CrowdDiff: Multi-hypothesis Crowd Density Estimation using Diffusion Models

Yasiru Ranasinghe, Nithin Gopalakrishnan Nair, Wele Gedara Chaminda Bandara, and Vishal M. Patel
Johns Hopkins University, Baltimore, USA
{dranasi1, ngopala2, wbandar1, vpatel36}@jhu.edu

Abstract

Crowd counting is a fundamental problem in crowd analysis which is typically accomplished by estimating a crowd density map and summing over the density values. However, this approach suffers from background noise accumulation and loss of density due to the use of broad Gaussian kernels to create the ground truth density maps. This issue can be overcome by narrowing the Gaussian kernel. However, existing approaches perform poorly when trained with ground truth density maps with broad kernels. To deal with this limitation, we propose using conditional diffusion models to predict density maps, as diffusion models show high fidelity to training data during generation. With that, we present CrowdDiff that generates the crowd density map as a reverse diffusion process. Furthermore, as the intermediate time steps of the diffusion process are noisy, we incorporate a regression branch for direct crowd estimation only during training to improve the feature learning. In addition, owing to the stochastic nature of the diffusion model, we introduce producing multiple density maps to improve the counting performance contrary to the existing crowd counting pipelines. We conduct extensive experiments on publicly available datasets to validate the effectiveness of our method. CrowdDiff outperforms existing state-of-the-art crowd counting methods on several public crowd analysis benchmarks with significant improvements. CrowdDiff project is available at:
https://dylran.github.io/crowddiff.github.io.

1. Introduction

Crowd counting has been a fundamental problem in surveillance, public safety, and crowd control. Various methods have been proposed in the literature, including methods that directly predict the count [21, 51, 58] or use a surrogate task such as density estimation [13, 16, 46, 47, 49, 50], object detection [30, 39], or point localization [9, 22, 45, 56].

While density-based methods sum the estimated pixel density values for counting [13], localization-based methods count proposals with confidence scores higher than a threshold [45]. As a result, density-based methods are more susceptible to introducing background noise into the final

(a) Ground truth: 1155 (4) (b) Ours: 1142 (4)

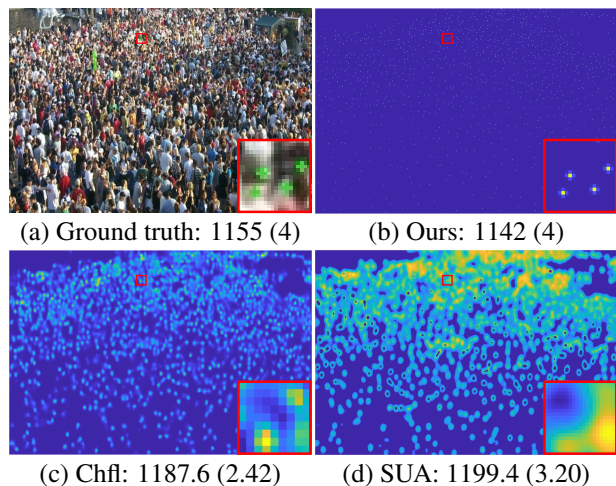(c) Chfl: 1187.6 (2.42) (d) SUA: 1199.4 (3.20)

Figure 1. Predicted density results for (a) a dense crowd from (b) our method, (c) Chfl [41], and (d) SUA [34]. The count of the enlarged crop is given in brackets.

count compared to localization-based methods [48]. Furthermore, density estimation methods are affected by variations in crowd density distributions that arise due to different congestion levels of the crowd [3]. This could result in a loss of accuracy in the density estimation. In contrast, recent localization-based methods with point queries do not have the issue of background noise accumulation [22], like in density-based methods [41], as there is no interference between neighboring point proposals. However, localization-based methods require crowd density heuristics for proposal setting [45], which is not required by density-based methods. Thus, if the premise of point supervision is translated into density-based methods, it is possible to circumvent the requirement for crowd density heuristics, and the flaws of conventional density-based methods, and a narrow density kernel can be used to achieve this. However, Xu et al. [56] demonstrated that using a narrow kernel with density regression methods is ineffective.

Alternatively, it is feasible to use a generative model to predict the density map of a given crowd image that

would learn the distribution of the values in the density map. Though Generative Adversarial Network (GAN) based architectures have been used for density map prediction [8, 40, 57], these methods still rely upon broad kernel sizes and overlook the benefits of point supervision. Since the model learns the distribution of the density pixel values, it is advantageous to maintain the sample space of the density pixel values, and employing a broad kernel will only discourage it. Furthermore, the use of both point supervision and crowd density prediction with generative models has not been thoroughly investigated before. Also, the aforementioned GAN-based approaches restrict to a single crowd density map realization similar to the regression-based methods and jettison the stochastic nature of generative models to produce multiple density map realizations, which could improve the counting performance.

We propose using denoising diffusion probabilistic models (diffusion models), [12, 36] to generate crowd density maps for a given image. Though diffusion models have been applied to segmentation [2, 10], super-resolution [18], object detection [4], etc., to the best of our knowledge, neither crowd counting nor density map generation has been studied with diffusion models. Furthermore, with the narrow kernel, we minimize the interference between adjacent densities, which helps to maintain the bounds and the distribution of density pixel values. This, in turn, simplifies the distribution learning for the diffusion model and improves density prediction as illustrated in Fig. 1, where the proposed method has reproduced the narrow kernel even in a dense region, while the other two recent methods failed.

Additionally, to eschew the probable loss of density with the density-based crowd counting methods, we count the number of blobs observed in the predicted density map by thresholding pixel density values. Consequently, we eliminate the effect of background noise as there is no requirement to sum over the density pixel values. Then, we introduce the crowd map fusion mechanism, combining multiple dot maps constructed after thresholding to improve the counting performance. This is only possible with generative models due to their stochastic nature. In addition, inspired by [7] on joint learning with diffusion models, we introduce an auxiliary regression branch only during training, which estimates the count based on encoder-decoder features from the denoising network to improve feature learning.

In summary, our contributions are:

- **We formulate crowd density map generation as a denoising diffusion process.** CrowdDiff is the first study to perform crowd counting with diffusion models.
- **We promote using a narrow Gaussian kernel** to ease the learning process and facilitate the high-quality density map generation with more fidelity to the ground truth.
- **We propose a mechanism to consolidate multiple crowd density realizations** to improve performance uti-

lizing the stochastic nature of diffusion models.
- **We show that the proposed method surpasses the state-of-the-art performance** on public datasets.

2. Background and Related Work

2.1. Crowd counting

Localization-based methods perform counting by predicting the locations of heads, and generally, they involve predicting a bounding box [20, 30, 39, 60] for each head. The literature has also proposed localization by points [20] or blobs [28]. Recently, to remove the necessity of post-processing, such as non-maximum suppression, point localization [22, 45] was introduced to crowd counting.

Density-based methods [3, 19, 26, 27, 29, 35, 55] attempt to produce a density map for a given crowd image. However, density-based methods suffer from background noise and loss of density [31, 33, 37] in congested regions due to broad kernels. But, using a narrow Gaussian kernel to generate ground truths is ineffective according to [56] with regression networks. Hence, we treat the prediction of the density map as a generative task.

2.2. Diffusion models for crowd density generation

Diffusion models [43] are defined based on a Markov chain with a forward and a reverse process. In the forward process, noise is gradually added to data; and is denoised in the reverse process. The forward process is formulated as,

$$q(\mathbf{x_t}|\mathbf{x_{t-1}}) = \mathcal{N}(\mathbf{x_t}|\sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t \mathbf{I}),$$

where the sample datum $\mathbf{x_0}$ is gradually transformed to a noisy sample $\mathbf{x_t}$ for $t \in \{1, \ldots, T\}$ by adding Gaussian noise according to a noise variance schedule $\beta_1, \ldots, \beta_T$. Here, $\mathbf{I}$ is the identity matrix. Nonetheless, $\mathbf{x_t}$ can be computed using $\mathbf{x_0}$ and a noise vector $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and with the forward transformation,

$$\mathbf{x_t} = \sqrt{\bar{\alpha}_t}\mathbf{x_0} + \sqrt{(1-\bar{\alpha}_t)}\epsilon,$$

where $\bar{\alpha}_t \coloneqq \prod_{\tau=1}^{t} \alpha_\tau = \prod_{\tau=1}^{t}(1-\beta_\tau)$ and β_τ.

In this work, we aim to perform crowd density map generation via the diffusion model. Hence, our data samples will be crowd density maps $\mathbf{x_0} \in \mathbb{R}^{H \times W}$, where H and W are the height and width dimensions. However, in lieu of training a neural network to predict $\mathbf{x_0}$ from $\mathbf{x_t}$ for various time steps, we predict the amount of noise ($\hat{\epsilon}$) in $\mathbf{x_t}$ at each time step conditioned on the crowd image ($\mathbf{y}$) and apply the reverse diffusion process to obtain $\mathbf{x_0}$ ultimately.

To that end, to train the denoising diffusion network, we use the hybrid loss ($\mathcal{L}_{hybrid}$) function proposed in [36]. To promote learning coarse features at lower SNR stages, we adopt the weighting scheme [6] defined as,

$$\lambda_t = \frac{(1-\beta_t)(1-\bar{\alpha}_t)/\beta_t}{(k+\text{SNR}(t))^\gamma}, \text{where } \text{SNR}(t) = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t} \quad (1)$$

with k and γ as hyperparameters. Hence, the final loss over which the denoising network is optimized is as follows,

$$\mathcal{L}_{hybrid} = \mathbf{E}_{\mathbf{x_0}, \mathbf{y}, \epsilon} \left[\lambda_t \|\hat{\epsilon}_{(\mathbf{x_0}, \mathbf{y}, t)} - \epsilon\|_2^2 \right] + \lambda_{vlb} \mathcal{L}_{vlb},$$
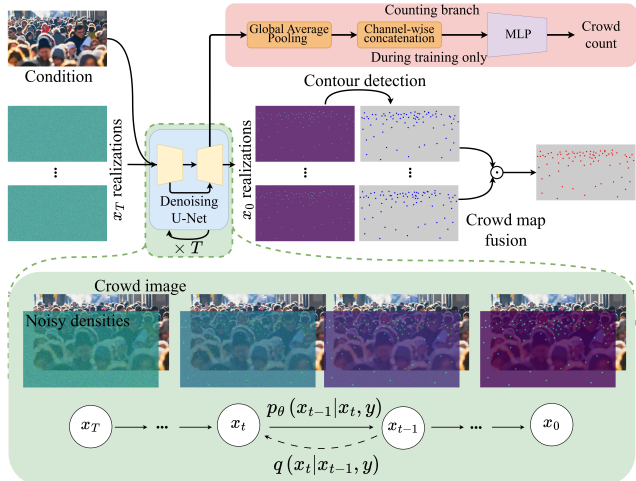
Figure 2. Overall crowd counting pipeline. The crowd density maps are generated from the denoising diffusion process for a crowd image. Next, thresholding is performed on the resulting crowd density realizations to create crowd maps. The crowd maps are then fused into a single crowd map. The counting branch is trained in parallel using the encoder-decoder features of the denoising U-Net and discarded during inference.

Figure 3. Change in the pixel values of the Gaussian kernel (red stems) and the resulting density map (blue stems) for a crowd image with a $3,547$ crowd count. The kernel size and variance increase from left to right.

where $\mathcal{L}_{vlb}$ is the original variational lower bound defined in [36] and λ_{vlb} is its weighting factor.

3. CrowdDiff

In this section, we first review the motivation for selecting an appropriate kernel size. We present the joint learning of counting as an auxiliary task to improve the density map generation performance. Finally, we introduce a method to combine different realizations for density maps to improve crowd counting performance. The overall crowd counting pipeline is illustrated in Fig. 2.

3.1. Narrow kernel
The diffusion process requires a density map to learn the conditional distribution of crowd density. The crowd density map can be acquired by convolving point information with a pre-defined Gaussian kernel. For that, selecting a proper kernel size and variance is important as it governs the distribution of the pixel values of the crowd density maps.

As demonstrated in Fig. 3, the divergence between the distribution of the Gaussian kernel (values) and the resulting density map increases as the kernel size and variance increase, especially for the congested scene. This might not be the case for sparse crowd scenes, as there is minimal or no interference between density kernels. However, this implies that the density pixel value distribution is highly image-dependent, hindering the crowd densities' learning. This can be eschewed by narrowing the distribution of the Gaussian kernel as illustrated in Fig. 3. This also helps the
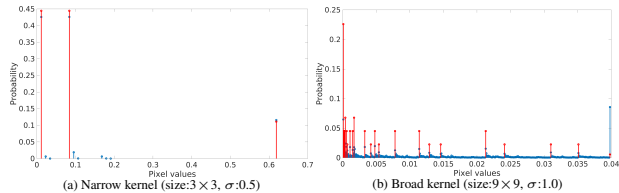
denoising network to maintain the pixel values within a pre-defined range. The difference between the probability mass of a broad Gaussian kernel and the resulting density map is significant. This can lead to the clipping of many pixel values, resulting in a loss of information in congested scenes.

The aforementioned issue can be solved by a narrow kernel. A narrow kernel provides an alternative path to crowd counting without summing over the density map values. As shown in Fig. 1, the crowd count can be obtained by simply counting the observable kernels. For that, we perform thresholding on the density maps and obtain the location of each kernel. Then, the crowd count is computed as the total number of locations. This provides the means to avoid background noise in the generated density maps and to obtain the crowd count by detecting these narrow kernels in the crowd density maps. Unlike the local maxima detection strategy proposed in [23] to detect head locations from a crowd density map, our method is not dependent on any hyperparameter tuning for detection.

3.2. Joint learning of counting
Directly regressing the crowd count from image features is a difficult task [21] compared to counting with a surrogate task. To perform the direct computation of the crowd count, we consider the intermediate features of the encoder-decoder of the denoising U-Net. Let's denote the set of intermediate features from the denoising network for a particular timestep t as $\mathcal{Z}_t = \{\mathbf{z}_t^1, \mathbf{z}_t^2, \ldots, \mathbf{z}_t^d\}$, where $\mathbf{z}_t^*$ is the representation vector at the corresponding feature level of the decoder. Since the spatial dimensions of the intermediate representations at different depth levels are incompatible, global average pooling is performed on each $\mathbf{z}_t^*$, which are then concatenated to construct a single feature vector $\mathbf{z}_t$. This is then passed through the regression network to estimate the crowd count at various noise levels.

However, for a sampled pair $(\mathbf{x_0}, \mathbf{y})$, only the density map $\mathbf{x_0}$ is diffused with noise according to a noise schedule. Hence, the noise level in the set of intermediate features $\mathcal{Z}_t$ will vary with the timestep, and SNR will be lower in the later stages of the diffusion process than in the earlier stages. Hence, we utilize the weighting scheme discussed in
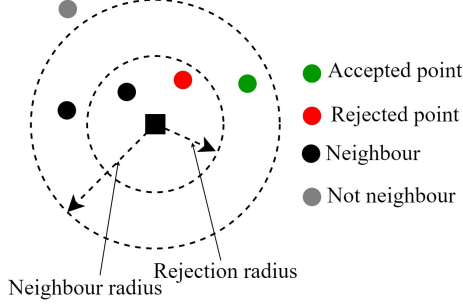
Figure 4. Crowd map fusion criterion. The rejection radius is computed from the neighbors (black) inside the neighbor radius. New points (colored) that fall inside the rejection radius are removed (red), and the rest (green) are combined into the compound map.

Sec. 2.2 during the training of the count regression network. We utilize $\mathcal{L}_1$ loss as follows,

$$\mathcal{L}_1^t = \lambda_t \|\bar{c}_t - c\|_1$$

to measure the difference between the prediction ($\bar{c}_t$) and the ground truth (c) for a given time step t and a given sampled pair, where λ_t is the same weighting factor used in Eq. (1). As the training loss for the denoising model is the Monte-Carlo approximation of the sum over all timesteps, the training loss can be written as

$$\mathcal{L}_{count} = \mathbb{E}_{\mathbf{x_0}, \mathbf{y}, t} \left[\lambda_t \|\bar{c}_t - c\|_1 \right].$$

The overall training includes optimization over the parameters of the denoising network and the regression branch. Hence, the overall training objective is as follows,

$$\mathcal{L}_{overall} = \mathcal{L}_{hybrid} + \lambda_{count} \mathcal{L}_{count},$$

where λ_{count} is a weightage on the counting task.

3.3. Stochastic crowd map fusion

The stochastic nature of the diffusion models could generate different realizations of the crowd density map for the same crowd image. Therefore, the counting performance with diffusion models can be improved with multiple realizations contrary to the traditional crowd counting methods as evidenced by other tasks based on diffusion models such as segmentation [10] and detection [4]. However, rather than averaging individual counts from different realizations, they could be combined to compute a more improved count because individual realizations could infer crowd densities that were not present in other realizations.

To combine different realizations for the density maps, only the new information should be transferred to the compound density map. For that, we first compute the locations of the density kernels by density thresholding. Once these locations are found, a dot map is constructed for each density map, referred to as the 'crowd map.' Then, we consider the dissimilarity between the crowd maps from different realizations, and to measure that, we consider the structural similarity index measure (SSIM) [53]. We assign a similar-

ity score measured as the cumulative SSIM with the remaining crowd map realizations for each crowd map. Then, the maps will be arranged in the ascending order of the SSIM before combining. Further, we don't require the ground truth locations to combine different realizations; they are combined depending on the similarity of the crowd maps.

Let's consider four crowd maps. For a given crowd map (source map), we'll measure the SSIM with each of the remaining three crowd maps, and the sum of those three SSIMs will be assigned as the similarity score of the source map. If the similarity score is the highest of a map, then it is the most similar to the remaining maps and likely to contain most of the point locations available in the remaining maps. Hence, the new points that can be added to and from the most similar map are minimal. Conversely, the crowd map with the least similarity score differs the most from the remaining maps; therefore, the new points that can be added to / from this map are maximal. Consequently, the best map to start the fusion process is the crowd map with the lowest similarity score. Likewise, we order the crowd maps in the ascending order of the similarity score to combine.

When fusing two crowd maps, it is necessary to reject repeating point locations. This is performed based on the locations of the new points compared to the points in the combined list. We take the crowd map first and the head locations from that realization as the reference. Next, we define a rejection radius for each head location as:

$$r_n = \beta \frac{\sum_{i=1}^{\tilde{k}} r_{ni}}{2\tilde{k}}$$

by considering the k-nearest neighbors within a fixed range. Here β is a scaling factor and $\tilde{k}$ is the total nearest neighbors within the range. Next, we remove the head locations of the next crowd map locations that fall within the rejection radii in the reference map as illustrated in Fig. 4, and the remaining locations are added to the reference map. This procedure is performed until all realizations are exhausted.

4. Experimental Details

CrowdDiff pipelines. During training, we create the ground truth density map with narrow kernels as described in Sec. 3.1. Next, we randomly sample a timestep t. Then, we sample a Gaussian noise according to the variance at t and add it to the ground truth map, resulting in the noisy map ($\mathbf{x_t}$). Then, we input the image and $\mathbf{x_t}$ to the denoising U-Net (network) and predict the noise added to the ground truth. Hence, based on the crowd image, the network is trained to predict the noise in $\mathbf{x_t}$. During inference, we sample a Gaussian noise from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ at time T, which is used as the initial noisy density map $\mathbf{x_T}$. Then, the network will estimate the noise present in $\mathbf{x_T}$, and by removing that noise, we produce the noisy density map ($\mathbf{x_{T-1}}$) at time $T - 1$. Likewise, we'll repeat the process where the noisy density map $\mathbf{x_{t-1}}$ at $t - 1$ is estimated from the noisy den-

sity map $\mathbf{x_t}$ at time t until we produce the density map $(\mathbf{x_0})$ for the image. Besides that, the counting branch output is discarded during inference.

Diffusion process uses 1,000 timesteps and DDIM sampling [44] during inference. We use a linear noise schedule with noise variance ranging from 1×10^{-3} to 0.02.

Hyperparameter values λ_{count} is set as 5×10^{-3} to match the range of the value for L_{hybrid}. The γ and k values are set as 0.5 and 1, respectively, to compute the SNR-based weighting factors. We adopt the original scaling factor of 1×10^{-3} for λ_{vlb} following [36]. For crowd map fusion, we set β equal to 0.85 and the maximum nearest neighbors to four. The radius for the neighbor search was restricted to 0.05 of the minimum of the image dimensions.

Training of the denoising network is initialized with the ImageNet pre-trained weights for the super-resolution [38] task except for the input and output layers. The network is trained for 2×10^5 iterations with a batch size of 8 for 256×256 images. We use an AdamW optimizer with a fixed learning rate 1×10^{-4} and a linear warm-up schedule over 5×10^3 training steps following [54].

Evaluations are performed on six public datasets: JHU-Crowd++[42], ShanghaiTech A[59], ShanghaiTech B[59], UCF-CC-50[14], UCF-QNRF[15], and NWPU-Crowd[52]. We use MAE and MSE as the performance metrics.

5. Results

5.1. Crowd counting performance

Quantitative results for crowd counting are presented in Tab. 1 for the proposed method with other existing methods. The proposed method achieves state-of-the-art crowd counting results on public crowd counting datasets, and two factors can explain the improvement. First, the proposed use of a narrow kernel has improved the counting results in dense regions by mitigating the loss of density values in contrast to conventional density-based methods. Second, we eliminate the effect of background noise on the crowd count, which scales with the image dimensions, by replacing density summation with thresholding followed by summation. The performance on JHU-Crowd++, UCF-QNRF, and NWPU-Crowd datasets explains the above effect of CrowdDiff as these datasets contain dense crowd scenes and large image dimensions. This was possible due to the capability to produce accurate density maps with better resemblance to ground truth maps with diffusion models. In Tab. 2, we provide the performance on the *test* set of the NWPU-Crowd dataset. In addition to the overall MAE, CrowdDiff has the best performance in negative samples or sparse crowds similar to detection or localization-based methods. This is due to density thresholding because of the ability to produce narrow kernels without intersections.

Qualitative results are presented in Fig. 1 and Fig. 5 for density map generation with diffusion models and crowd

GT: 99 Prediction: 101   GT: 2607 Prediction: 2566

Figure 5. Qualitative results for the proposed method with the ground truths. The prediction is produced after combining multiple realizations.

map generation with the proposed pipeline. As depicted in Fig. 1, the proposed method and the narrow kernel can accurately perform counting even in a dense region. In contrast, the other two methods have suffered from loss of density. Furthermore, our proposed pipeline has identified head locations accurately, which is impossible with existing density-based methods and without data heuristics, unlike localization-based methods.

5.2. Ablation study

Diffusion models are considered to have more fidelity to training data than GAN-based methods. From Fig. 6, one can see that diffusion models have produced high-quality density maps with more accurate crowd counts while ASCSP [40], a GAN-based method, has failed. Furthermore, without the ability to produce narrow kernels in the predicted density maps, GAN-based methods have to use density summation as the counting operation, bringing back noise accumulation and loss of density. This highlights the importance of using diffusion models for crowd density map generation with detection as the counting operation.

Stochastic crowd map generation is a key benefit of diffusion-based generative models. In Fig. 7, we provide qualitative results of two realizations for each crowd image. From Fig. 7, we can see that different realizations have information that is not present in other realizations. Furthermore, it is noteworthy that using a narrow kernel facilitates the ability to produce new knowledge that can be included in the final prediction. Otherwise, novel information captured by different realizations will be diluted by averaging the density maps had a larger kernel been used. Though this is a generative model, the proposed method has reassigned densities perfectly in certain instances, and for some cases, there is a slight shift in the location between realizations. This necessitates the need for the proposed crowd map fu-

Table 1. Comparison with state-of-the-art methods on the public crowd analysis benchmarks: JHU-Crowd++, ShanghaiTech, UCF, and NWPU-Crowd. The best results are shown in red. The second-best results are shown in blue.

Method	Venue	JHU-Crowd++		ShanghaiTech A		ShanghaiTech B		UCF-CC-50		UCF-QNRF		NWPU-Crowd	
		MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓
TopoCount [1]	AAAI'21	60.9	267.4	61.2	104.6	7.8	13.7	184.1	258.3	89.0	159.0	107.8	438.5
SUA [34]	ICCV'21	80.7	290.8	68.5	121.9	14.1	20.6	-	-	130.3	226.3	111.7	443.2
ChfL [41]	CVPR'22	57.0	235.7	57.5	94.3	6.9	11.0	-	-	80.3	137.6	76.8	343.0
MAN [24]	CVPR'22	53.4	209.9	56.8	90.3	-	-	-	-	77.3	131.5	76.5	323.0
GauNet [5]	CVPR'22	58.2	245.1	54.8	89.1	6.2	9.9	186.3	256.5	81.6	153.7	-	-
CLTR [22]	ECCV'22	59.5	240.6	56.9	95.2	6.5	10.6	-	-	85.8	141.3	74.3	333.8
CrwodHat [54]	CVPR'23	52.3	211.8	51.2	81.9	5.7	9.4	-	-	75.1	126.7	68.7	296.9
STEERER [11]	ICCV'23	54.3	238.3	54.5	86.9	5.8	8.5	-	-	74.3	128.3	63.7	309.8
PET [25]	ICCV'23	58.5	238.0	49.3	78.8	6.2	9.7	-	-	79.5	144.3	74.4	328.5
CrowdDiff		47.3	198.9	47.4	75.0	5.7	8.2	160.8	225.0	68.9	125.6	57.8	221.2

Table 2. Comparison with state-of-the-art methods on the NWPU-Crowd *test* dataset with performance under different scene constraints and luminance conditions. The best results are shown in red. The second-best results are shown in blue.

Method	Venue	Overall			Scene Level (MAE↓)						Luminance (MAE↓)			
		MAE↓	MSE↓	NAE↓	Avg.	S0	S1	S2	S3	S4	Avg.	L0	L1	L2
BL [32]	ICCV'19	105.4	454.2	0.203	750.5	66.5	8.7	41.2	249.9	3386.4	154.7	293.4	102.7	68.0
DM-Count [50]	NeurIPS'20	88.4	388.6	0.169	498.0	146.7	7.6	31.2	228.7	2075.8	117.6	203.6	88.1	61.2
UOT [33]	AAAI'21	87.8	387.5	0.185	566.5	80.7	7.9	36.3	212.0	2495.4	127.2	240.3	86.4	54.9
P2PNet [45]	ICCV'21	72.6	331.6	0.192	510.0	34.7	11.3	31.5	161.0	2311.6	107.8	203.8	69.6	50.1
MAN [24]	CVPR'22	76.5	323.0	0.170	464.6	43.3	8.5	35.3	190.9	2044.9	102.2	180.1	77.1	49.4
Chfl [41]	CVPR'22	76.8	343.0	0.171	470.1	56.7	8.4	32.1	195.1	2058.0	113.9	217.7	74.5	49.6
CLTR [22]	ECCV'22	74.4	333.8	0.165	532.4	4.2	7.3	30.3	185.5	2434.8	106.0	197.1	73.5	47.3
STEERER [11]	ICCV'23	63.7	309.8	0.133	410.6	48.2	6.0	25.8	158.3	1814.5	87.2	155.7	63.3	42.5
CrowdHat [11]	CVPR'23	68.7	296.9	0.182	371.7	5.3	6.9	37.8	183.3	1625.3	108.8	220.4	66.3	39.6
CrowdDiff	-	57.8	221.2	0.120	305.3	4.1	4.9	28.8	166.2	1322.4	79.7	131.8	53.1	54.3

sion method since simply combing these shifted dots results in double counts and worsens the performance otherwise.

The counting branch is added to improve the counting performance with the crowd density maps. We present the counting results for individual realizations with and without the regression network in Tab. 3. We consider the average error performance from different realizations to identify characteristic effects of the counting branch before combining them into a single crowd map. Adding the counting branch has improved the average counting results, and the variation in the counting results has been reduced. The counting branch also promotes feature learning for intermediate time steps with noisy features.

Furthermore, we considered the performance of the counting branch even though it is not used to predict the final count of CrowdDiff. The error metrics for the counting branch are provided in Tab. 4 along with state-of-the-art weakly-supervised crowd counting methods [17, 21]. The counting branch can be considered as a subnetwork that was weakly supervised with features from the denoising network and in this regard, the counting branch of CrowdDiff outperforms existing SOTA weakly-supervised methods.

Crowd map fusion leverages the stochastic nature of the crowd density maps produced by the diffusion process, and

we adopt a systematic way to fuse the maps. In Tab. 5, we present the error metrics for three different methods: Random, Descend-SSIM, and Ascend-SSIM. In the *Random* method, we combine the maps in the order in which they are produced. In the *Descend-SSIM* method, we combine the maps in the order of decreasing similarity. In the *Ascend-SSIM* method, we combine the maps in the order of increasing similarity as described above. The iterative improvement with stochastic generation and the proposed fusion method is shown in Fig. 8. From Tab. 5, the counting performance has improved with the Ascend method, where more locally dissimilar realizations are combined initially. This observation is validated by the performance degradation with the Descend-SSIM method compared to both Ascend-SSIM and Random methods.

Additionally, the fusion of multiple realizations is prone to introduce false positives. Hence, we considered the localization and counting performance after fusing different realizations for UCF-QNRF. We generated four additional realizations for this ablation study, and the corresponding results are provided in Tab. 6 along with the respective inference time. From Tab. 6, we see that the localization and counting performance improves with multiple realizations, demonstrating the advantage of using a generative model

(a) Ground truth: 1174   (b) Ground truth: 70

(c) Ours: 1168   (d) Ours: 69

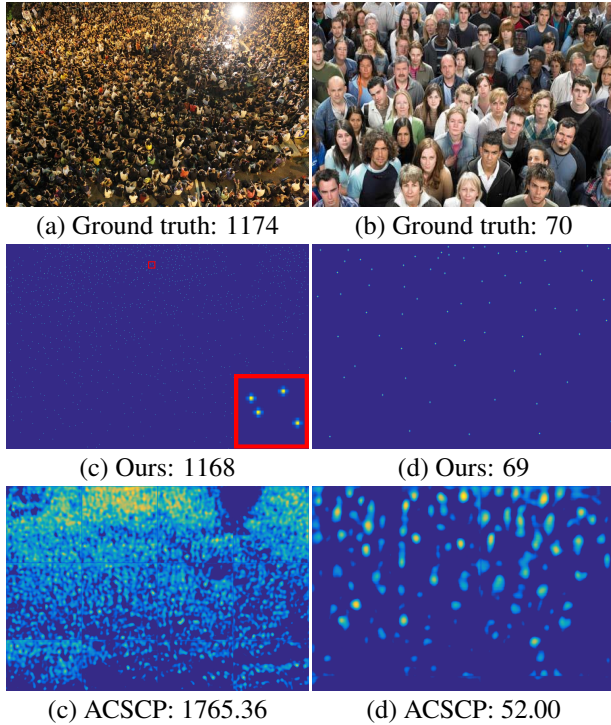(c) ACSCP: 1765.36   (d) ACSCP: 52.00

Figure 6. Generation quality and crowd performance comparison with a narrow kernel between the diffusion models and a GAN-based method (ACSCP) [40] for different crowd scenes.

and fusing the information from multiple realizations. However, a higher number of realizations increases the inference time, and the performance gain from four realizations to eight realizations is insignificant while the inference time has doubled. Consequently, we chose to produce four realizations as the optimal setting considering the performance and inference time trade-off.

Density thresholding is used as an alternative to density summation for the counting operation. The performance comparison between the two methods is tabulated in Tab. 8 for the best-performing realization of each dataset. From Tab. 8, we see that the density summation produces inferior counting results than thresholding despite both methods using the same density map. This is because background noise accumulation with the summation operation and the thresholding method display better noise immunity.

Kernel size of the density kernels used to generate the ground truth density map x_0 affects the generation ability and performance of CrowdDiff. We tabulate the performance with different kernel sizes in Tab. 7. We observed similar performance at 1×1 and 3×3 kernels, and the performance significantly dropped for the latter kernel sizes. This is because the kernel size affects the pixel value distribution of the density map, and the interference between adjacent kernels introduces true positives at local maxima.

Figure 7. Qualitative results for stochastic crowd map generation from two realizations. Green boxes include new dots created at different realizations. Blue boxes include dots present in both realizations but with a shift, and pink boxes include perfectly reassigned dots. (*best viewed in highest zooming level*)

Rejection radius (β) **and nearest neighbors** (k) influence the performance of the fusion of multiple realizations. The results for different β and k values are tabulated in Tab. 9. The rejection criterion is stable around the β values from 0.80 to 0.85. Because a low β value is susceptible to including false positives or duplicates, and a high β value could also reject true positives. However, the performance difference for different k values was insignificant and the best performing setting was selected.

The inference process of the diffusion-based models is iterative and, therefore, exacts higher inference times. However, since we threshold the density map to count the number of kernels rather than summing over the pixel density values, the proposed method is robust to residual noise in the background. With the above exception, we used DDIM sampling to improve the inference procedure by a factor of 20 rather than using the original number of diffusion steps without a significant performance drop.

More results and details can be found in the supplementary.

6. Conclusions

We proposed a novel crowd counting framework where density map generation was treated as a denoising diffusion process. The new framework allows using extremely narrow density kernels with which noise can be suppressed more robustly in crowd density maps. Consequently, we performed density kernel detection on crowd density maps which offered more immunity to noise than density summation. Also, the proposed method could iteratively improve the counting performance via multiple realizations, unlike other crowd counting frameworks, due to the stochastic nature of the generative models. Further, unlike existing density-based methods, our proposed method assigns den-
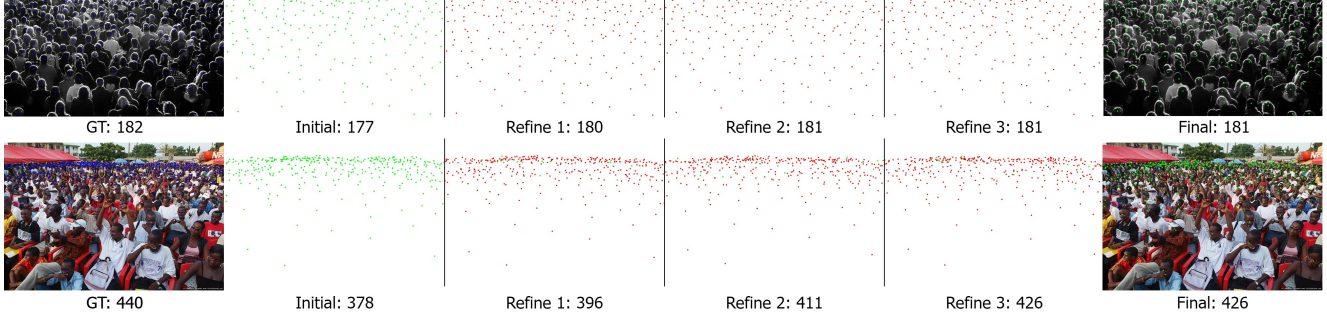
Figure 8. Qualitative results for the crowd density map fusion method. Green dots represent the points combined to the final prediction and red dots represent the points removed from each realization.

Table 3. Error metrics for individual realizations without (top half) and with (bottom half) the counting decoder.

	Method	JHU-Crowd++		ShanghaiTech B		UCF-QNRF	
		MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓
w/o	Best	50.24	206.82	5.90	8.40	75.87	136.85
	Average	52.29	212.22	5.97	8.50	78.35	140.87
	Variance	1.5854	4.1764	0.0926	0.1278	2.3092	3.7404
w/	Best	48.24	201.54	5.82	8.30	72.17	130.86
	Average	48.56	202.38	5.85	8.33	73.08	132.33
	Variance	0.2546	0.6708	0.0209	0.0289	0.64	1.0366

Table 4. Performance of the counting branch in comparison to other weakly-supervised counting methods.

Method	JHU-Crowd++		ShanghaiTech B		UCF-QNRF	
	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓
Counting	53.1	223.5	7.7	12.0	76.6	135.3
TransCrowd [21]	56.8	193.6	9.3	16.1	97.2	168.5
MATT [17]	71.5	210.4	11.7	17.5	122.3	183.2

Table 5. Comparison for crowd map fusion methods.

Method	JHU-Crowd++		ShanghaiTech B		UCF-QNRF	
	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓
Random	47.77	200.3	5.78	8.23	71.04	129.03
Ascend-SSIM	47.26	198.97	5.74	8.18	68.95	125.65
Descend-SSIM	48.10	201.18	5.81	8.27	71.73	130.15

sity kernels at head positions without the need for data heuristics, as required in the localization-based methods.

Acknowledgements

Table 6. Counting and localization results with the number of realizations.

#	Time (ms)	Counting		Localization		
		MAE↓	MSE↓	P (%)↑	R (%)↑	F (%)↑
1	210	74.59	134.78	68.45	67.34	67.89
2	430	71.94	130.49	77.24	75.94	76.58
4	770	68.95	125.65	82.18	80.79	81.48
8	1360	66.97	122.44	83.06	81.67	82.36

Table 7. Performance comparison for different kernel sizes.

Kernel size	σ	MAE↓	MSE↓	P(%)↑	R(%)↑	F(%)↑
1×1	-	69.21	126.07	81.70	80.32	81.00
3×3	0.5	68.95	125.65	82.18	80.79	81.48
5×5	1	81.87	146.58	57.43	56.52	56.97
9×9	2	94.68	167.32	34.12	33.65	33.88

Table 8. Comparison between crowd counting operations and the effect of noise.

Method	JHU-Crowd++		ShanghaiTech A		UCF-CC-50	
	MAE↓	MSE↓	MAE↓	MSE↓	MAE↓	MSE↓
Density thresholding	48.24	201.54	47.81	75.91	163.56	228.32
Density estimation	215.40	515.63	156.96	243.70	180.58	254.91
Noise residual	200.94	502.31	186.04	294.44	70.68	99.38

Table 9. Ablation for different rejection radii (β) and nearest neighbor (k) values.

Metric	Rejection radius (β) $k = 4$				Nearest neighbors (k) $\beta = 0.85$		
	0.75	0.80	0.85	0.90	3	4	5
MAE↓	72.2	69.71	68.95	72.03	69.41	68.95	69.07
MSE↓	130.91	126.88	125.65	130.64	126.39	125.65	125.84
P(%)↑	79.58	81.67	82.18	81.64	81.83	82.18	82.01
R(%)↑	78.68	79.94	80.79	79.36	80.40	80.79	80.55
F(%)↑	79.13	80.80	81.48	80.48	81.11	81.48	81.27

References

[1] Shahira Abousamra, Minh Hoai, Dimitris Samaras, and Chao Chen. Localization in the crowd with topological constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 872–881, 2021. 6

[2] Tomer Amit, Eliya Nachmani, Tal Shaharbany, and Lior Wolf. Segdiff: Image segmentation with diffusion probabilistic models. *arXiv preprint arXiv:2112.00390*, 2021. 2

[3] Shuai Bai, Zhiqun He, Yu Qiao, Hanzhe Hu, Wei Wu, and Junjie Yan. Adaptive dilated network with self-correction supervision for counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4594–4603, 2020. 1, 2

[4] Shoufa Chen, Peize Sun, Yibing Song, and Ping Luo. Diffusiondet: Diffusion model for object detection. *arXiv preprint arXiv:2211.09788*, 2022. 2, 4

[5] Zhi-Qi Cheng, Qi Dai, Hong Li, Jingkuan Song, Xiao Wu, and Alexander G Hauptmann. Rethinking spatial invariance of convolutional networks for object counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19638–19648, 2022. 6

[6] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11472–11481, 2022. 2

[7] Kamil Deja, Tomasz Trzcinski, and Jakub M Tomczak. Learning data representations with joint diffusion models. *arXiv preprint arXiv:2301.13622*, 2023. 2

[8] Guoxiu Duan, Aichun Zhu, Lu Zhao, Xiaomei Zhu, Fangqiang Hu, and Xinjie Guan. Mask-based generative adversarial networking for crowd counting. *Journal of Electronic Imaging*, 30(4):043027–043027, 2021. 2

[9] Hui Gao, Wenjun Zhao, Dexian Zhang, and Miaolei Deng. Application of improved transformer based on weakly supervised in crowd localization and crowd counting. *Scientific Reports*, 13(1):1144, 2023. 1

[10] Zhangxuan Gu, Haoxing Chen, Zhuoer Xu, Jun Lan, Changhua Meng, and Weiqiang Wang. Diffusioninst: Diffusion model for instance segmentation. *arXiv preprint arXiv:2212.02773*, 2022. 2, 4

[11] Tao Han, Lei Bai, Lingbo Liu, and Wanli Ouyang. Steerer: Resolving scale variations for counting and localization via selective inheritance learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21848–21859, 2023. 6

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. 2

[13] Yutao Hu, Xiaolong Jiang, Xuhui Liu, Baochang Zhang, Jungong Han, Xianbin Cao, and David Doermann. Nascount: Counting-by-density with neural architecture search. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, pages 747–766. Springer, 2020. 1

[14] Haroon Idrees, Imran Saleemi, Cody Seibert, and Mubarak Shah. Multi-source multi-scale counting in extremely dense crowd images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2547–2554, 2013. 5

[15] Haroon Idrees, Muhmmad Tayyab, Kishan Athrey, Dong Zhang, Somaya Al-Maadeed, Nasir Rajpoot, and Mubarak Shah. Composition loss for counting, density map estimation and localization in dense crowds. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–546, 2018. 5

[16] Xiaoheng Jiang, Li Zhang, Mingliang Xu, Tianzhu Zhang, Pei Lv, Bing Zhou, Xin Yang, and Yanwei Pang. Attention scaling for crowd counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4706–4715, 2020. 1

[17] Yinjie Lei, Yan Liu, Pingping Zhang, and Lingqiao Liu. Towards using count-level weak supervision for crowd counting. *Pattern Recognition*, 109:107616, 2021. 6, 8

[18] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479:47–59, 2022. 2

[19] Yuhong Li, Xiaofan Zhang, and Deming Chen. Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1091–1100, 2018. 2

[20] Dongze Lian, Jing Li, Jia Zheng, Weixin Luo, and Shenghua Gao. Density map regression guided detection network for rgb-d crowd counting and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1821–1830, 2019. 2

[21] Dingkang Liang, Xiwu Chen, Wei Xu, Yu Zhou, and Xiang Bai. Transcrowd: weakly-supervised crowd counting with transformers. *Science China Information Sciences*, 65(6): 160104, 2022. 1, 3, 6, 8

[22] Dingkang Liang, Wei Xu, and Xiang Bai. An end-to-end transformer model for crowd localization. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part I*, pages 38–54. Springer, 2022. 1, 2, 6

[23] Dingkang Liang, Wei Xu, Yingying Zhu, and Yu Zhou. Focal inverse distance transform maps for crowd localization. *IEEE Transactions on Multimedia*, 2022. 3

[24] Hui Lin, Zhiheng Ma, Rongrong Ji, Yaowei Wang, and Xiaopeng Hong. Boosting crowd counting via multifaceted attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19628–19637, 2022. 6

[25] Chengxin Liu, Hao Lu, Zhiguo Cao, and Tongliang Liu. Point-query quadtree for crowd counting, localization, and more. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1676–1685, 2023. 6

[26] Liang Liu, Hao Lu, Haipeng Xiong, Ke Xian, Zhiguo Cao, and Chunhua Shen. Counting objects by blockwise classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(10):3513–3527, 2019. 2

[27] Liang Liu, Hao Lu, Hongwei Zou, Haipeng Xiong, Zhiguo Cao, and Chunhua Shen. Weighing counts: Sequential crowd

counting by reinforcement learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 164–181. Springer, 2020. 2

[28] Weizhe Liu, Mathieu Salzmann, and Pascal Fua. Context-aware crowd counting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5099–5108, 2019. 2

[29] Xiyang Liu, Jie Yang, Wenrui Ding, Tieqiang Wang, Zhijin Wang, and Junjun Xiong. Adaptive mixture regression network with local counting map for crowd counting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 241–257. Springer, 2020. 2

[30] Yuting Liu, Miaojing Shi, Qijun Zhao, and Xiaofang Wang. Point in, box out: Beyond counting persons in crowds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2019. 1, 2

[31] Ao Luo, Fan Yang, Xin Li, Dong Nie, Zhicheng Jiao, Shangchen Zhou, and Hong Cheng. Hybrid graph neural networks for crowd counting. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11693–11700, 2020. 2

[32] Zhiheng Ma, Xing Wei, Xiaopeng Hong, and Yihong Gong. Bayesian loss for crowd count estimation with point supervision. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6142–6151, 2019. 6

[33] Zhiheng Ma, Xing Wei, Xiaopeng Hong, Hui Lin, Yunfeng Qiu, and Yihong Gong. Learning to count via unbalanced optimal transport. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2319–2327, 2021. 2, 6

[34] Yanda Meng, Hongrun Zhang, Yitian Zhao, Xiaoyun Yang, Xuesheng Qian, Xiaowei Huang, and Yalin Zheng. Spatial uncertainty-aware semi-supervised crowd counting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15549–15559, 2021. 1, 6

[35] Yunqi Miao, Zijia Lin, Guiguang Ding, and Jungong Han. Shallow feature based dense attention network for crowd counting. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11765–11772, 2020. 2

[36] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 2, 3, 5

[37] Min-hwan Oh, Peder Olsen, and Karthikeyan Natesan Ramamurthy. Crowd counting with decomposed uncertainty. In *Proceedings of the AAAI conference on artificial intelligence*, pages 11799–11806, 2020. 2

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 5

[39] Deepak Babu Sam, Skand Vishwanath Peri, Mukuntha Narayanan Sundararaman, Amogh Kamath, and R Venkatesh Babu. Locate, size, and count: accurately re-

solving people in dense crowds via detection. *IEEE transactions on pattern analysis and machine intelligence*, 43(8): 2739–2751, 2020. 1, 2

[40] Zan Shen, Yi Xu, Bingbing Ni, Minsi Wang, Jianguo Hu, and Xiaokang Yang. Crowd counting via adversarial cross-scale consistency pursuit. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5245–5254, 2018. 2, 5, 7

[41] Weibo Shu, Jia Wan, Kay Chen Tan, Sam Kwong, and Antoni B Chan. Crowd counting in the frequency domain. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19618–19627, 2022. 1, 6

[42] Vishwanath A Sindagi, Rajeev Yasarla, and Vishal M Patel. Pushing the frontiers of unconstrained crowd counting: New dataset and benchmark method. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1221–1231, 2019. 5

[43] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 2

[44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020. 5

[45] Qingyu Song, Changan Wang, Zhengkai Jiang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yang Wu. Rethinking counting and localization in crowds: A purely point-based framework. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3365–3374, 2021. 1, 2, 6

[46] Qingyu Song, Changan Wang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Jian Wu, and Jiayi Ma. To choose or to fuse? scale selection for crowd counting. In *Proceedings of the AAAI conference on artificial intelligence*, pages 2576–2583, 2021. 1

[47] Ye Tian, Xiangxiang Chu, and Hongpeng Wang. Cctrans: Simplifying and improving crowd counting with transformer. *arXiv preprint arXiv:2109.14483*, 2021. 1

[48] Jia Wan and Antoni Chan. Adaptive density map generation for crowd counting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1130–1139, 2019. 1

[49] Jia Wan, Qingzhong Wang, and Antoni B Chan. Kernel-based density map generation for dense object counting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1357–1370, 2020. 1

[50] Boyu Wang, Huidong Liu, Dimitris Samaras, and Minh Hoai Nguyen. Distribution matching for crowd counting. *Advances in neural information processing systems*, 33:1595–1607, 2020. 1, 6

[51] Qi Wang, Junyu Gao, Wei Lin, and Yuan Yuan. Learning from synthetic data for crowd counting in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8198–8207, 2019. 1

[52] Qi Wang, Junyu Gao, Wei Lin, and Xuelong Li. Nwpu-crowd: A large-scale benchmark for crowd counting and localization. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):2141–2149, 2020. 5

[53] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 4

[54] Shaokai Wu and Fengyu Yang. Boosting detection in crowd analysis via underutilized output features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15609–15618, 2023. 5, 6

[55] Haipeng Xiong, Hao Lu, Chengxin Liu, Liang Liu, Zhiguo Cao, and Chunhua Shen. From open set to closed set: Counting objects by spatial divide-and-conquer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8362–8371, 2019. 2

[56] Chenfeng Xu, Dingkang Liang, Yongchao Xu, Song Bai, Wei Zhan, Xiang Bai, and Masayoshi Tomizuka. Autoscale: learning to scale for crowd counting. *International Journal of Computer Vision*, 130(2):405–434, 2022. 1, 2

[57] Hai-Yan Yao, Wang-Gen Wan, and Xiang Li. Mask guided gan for density estimation and crowd counting. *IEEE Access*, 8:31432–31443, 2020. 2

[58] Anran Zhang, Lei Yue, Jiayi Shen, Fan Zhu, Xiantong Zhen, Xianbin Cao, and Ling Shao. Attentional neural fields for crowd counting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5714–5723, 2019. 1

[59] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 589–597, 2016. 5

[60] Xiaopin Zhong, Guankun Wang, Weixiang Liua, Zongze Wua, and Yuanlong Deng. Mask focal loss for dense crowd counting with canonical object detection networks. *arXiv preprint arXiv:2212.11542*, 2022. 2