures: analysis of the love parade disaster," *EPJ Data Sci.*, vol. 1, no. 1, pp. 1–40, 2012.

[2] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection - a new baseline," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6536–6545.

[3] T.-N. Nguyen and J. Meunier, "Anomaly detection in video sequence with appearance-motion correspondence," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 1273–1283.

[4] H. Park, J. Noh, and B. Ham, "Learning memory-guided normality for anomaly detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14 372–14 381.

[5] R. Cai, H. Zhang, W. Liu, S. Gao, and Z. Hao, "Appearance-motion memory consistency network for video anomaly detection," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 938–946.
[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.
[7] X. Gu, J. Cui, and Q. Zhu, "Abnormal crowd behavior detection by using the particle entropy," *Optik*, vol. 125, no. 14, pp. 3428–3433, 2014.
[8] B. Zhou, X. Tang, and X. Wang, "Measuring crowd collectiveness," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 3049–3056.
[9] A. Johansson, D. Helbing, H. Z. Al-Abideen, and S. Al-Bosta, "From crowd dynamics to crowd safety: a video-based analysis," *Adv. Complex. Syst.*, vol. 11, no. 04, pp. 497–527, 2008.