PDANet: Pyramid Density-aware Attention based Network for Accurate Crowd Counting

Saeed Amirgholipour[a,b], Wenjing Jia[a], Lei Liu[c], Xiaochen Fan[a], Dadong Wang[b], Xiangjian He[*,a]

[a] *School of Electrical and Data Engineering, University of Technology Sydney, Australia*
[b] *The Quantitative Imaging Research Team, CSIRO Data61, Australia*
[c] *School of Instrumentation Science and Opto-Electronics Engineering, Beihang University, China*

Abstract

Crowd counting, *i.e.*, estimating the number of people in crowded areas, has attracted much interest in the research community. Although many attempts have been reported, crowd counting remains an open real-world problem due to the vast density variations and severe occlusion within the interested crowd area. In this paper, we propose a novel Pyramid Density-Aware Attention based network, abbreviated as PDANet, which leverages the attention, pyramid scale feature, and two branch decoder modules for density-aware crowd counting. The PDANet utilizes these modules to extract features of different scales while focusing on the relevant information and suppressing the misleading information. We also address the variation of crowdedness levels among different images with a Density-Aware Decoder (DAD) modules. For this purpose, a classifier is constructed to evaluate the density level of the input features and then passes them to the corresponding high and low density DAD modules. Finally, we generate an overall density map by considering the summation of low and high crowdedness density maps. Meanwhile, we employ different losses aiming to achieve a precise density map for the input scene. Extensive evaluations conducted on the challenging benchmark datasets well demonstrate the superior performance of the proposed PDANet in terms of the accuracy of counting and generated density maps over the well-known state-of-the-art approaches.

Key words: Crowd Counting, Pyramid Module, Density ware, Attention Module, Classification Module, Convolutional Neural Networks.

1. Introduction

Nowadays, crowd counting has become an important task for a variety of applications, such as traffic control [1], public safety [2], and scene understanding [3, 2]. As a result, density estimation techniques have become a research trend for various counting tasks. These techniques utilize trained regressors to estimate people density for each area so that the summation of the resultant density functions can yield the final count of the crowd. A variety of regressors, such as Gaussian Processes [4], Random Forests [5], and more recently, deep learning based networks [6, 7, 8] have been used for crowd counting and density estimation. The state-of-the-art approaches are mostly deep learning based approaches due to their capabilities of producing accurate density maps and precise crowd counting [1, 9].

Generally, the approaches based on deep neural networks (DNNs) utilize standard convolutions or dilated convolutions as their backbone network to learn local patterns and density maps [8, 10]. Most of them use the same filters, pooling matrices, and settings across the whole image, and implicitly assume the same congestion level everywhere [6]. However, this assumption often does not hold in reality.

To better understand the effect of this mis-assumption, let us show some examples with clearly different levels of crowdedness. Fig. 1 presents some exemplar images of different congestion scenarios. Fig. 1(a) shows a highly crowded image with more than 1,000 people, while Fig. 1(c) presents a less crowded scene having less than 70 people. However, if we look at Fig. 1(a), we notice that there is a relatively more congested area, which is shown in Fig. 1(b). The same situation can be seen in Fig. 1(c), where a small area within this crowd, as shown in Fig. 1(d), is clearly more crowed than other areas.

Due to this dynamic variation in crowded scenes, naturally we should utilize different features and branches to respond and capture details at different levels of crowdedness. In the past, this has been attempted by four major types of approaches, *i.e.*, defining separate pathways from the lower layers and utilizing different sizes of the convolutional filters, image pyramid-based methods [11, 1], detection-based crowd counting [6], patch-based crowd counting [12, 10, 13], and multi-level feature based methods [11]. Although these methods achieved robust performance with some different tactics, there are still lots of spaces to improve their performances by designing highly efficient DNN structures that can deal with crowd scenes with dramatic density varieties effectively.

First, generally speaking, a kernel size of 3 × 3 for a convolution filter is more effective than larger ones in terms of ex-

[*] Corresponding author
Email addresses: Saeed.AmirgholipourKasmani@student.uts.edu.au (Saeed Amirgholipour), Wenjing.Jia@uts.edu.au (Wenjing Jia), BY1417114@buaa.com.cn (Lei Liu), Xiaochen.Fan@student.uts.edu.au (Xiaochen Fan), Dadong.Wang@csiro.au (Dadong Wang), Xiangjian.He@uts.edu.au (Xiangjian He)

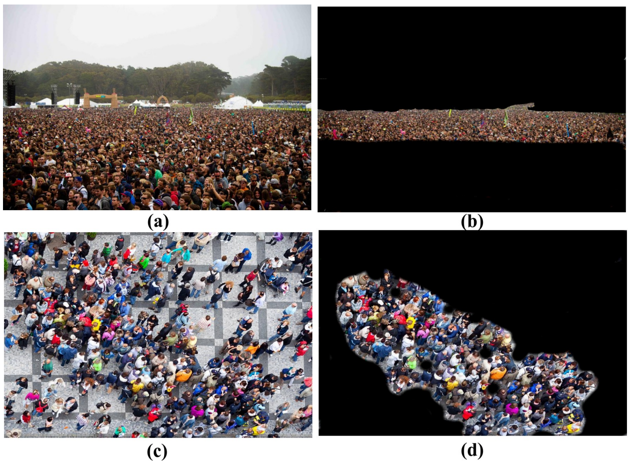


Figure 1: Examples of crowded and sparse images. (a) and (c) show an example of a highly crowded scene and a less crowded scene, respectively, while (b) and (d) show their corresponding congested areas.

tracting more meaningful features, because more details can be captured with lower complexities without making it more difficult to train the network [14, 15, 16]. Kang *et al.* [17] also proved that smaller receptive fields gave better performance. Secondly, using patch-based processing and multi-patch processing is time costly due to that the same features have to pass through different paths and patch multiple times. If we want to take the benefits of multi-patch or multi-column based approaches, it is better to extract some coarse features from the initial layers and then pass them to some branches for further zooming in to find more sophisticated features. To utilize a deeper network for crowd counting, we need an approach that can deploy the aforementioned proposals on the multi-column structure to achieve better performance.

In this paper, we present a deep encoder-decoder based architecture named as Pyramid Density-aware Attention-based Network (PDANet), which combines the pyramid feature extraction with spatial and channel attentions to produce richer features for estimating crowds of various levels of crowdedness and scales. In our work, we use the VGG16 as the feature extractor for the encoder to produce features for the decoder of the model. To learn multi-scale features, we first use a cascade of Global Average Pooling (GAP), 1×1 convolution and dilated convolutions with kernels of 3×3 to extract more descriptive features with different scales from the VGG16 features. Then, we apply the channel and spatial attentions on different layers to enhance and boost the quality of the features in order to obtain more accurate density maps. On the other hand, to make the model adaptive to different density levels within an image, we introduce a classification module to classify the crowdedness level of the input scene and develop generation models of low and high crowded density maps for the input image.

This work is different in several ways from the existing crowd counting approaches that use the pyramid contextual information and attention modules. (a) Unlike the DENet [6], the proposed PDANet does not separate models for counting people in sparsely crowded areas and estimating the human density maps in the remaining areas in the image. (b) The first main characteristic of our proposed PDANet is its density awareness by adopting the pyramid and attention modules. Different from other works attempting to address this problem of density variety, *e.g.* [12], our PDANet does not separate the input scene into different patches. Instead, we use multipath branches to address the intra-density variations within the input scene. Experimental results show that the pyramid and attention modules contribute a 5 to 20 percent improvement over the baseline model. (c) Pyramid Feature Extractor (PFE) is the second noticeable contribution of our PDANet, where we utilize a new combination of GAP, 1×1 convolution, and Atrous convolution, resulting in a difference from the existing approaches in terms of the orders and parameters that can better aggregate local scale features and is more effective than the existing solutions. (d) The third remarkable feature of PDANet is its attention modules. The architecture of our end-to-end attention modules is also different from ADCrowdNet [18] because it uses the combination of the spatial and channel attention modules within the architecture. Furthermore, it is trained in an end-to-end way based on the crowd counting dataset, instead of separately using external datasets to train the attention module as in ADCrowdNet [18]. Compared with the work in [19], our PDANet has also adopted another spatial-based module in the DAD module to optimize the density map results based on feature maps of the sparse and dense areas within the input scene. (e) The last distinctive characteristic of the PDANet is its classification modules, which are different from the existing work [19]. Our PDANet passes the input image to two different sub-models with different receptive fields to evaluate the lower and higher bounds of the density map and then combines them with the help of the channel attention module. Our PDANet introduces a classification module that classifies the input image into low or high density and then passes them to the corresponding DAD modules.

To summarize, the major contributions of this paper are:

- We propose a density-aware crowd counting solution to address crowd areas of various scales and density levels.This density-awareness feature helps the model to handle density variation between different images as well as within each input scene.
- We first integrate the pyramid multi-scale feature extraction mechanism in the feature extractor to extract rich features. Then, we integrate the channel and spatial attention modules and propose an end-to-end trainable density estimation pipeline.
- For estimating densities of crowd with not only high and low crowdedness levels but also medium-level density areas, we propose to use a combination of classification and regression losses to address the overall and within-the-scene density variation in the density maps.
- Extensive experiments are conducted on several challenging benchmark datasets to demonstrate the superior performance of our proposed PDANet approach over the state-of-the-art solutions. We also preform comprehensive ablation studies to validate the effectiveness of each component in our proposed approach.

The rest of the paper is organized as follows. In Sect. II, we introduce the existing works related to our approach. The proposed PDANet model for crowd density estimation is introduced in detail in Sect. III. In Sect. IV, we evaluate the performance of our PDANet on benchmark datasets. Sect. V provides ablation studies on various parts of the proposed model. Finally, we draw conclusions in Sect. VI.

2. Related Work

In this section, we review the recent crowd counting works related to our PDANet model. In 2018, Qi *et al.* [20] proposed a new multiview clustering model for detecting coherent groups within crowd images. Their proposed techniques helped researchers to find the areas with similar patterns in the input scene. Recently, [21] proposed two effective methods, i.e., domain adaptation and pre-training scheme, to boost the performance of the existing crowd counting solutions. The new pre-training method with the synthetic data has improved the accuracy of crowd counting solutions by a large margin. The second contrition of [21] helped researchers in the crowd understanding tasks to obtain a huge amount of crowd counting data without extra labelling work.

However, many studies have been done based on multi-column architectures [22, 23]. One of the initial works was done by Zhang *et al.* [24], who proposed a three-CNN-column based MCNN structure, each with different receptive parameters to handle different head sizes. Recently, Tian *et al.* proposed the PaDNet [23], which was composed of several components such as a Density-Aware Network (DAN), Feature Enhancement Layer (FEL), and a Feature Fusion Network (FFN). PaDNet improved the-state-of-the-art results remarkably by capturing pan-density information and utilizing global and local contextual features. IG-CNN [25] is another approach that combined the clustering and crowd counting for estimating the density map adaptively based on training a mixture of experts that could incrementally adapt and grow based on the complexity of the dataset. Sindagi *et al.* proposed a new multi-column network, *i.e.*, CP-CNN [26], which added two other branches to classify the image-wise density to provide the global and local context information to the MCNN model. Deb *et al.* [27] incorporated the Atrous convolutions into the multi-branch network by assigning different dilation rates to various branches.

Most recently, Shi *et al.* [28] proposed a perspective information CNN-based model PACNN for crowd counting. Their model combined the perspective information with a density regression to address the person's scale change within an image. They created the ground-truth perspective map and used it to generate perspective-aware weighting layers to combine the results of multi-scale density adaptively. Wan *et al.* [29] proposed a new model called RRSP to utilized the correlation information in a training dataset for accurate crowd counting.

Some of the recent studies focused on utilizing pyramid and attention-based modules [30]. The Pyramid modules were introduced by Zhao *et al.* [31] to extract quality features for scene semantic segmentation tasks. They introduced an efficient method to estimate human head sizes and integrated them to an attention module to aggregate density maps from different layers and generate the final density map. Liu *et al.* [1] presented another end-to-end multi-scaled solution CAN based on fusing multi-scale pyramid features. They used modified PSP modules for extracting multi-scale features from the VGG16 features to address the rapid scale changes within the scene. Their model leveraged multi-scale adaptive pooling operations to cover a variety range of receptive fields. Compared to CAN, Chen *et al.* proposed an end-to-end single-column structure as a Scale Pyramid Network (SPN), which extracted multi-scale features with the dilated convolutions with various dilation rates $(2, 4, 8,$ and $12)$.

On the other hand, the attention module proposed in [32] aimed to re-calibrate the features adaptively, so as to highlight the effect of those valuable features while suppressing the impact of weak ones [33]. Rahul *et al.* proposed an attention-based model to regress multi-scale density maps from several intermediate layers [18]. We recently proposed a DENet [6], which utilized mask-RCNN for counting people in low crowded areas and an Xception based regressor for regressing crowd density in highly crowded areas. One of the latest research in the area of crowd counting is ADCrowdNet [18], which utilized a two-step cascade encoder-decoder architecture, one for the detection of crowded areas and producing the attention map as Attention Map Generator (AMG), and the other one for generating the density map called Density Map Estimator (DME). Their method achieved excellent results on the ShanghaiTech Part A dataset. However, it has some significant drawbacks, such as that (a) it needed an external dataset to train AMG to detect the crowded areas, and (b) after producing the attention map, they applied it on the input scene to create masked input data for DME and again extracting features with a similar encoder-decoder structure. We believe that it is redundant and time consuming due to passing the input scene twice rather than applying the generated mask on the latest layer of AMG module and use the feature maps for the next stage.

3. Pyramid Density-aware Attention Net

In this section, we first present the general structure of our proposed PDANet, and then present more details about each of the modules used.

3.1. Overview

The overall architecture of our PDANet is illustrated in Fig. 2. This framework contains five main components, *i.e.*, a Feature Extractor, a Pyramid Feature Extractor (PFE), a Classifier, a Density Aware Decoder (DAD), and an Attention Module.

The backbone of our PDANet is based on VGG16 [14], which is widely used for extracting low-level features. We eliminate the layers between the last two pooling layers considering the trade-off between resource consumption and accuracy [34]. Then, we apply a channel and spatial attention modules to highlight the most significant features. Then, these features are fed into the PFE module, which incorporates the combination of adaptive pooling and 1×1 and 3×3 dilated convolutions to

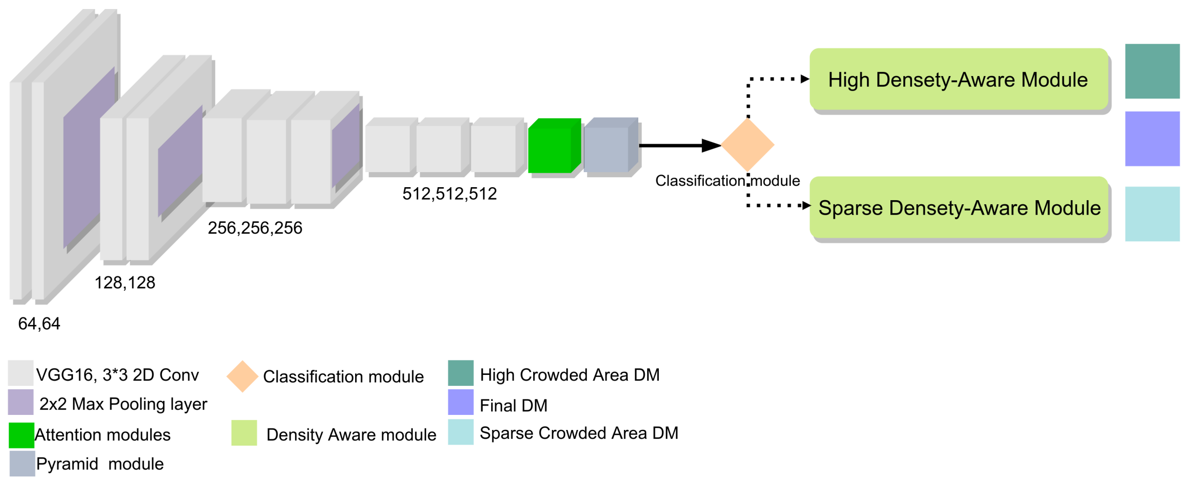


Figure 2: The overview of our proposed PDANet network. This architecture contains a VGG16 based feature extractor, a Pyramid module, an Attention module, a Classification module, and a Decoder module.

produce scale-aware features for the last layers of the decoder module. Next, we incorporate a GAP and a fully connected layer to classify the input scene as a highly dense or a sparse one. This information is then passed to the respective decoder which contains four 3×3 dilated convolution layers, each empowered with an attention module. Furthermore, to address the crowdedness differences in sparse and dense areas, we design two branches of the decoder module to generate low and high-density maps within the input scene and assign them to the corresponding regression losses. In the final step, we use the dense and sparse features from the last layer of the decoder to produce the final output density map (DM). Our PDANet uses the same loss for sparse, dense and final output DMs, and a classification loss to train the model in an end-to-end manner.

To summarize, in our proposed PDANet, each part plays a role in the overall performance.

- The Attention Module focus its attention on the significant features (crowded areas).
- The Pyramid Feature Extractor generates more descriptive features suitable for counting crowds with scale variation, through a combination of adaptive pooling algorithms and dilated convolutions with different scales.
- The Classifier helps to find the proper branch of the decoder according to the crowdedness level of the area.
- The mid-branch Decoder is to address congestion changes within the input image.

3.2. Channel and Spatial Attention Modules

In this study, we re-calibrate the feature maps adaptively by mixing attention modules to augment the effect of the essential features, while suppressing the weak ones. We use the combination of spatial and channel attentions for finding and separating the crowded areas within the input image. As it is shown in Fig. 2 (see the green module), we utilize the channel and spatial attention [33] after the convolution layers. This module contains the channel and the spatial attentions to produce the final attention features in each layer. We combine the results of these two attentions by an element-wise max of the channel and the spatial excitations to generate output features in each layer. The other attention module is a spatial attention map that is generated based on the density map of the sparse and dense crowded areas within the image. We apply a sigmoid on this attention module and multiply it with the joint convolution feature maps from the last layer of a sparse and dense decoder.

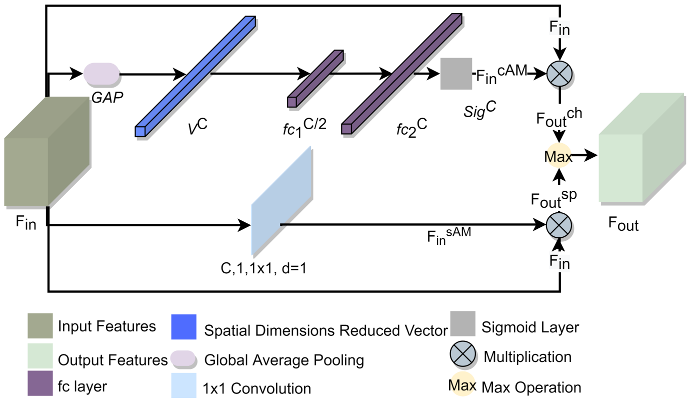


Figure 3: Illustration of the attention module of our model. The top branch generates channel-based attention, while the bottom branch generates the spatial attention map.

Fig. 3 illustrates this attention module. As shown in this figure, there are two branches in this illustration, *i.e.*, the channel attention branch on the top, and the spatial attention branch on the bottom. The channel attention branch utilizes a cascade of GAP and two fully connected layers with the size of $\frac{C}{2}$ and C, respectively (C is the channel size of a convolution layer). Then, after applying a sigmoid on the result, we do element-wise multiplication between the channel attention map and the input feature maps to obtain channel-wise weight corrected feature maps.

As we explained in the previously, to apply the channel based attention mechanism, we first perform GAP on the input feature map F_{in}, to obtain V^C, and then transform them by two fully-connected layers $f_{c_1}^{C/2}$ and $f_{c_2}^{C}$, as shown in Fig. 3 and Eq. 1 as:

$$F_{in}^{cAM} = \mathrm{Sig}(f_{c2}^{C}(f_{c1}^{C/2}(V^C))). \tag{1}$$

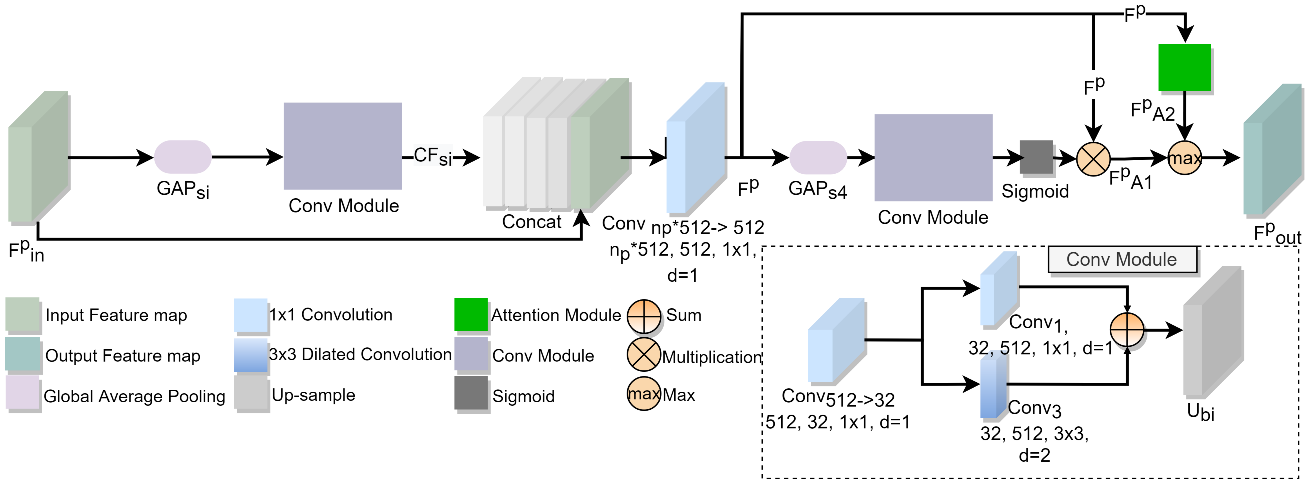


Figure 4: The overview of the Pyramid Feature Extractor (PFE) module. The PFE module uses 1 × 1 and 3 × 3 dilated kernel convolutions with the GAP to extract features of different scales from the VGG16 features.

where Sig is a sigmoid function that yields the value in a range of [0, 1] to find the impact of each layer in the feature maps.

Therefor, for channel based attention, features F^{ch}_{out} are obtained by multiplying the encoded channel-wise dependencies (F^{cAM}_{in}) with F_{in} to get F^{ch}_{out}. On the other hand, to obtain the spatial attention map, F^{sAM}_{in}, we perform a 1×1 convolution, *i.e.*, $Conv \in \mathbb{R}^{1\times C\times 1\times 1}$, on the input feature map. Thus, we can measure the importance of a spatial information of each pixel or location within F_{in}. In the next stage, we multiply the spatial attention map with the input feature maps to get the final spatial attention features F^{sp}_{out}, which augment relevant spatial locations and suppress irrelevant ones. Finally, we combine the results of these two attentions by element-wise max of the channel and spatial excitation, *i.e.*, $F_{out} = \max(F^{ch}_{out}, F^{sp}_{out})$. These feature maps amplify the input feature map data and re-calibrate the crowded area within each input convolution layer.

3.3. Pyramid Feature Extractor (PFE)

In this section, we present the details of the Pyramid Feature Extractor (PFE), which is inspired by the Spatial Pyramid Pooling [11]. The PFE fuses features under various pyramid scales by a combination of GAP and two shared 2D convolution layers with a mixture of 1×1 and 3×3 dilated kernels. The general operation of PFE is illustrated in Fig. 4.

We extract contextual features with various GAPs. In the PFE module, we keep the ratio of the input feature map with GAP_{s_i} at scale s_i, for $i = 2, 3, \cdots, 10$ and produce contextual features for each channel with a size of $Hs_i \times W_{s_i}$. For example, if we have an input feature map with $\mathbb{R}^{1\times C\times H\times W}$, GAP_{s_2} utilize the global average pooling layer to generate scaled feature map with a size of $\mathbb{R}^{1\times C\times \frac{H}{2}\times \frac{W}{2}}$, where H_{s_2} and W_{s_2} are equal to $\frac{H}{2}$ and $\frac{W}{2}$, respectively. Various scales of contextual features form the pooled representations for different areas and provide rich information about the density levels in various sub-regions of the input image. The results presented in the Experiments section are based on the scenario utilizing three GAP_{s_i} with scale of s_2 , s_4 , and s_8, respectively. In the Ablation Study shown in Section V, we compare several scenarios for the use of GAP_{s_i}.

Then, we feed GAP_{s_i} to the Conv Module to improve the representation power of the feature map. Note that this procedure is different from the architectures that reduce the dimension of the input feature map with convolutions [11], which utilized various dilation rates to capture multi-scale features and increased time and computational cost. As illustrated in Fig. 4, we perform the Conv operation as:

$$\begin{aligned}\text{CF}_{s_i} = U_{bi}(&\text{Conv}_1(\text{Conv}_{512\to 32}(\text{GAP}_{s_i}))\\ &+ \text{Conv}_3(\text{Conv}_{512\to 32}(\text{GAP}_{s_i}))),\end{aligned} \quad (2)$$

where, for each scale s_i, CF_{s_i}, is the shared Conv module that comes with a bi-linear interpolation to up-sample the contextual features U_{bi} to the ones of the same size as that of F^{p}_{in}.

The shared layer contains one 1×1 convolution ($\text{Conv}_{512\to 32}$) to reduce the number of channels from 512 to 32. We do this to reduce the number of parameters that need to train and reduce the computational cost of PFE. In the subsequent stage, we get the summation of a 1×1 convolution (Conv_1), and a 3×3 dilated convolution (Conv_3) as a piece of extra bonus information captured from surrounding contextual features within GAP_{s_i}. Experimentally, we verify that this combination of convolution filters improves the performance of the PFE module in the density estimation task. Finally, we concatenate all CF_{s_i} and the input features F^{p}_{in} with a 1×1 convolution. We reduce the number of the channels to that of the original VGG features F^{p}_{in}, which is defined as:

$$F^{p} = \text{Conv}_{n_p*512\to 512}(\text{Concat}(\text{CF}_{s_i}, F^{p}_{in})), \quad (3)$$

where n_p is the number of pyramid contextual features CF_{s_i}, plus the features in the original feature map, and $\text{Conv}_{n_p*512\to 512}$ is a 1 convolution to reduce the number of channel to 512.

Then, we utilize a spatial attention module, which is the combination of the Conv module and attention module as explained in Section 3.2. We pass F^{p} to two separate attention branches. As illustrated in Fig. 4, in the bottom, we feed the F^{p} to the GAP_{s_4} layer and reduce the input size to $\frac{H}{4} \times \frac{W}{4}$, and then apply the Conv module. We apply the GAP to highlight and refine the most important parts of the output feature maps. Then,

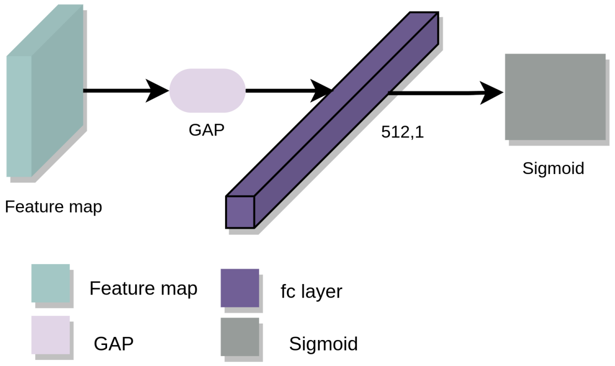


Figure 5: Illustration of the classification module of PDANet. It uses the global average pooling with a fully connected layer to determine the dense level of input scene.

after performing Sig on the output of the Conv module, we apply by the element-wise multiplication to produce the attention map by Conv module (F^p_{A1}). On the top, we also perform the attention module that we discussed in the Section 3.2 to generate the attention feature map output (F^p_{A2}). Finally, we combine the results of these two attentions by the element-wise max operation as:

$$F^p_{out} = \max(F^p_{A1}, F^p_{A2}). \tag{4}$$

At the end of the PFE module, we use multiplication in the bottom path to achieve the effect of pyramid module on the output feature maps. In the top path, we utilize the max operation to obtain the maximum feature value at each pixel of the feature map. This will help us to take the advantage of both attention maps from the pyramid feature and attention module [33]. Altogether, as illustrated in Fig. 4, the PFE module extracts contextual features CF_{s_i} as discussed above, and then feed them to the classification module and a Density Aware Decoder (DAD) module that produces the density map.

3.4. Classification Module

The next step in our framework, as illustrated in Fig. 2, is to decide whether the input contextual features are dense or sparse. In other words, this module classifies the input feature maps as a high density one or the low density (sparse) one. We do this to address the huge variation of crowd densities among different images. It categorizes the input images into either highly crowded images or sparsely crowded images, as shown in Fig. 1(a) and (c), respectively. We pass the input features to the suitable DAD to adaptively react to the density level of the input image and provide a better estimation for crowd density. To model this, as shown in Fig. 5, we introduce a binary classification module to learn how to classify the input feature maps into two classes, *i.e.*, dense or non-dense (*aka*, sparse), as:

$$\text{Cl}^{est}_t = \text{Sig}(f_c(\text{GAP})), \tag{5}$$

where GAP is global average pooling with the scale of 1 × 1 and produces a vector with the size of 512, f_c is a fully connected layer, and Sig is the sigmoid function that yields the value in a range of [0,1] indicating the impact of each layer on the feature maps.

Thus, the classification module produces a class probability, which is a value in the range of [0,1]. If the output probability (Cl^{est}_t) is less than 0.5, the model considers the input as a non-dense crowd image and passes it to the sparse DAD branch. Otherwise, it passes it to the high DAD branch, as shown in Fig. 2.

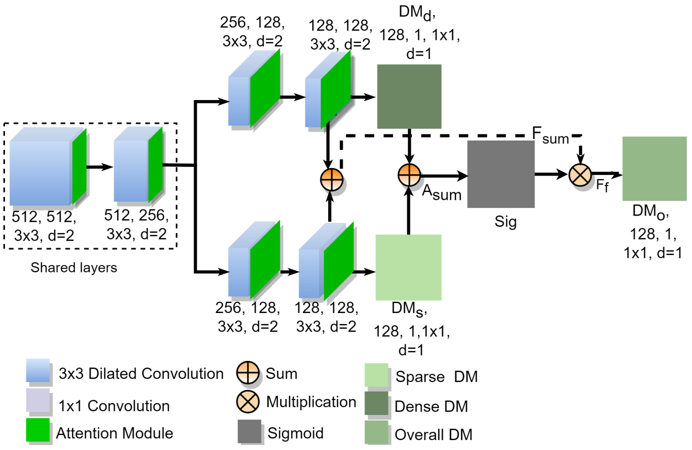


Figure 6: The illustration of the DAD module. The input feature maps are fed to the two shared layers and then we use the two branches with three convolution layers to handle the dense and sparse areas within the scene.

3.5. Density Aware Decoder (DAD)

DAD is one of the special modules of our proposed PDANet model, as it dynamically handles intra-variation of the density level within the input image. To achieve this, we use four dilated convolution layers with the attention module attached to each layer, similar to the one introduced in Section 3.2. According to the result of the classification module, we pass the output of the PFE module (F^p_{out}) to one of two DAD modules. If the input scene is highly crowded, we direct F^p_{out} to the high DAD branch. Otherwise, we pass it to the sparse branch. We achieve a model that can address the density variation of among different input image adaptively. Furthermore, the DAD module by itself is composed of two parts, *i.e.*, the shared layers, and the low or high-density decoder branches. This design enables us to cope with various occlusions, internal changes, and diversified crowd distributions within every single input scene, as illustrated in Fig. 1.

The structure of DAD is illustrated in Fig. 6. As shown in the figure, we consider the first two layers as shared layers and then pass the output feature map along two separate paths with the other three convolution layers to manage the within-image density variation, as shown in Fig. 1. The number of channels in the dilated convolution in DAD is ($N_{ch} = 512, 256, 128, 128, 1$) with the kernel filter size 3 × 3 and the dilation rate $d_{rate} = 2$ for the first four layers and 1 × 1 convolution at the end to produce the density maps. We call the outpot of dense and sparse branches as DM_d and DM_s, respectively. Furthermore, to reduce the number of training parameters, we utilize a 1×1 convolution to reduce the input channels to 32 and then perform a 2D dilated convolution on the reduced channel feature maps. This process speeds up the training and convergence of our model.

Moreover, there is a small notation for the dense and non-dense crowded areas. We use the DM_d for the high dense regions within the image. However, for the low density regions, within a low or highly dense input image, we use a shared DM$_s$ layer. This design gives us the benefit of using more information to train the model to map the low and dense regions with the input image. Therefore, we are able to have a better density estimation for the low crowded areas. On the other hand, by utilizing a different DM$_d$ for the highly dense areas within the input image, our DAD module is able to improve its estimation for these areas too.

By utilizing this architecture in the DAD, we will have two resultant density maps for the low and high crowded areas of the input image. Besides this, we pick up these feature maps of the last layer in the dense and non-dense branches. Then, we sum up these feature maps to form an attention module A_{sum}, and name the summation as F_{sum}. Therefore, we use the following equation to produce the final overall feature map:

$$F_f = F_{sum} \times \text{Sig}(A_{\text{sum}}), \tag{6}$$

where $\text{Sig}(A_{\text{sum}})$ is the sigmoid scaling of A_{sum}, and F_f is the final overall feature map, which is fed to the final layer to produce an overall dense map.

4. Implementation Details

The last part of PDANet is about the loss function. The PDANet uses two types of losses, *i.e.*, the regression loss and the classification loss. We explain them in details in this sections.

4.1. Regression Loss and Ground Truth

For the regression loss, we utilize a combination of three different error measurements, i.e., counting error ℓ^c, various scale error ℓ^2, and escalated error ℓ^{es}, respectively.

We measure the counting error ℓ^c as an absolute difference between ground-truth and the estimated crowd count, with the following equation:

$$\ell^c = \left| \sum_{i=1}^{N} D_i^{gt} - \sum_{i=1}^{N} D_i^{est} \right|, \tag{7}$$

where N is the number of pixels in an input scene, D_i^{gt}, and D_i^{est} are the ground-truth and the estimated crowd count at location i for $i = 1, 2, \cdots, N$, respectively.

We use the same methodology as in other works to obtain the ground-truth density map D_i^{gt} [35], which is generated by convolving each delta function $\delta(x - x_i)$ with a normalized Gaussian kernel G_σ [35] as:

$$D_i^{gt} = \sum_{x \in S_I} \delta(x - x_i) \times G_\sigma(x), \tag{8}$$

where S_I represents the number of annotated points in the image I, and x_i is the i-th annotated point. Instead of using the geometry-adaptive kernels [34] in Eq. 8, we use a fixed spread parameter σ of the Gaussian kernel for generating ground truth density maps. Also note the summation of the density maps (D_i^{gt}, $i = 1, 2, \cdots, N$) is equal to the crowd count in the image.

For the proposed PDANet, we need to separate the sparse and dense regions within the input scene to extract the dense and sparse regression losses for each input image. To obtain $D^{gt}_{d,i}$ or $D^{gt}_{s,i}$ representing the density map at location i falling into a dense or sparse region in the input image, we utilize a simple rule, which is defined by:

$$D^{gt}_{d,i} = \begin{cases} \text{D}_i^{gt}, & \text{if } D_i^{gt} > mean\{D_i^{gt}, i = 1, 2, \cdots, N\}, \\ 0, & \text{else.} \end{cases} \tag{9}$$

$$D^{gt}_{s,i} = \begin{cases} \text{D}_i^{gt}, & \text{if } D_i^{gt} <= mean\{D_i^{gt}, i = 1, 2, \cdots N\}. \\ 0, & \text{else.} \end{cases} \tag{10}$$

The various scale error ℓ^2 measures the pixel-wise errors of various scales. To obtain this error, we again utilize the Global Average Pooling (GAP) on the ground truth and the estimated density map. We apply three scales of GAP, *i.e.*, dividing each of the target and input density maps with divisors of 2, 4 and 8, respectively. For example, for the input density map with a size of $H \times W$, we apply GAP with a divisor of 2 to generate the corresponding GAPs' density maps with the size of $\frac{H}{2} \times \frac{W}{2}$. Then, we measure the ℓ^2 for each scale by:

$$\ell^2 = \sum_{i=1}^{N} \left| D_i^{gt} - D_i^{est} \right|^2, \tag{11}$$

where, for each $i \in \{1, 2, \cdots, N\}$, we use the same D_i^{gt} and D_i^{est} to represent the ground-truth and the estimated maps for each of the three different scales, respectively. We denote these three scale errors as ℓ^2_{s2}, ℓ^2_{s4} and ℓ^2_{s8}, respectively. These errors help us to accurately handle the density variation in each scale of the input scene.

The escalated error focuses mostly on addressing the areas with a high difference between the ground truth and the estimated density map. This error is defined as the absolute difference between the estimated density map and its ground truth density at each location i, as shown in Eq. 12 below, where $i = 1, 2, \cdots, N$.

$$\ell_i^1 = \left| D_i^{gt} - D_i^{est} \right|. \tag{12}$$

Then, we calculate the average difference ℓ^1 from ℓ_i^1, $i = 1, 2, \cdots, N$, by:

$$\ell^1 = \frac{\sum_{i=1}^{N} \ell_i^1}{N}. \tag{13}$$

Then, ℓ^1 is used to apply extra weight to the area with higher mis-estimation value to speed up the training process. We also force PDANet to generate escalated errors for regions with no people or objects. This is done by augmenting the corresponding ℓ_i^1 values by 10 times. This augmentation has two main benefits, *i.e.*, faster convergence and more accurate estimation in highly crowd areas. The escalated difference error ℓ_i^{es} at location i, for $i = 1, 2, \cdots, N$, and the overall escalated error ℓ^{es} are defined by:

$$\ell_i^{es} = \begin{cases} 10 \times \ell_i^1, & \text{if } \ell_i^1 >= \ell^1, \\ 10 \times \ell_i^1, & \text{else if } D_i^{gt} == 0, \\ \ell_i^1, & \text{else,} \end{cases} \tag{14}$$

and

$$\ell^{es} = \sum_{i=1}^{N} \ell_i^{es}. \tag{15}$$

Thus, the regression loss, denoted by ℓ_o^{reg}, can be written as:

$$\ell_o^{reg} = \ell^{es} + \ell^{c} + \ell_{s2}^2 + \ell_{s4}^2 + \ell_{s8}^2. \qquad (16)$$

When D_i^{gt} in Eqs. 7, 11, and 12, is replaced by $D_{d,i}^{gt}$ and $D_{s,i}^{gt}$, respectively, Eq. 16 defines the dense regression loss and the sparse regression, denoted by ℓ_d^{reg} and ℓ_s^{reg}, respectively.

4.2. Classification Loss

In our model, input images are classified into highly crowded scenes and less crowded scenes. We first define the actual class tag of an image as Cl_t^{gt} based on a rule to decide whether the image is highly crowded or not.

Let Dl_p^{gt} measure the ground-truth dense level of an input scene, which is defined by:

$$Dl_p^{gt} = \frac{\sum_{i=1}^{N} D_i^{gt}}{\sum_{i=1}^{N} \mathrm{sgn}(D_i^{gt})}, \qquad (17)$$

where $\mathrm{sgn}(\cdot)$ is a sign function.

Based on our statistic on the numbers of people contained in an image of each dataset, we find a threshold τ and use it to label an input image as a high or low density scene as:

$$Cl_t^{gt} = \begin{cases} 1, & \text{if } Dl_p^{gt} > \tau, \\ 0, & \text{else.} \end{cases} \qquad (18)$$

That is to say, if the dense level of the input scene Dl_p^{gt} is larger than the threshold τ, we consider it as a high density input scene; otherwise, it is a low density one. We tested different threshold values of τ, and found that our model is not sensitive to it and able to classify the input scenes correctly.

Then, we consider the Binary Cross Entropy (BCE) loss to train the model to classify input images into sparse and dense scenes, where BCE_{loss} is defined by:

$$BCE_{loss} = -\left[\mathrm{Cl}_t^{gt} \cdot \log \mathrm{Cl}_t^{est} + (1 - \mathrm{Cl}_t^{gt}) \cdot \log(1 - \mathrm{Cl}_t^{est})\right]. \qquad (19)$$

4.3. Total Loss

Finally, we need to define a rule to train the model efficiently by a combination of proposed losses. As it is obvious from the structure of the model, we need to detect and correctly pass high and sparse dense input to the corresponding DAD. Therefore, we need to penalize the model whenever it cannot classify the dense level of the input scene.

Thus, we define the following losses:

$$\mathrm{Sum}_{loss} = \ell_o^{reg} + \alpha \times (\ell_d^{reg} + \ell_s^{reg}) \qquad (20)$$

and

$$\mathrm{Final}_{loss} = BCE_{loss} \times \ell_o^2 + \mathrm{Sum}_{loss}, \qquad (21)$$

where α is empirically set to 0.4.

According to the Final_{loss}, by adding the $BCE_{loss} \times \ell_o^2$, we are able to overcome the mis-classification of the input scene. With Sum_{loss}, the model can learn the dense and sparse area within an input image precisely.

5. Experiments

In this section, we evaluate the performance of our proposed approach. We first introduce the evaluation metrics and then report experimental results obtained on benchmark datasets. The experiments are conducted on four benchmark datasets, and results are compared with the recently published state-of-the-art approaches.

5.1. Evaluation Metrics

Previous works on crowd density estimation used the Mean Absolute Error (MAE) and the Root Mean Squared Error (MSE) as evaluation metrics [35, 34, 11, 1], which are defined by:

$$\mathrm{MAE} = \frac{1}{M} \sum_{m=1}^{M} \left| C_m^{est} - C_m^{gt} \right| \qquad (22)$$

and

$$\mathrm{MSE} = \sqrt{\frac{1}{M} \sum_{m=1}^{M} \left(C_m^{est} - C_m^{gt} \right)^2}, \qquad (23)$$

where M is the number of testing images, C_m^{gt} denotes the exact number of people inside the ROI of the m-th image and C_m^{est} is the corresponding, estimated number of people. In the benchmark datasets discussed below, the ROI is the whole image unless otherwise explicitly stated. We follow the methodology in [35] to prepare the ground truth density data. Note that the number of people in an image can be calculated by the summation over the pixels of the ground truth (D_i^{gt}) as it is defined in Eq. 8 and the predicted density maps (D_i^{est}). We follow the methodology in [35] to prepare ground truth density data.

5.2. Data Augmentation

We take the benefit of data augmentation to avoid the risk of over-fitting to small numbers of training samples. We use five types of cropping alongside with a resizing as data augmentations. We crop each image into 1/4 of the original dimension. The first four cropped images extract four non-overlapping patches based on each corner of the original image. Furthermore, the fifth crop is randomly cropped from the input scene. For resizing, we resize the input images to the dimension of 768 × 1024 or 1024 × 768 depending on the aspect ratio of the images.

5.3. Experimental Results on the ShanghaiTech Dataset

The ShanghaiTech dataset [24] is one of the most popular and large-scale crowd counting datasets, which contains 1,198 annotated images with a total of 330,165 people. It contains two parts, *i.e.*, Part A (ShanghaiTech-A) with 482 images randomly collected from the Internet, and Part B (ShanghaiTech-B), including 716 images taken from the urban areas in Shanghai. As the challenge caused by the diversity of scenes and crowdedness level differs, it is difficult to estimate the number of pedestrians precisely. Following [35] and as mentioned in Section 5.1, for setting σ for Part A, we use the KNN method to calculate the average distance between each head and its three nearest heads and β is set to 0.3. For Part B, we set a fixed value 15 for σ. We compare our method with state-of-the-art methods recently published on this dataset.

Table 1: Comparison of the MAE and MSE results obtained with our proposed PDANet and the-state-of-the-art crowd counting approaches on the ShanghaiTech Part A and Part B [24], the WorldExpo10 [36], the UCF CC 50 [37] as high crowded datasets and UCSD as low crowded dataset [38], and two latest challenging crowd counting datasets; NWPU-Crowd [39] and JHU-CROWD++ [40] .

	ShanghaiTech A		ShanghaiTech B		WorldExpo10						UCF CC 50		UCSD		NWPU-Crowd		JHU-CROWD++	
Methods	MAE	MSE	MAE	MSE	Sce.1	Sce.2	Sce.3	Sce.4	Sce.5	AVG	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE
A-CCNN[10]	85.4	124.6	19.2	31.5	-	-	-	-	-	-	367.3	423.7	1.36	1.51	176.5	520.6	171.2	453.1
BSAD[41]	90.4	135.0	20.2	35.6	4.1	21.7	11.9	11.0	3.5	10.5	409.5	563.7	1.00	1.40	-	-	-	-
ACSCP[42]	75.7	102.7	17.2	27.4	2.8	14.05	9.6	8.1	2.9	7.5	291.0	404.6	1.04	1.35	-	-	-	-
D-ConvNet-v1[9]	73.5	112.3	18.7	26.0	1.9	12.1	20.7	8.3	2.6	9.1	288.4	404.7	-	-	-	-	-	-
IG-CNN[25]	72.5	118.2	13.6	21.1	2.6	16.1	10.15	20.2	7.6	11.3	291.4	349.4	-	-	-	-	-	-
DRSAN[43]	69.3	96.4	11.1	18.2	2.6	11.8	10.3	10.4	3.7	7.76	219.2	250.2	-	-	-	-	-	-
ic-CNN[44]	68.5	116.2	10.7	16.0	17.0	12.3	9.2	8.1	4.7	10.3	260.9	365.5	-	-		–	-	-
CSRNet[34]	68.2	115.0	10.6	16.0	2.9	11.5	8.6	16.6	3.4	8.6	266.1	397.5	1.16	1.47	121.3	387.8	85.9	309.2
SANet[35]	67.0	104.5	8.4	13.6	2.6	13.2	9.0	13.3	3.0	8.2	258.4	334.9	1.02	1.29			91.1	320.4
SFCN[45]	64.8	107.5	7.6	13.0	-	-	-	-	-	-	214.2	318.2	-	-	105.4	424.1	77.5	297.6
TEDnet[46]64.2	109.1	8.2	12.8	2.3	10.1	11.3	13.8	2.6	8.0	249.4	354.5	-	-	-	-	-	-	
HA-CCN[22]	62.9	94.9	8.1	13.4	-	-	-	-	-	-	256.2	348.4	-	-	-	-	-	-
SPN[11]	61.7	99.5	9.4	14.4	-	-	-	-	-	-	259.2	335.9	1.03	1.32	-	-	-	-
SPN+L2SM[11]	64.2	98.4	7.2	11.1	-	-	-	-	-	-	188.4	315.3	1.03	1.32	-	-	-	-
ADCrowdNet[18]	63.2	98.9	7.7	12.9	**1.7**	14.4	11.5	7.9	3.0	7.7	257.1	363.5	0.98	1.25				
PACNN+CSRNet[28]	62.4	102.0	8.9	13.5	2.3	12.5	9.1	11.2	3.8	7.8	267.9	357.8	**0.89**	**1.18**	-	-	-	-
CAN[1]	62.3	100.0	7.8	12.2	2.9	12.0	10.0	7.9	4.3	7.4	212.2	243.7	-	-	106.3	**386.5**	-	-
DENet[6]	65.5	101.2	9.6	15.4	2.8	10.7	**8.6**	15.2	3.5	8.2	241.9	345.4	1.05	1.31	-	-	-	-
PaDNet[23]	59.2	98.1	8.1	12.2	-	-	-	-	-	-	185.8	278.3	0.85	1.06	-	-	-	-
DM-Count[47]	59.7	95.7	7.4	11.8	-	-	-	-	-	-	211.0	291.5	-	-	**88.4**	388.6	-	-
CG-DRCN-CC-Res101[40]	60.2	94.0	7.5	12.1	-	-	-	-	-	-	-	-	-	-	-	-	**71.0**	**278.6**
PDANet	**58.5**	**93.4**	**7.1**	**10.9**	1.8	**9.1**	9.6	**7.3**	**2.2**	**6.0**	**119.8**	**159**	0.93	1.21	89.9	387.4	75.6	284.8

The quantitative results for ShanghaiTech-A are listed in Table 1. We collect results of the state-of-the-art approaches from their original published papers. It can be seen that our PDANet has achieved an MAE of 58.5 and an MSE of 93.4. Our proposed method also exhibits significant advantages over many top ranked methods such as PaDNet [23], ADCrowd-Net [18], HA_CNN [22], and SPN [11].

On the ShanghaiTech-B dataset, our proposed PDANet has achieved an MAE of 7.1 and an MSE of 10.9, both are better than those of the state-of-the-art results. These results suggest that our proposed PDANet is able to cope with sparse and dense scenes, thanks to the combination of the pyramid module as mentioned in Sect. 3.3 and the two-branch DAD as described in Sect. 3.5. Because of these, our proposed model can distinguish the crowd level of the input scene and analyze the crowd accordingly for better estimation.

5.4. *Experimental Results on the WorldExpo10 Dataset*

The WorldExpo10 dataset [36] is another large-scale crowd counting benchmark dataset. During the Shanghai WorldExpo 2010, 1,132 video clips were captured by 108 surveillance cameras to produce this large dataset. We follow the standard procedures [36] to do experiments on this dataset. Table 1 also provides MAE results based on five different scenes. The best-performing state-of-the-art methods are CAN [1], ADCrowd-Net [18], and PACNN [28] with an average MAE less than 8. However, as shown in the table, our proposed PDANet has achieved an average MAE of 6.0, which suppresses the-state-of-the-art results with a margin of 1.4 over the results achieved by CAN [1]. Furthermore, our PDANet yields the lowest MAE in 4 out of all 5 scenes with an MAE values equal to 1.8, 9.1, 7.3, and 2.2, respectively. As it is demonstrated, the overall performance of our PDANet across various scenes is superior compared with the-state-of-the-art approaches.

5.5. *Experimental Results on the UCF Dataset*

The UCF CC 50 [37] is one of the most challenging data sets in crowd counting research area due to its limited number of training images and significant variation in the number of people within the datasets (from 94 to 4,543 across images). We choose the setting similar to the ShanghaiTech-A [24] setting for generating ground truth density maps. Table 1 shows that our PDANet outperforms the state-of-the-art models by a large margin. We achieve an MAE of 119.8 with an MSE of 159, which is about 35 percent better than PaDNet [23], the best-performing benchmark model. In our experiments, we observe that our PDANet is able to estimate the number of people accurately in all subsets.

5.6. *Experimental Results on the UCSD Dataset*

The UCSD dataset [38] is another dataset that we conduct experiments on. In the experiments, we use Frames 601 through 1400 for training and the remaining out of 2000 for testing. Table 1 shows the MAE and MSE results obtained on this dataset. Compared with nine currently best approaches tested on this low crowed dataset, the proposed PDANet achieves the second best results with an MAE of .93 and an MSE of 1.21, which are very close and very comparative to the best results from the PaDNet model. We believe that the results is good enough taking into account that extreme resizing (image size in the UCSD dataset is 238 × 158) is needed for PDANet as we mentioned in Sect. 5.2 to work in our model.

5.7. *Experimental Results on the NWPU-Crowd Dataset*

We conduct some additional experiments on a new crowd counting dataset, *i.e.*, NWPU-Crowd [39]. The NWPU-Crowd dataset consists of 2,133,375 annotated instances for the total 5,109 images. This dataset has some advantages in comparison with other datasets such as large appearance variation, high resolution, fair evaluation and containing negative samples. Due to that this dataset was released only recently, there are not many crowd counting models that have reported their results based on this dataset. As shown in Table 1, we can see that the recently proposed crowd counting solution DM-Count [47] has achieved the best results on this dataset in term of MAE and MSE. However, our PDANet solution is the second best model in terms of MSE 387.4, which is among the three best models in terms of MAE and MSE. This result again proves that our PDANet detects and handles the intra-density variation the same as best crowd counting solutions.

5.8. *Experimental Results on the JHU-CROWD++ Dataset*

Finally, we also conduct experiments on another new dataset, JHU-CROWD++ [40]. The JHU-CROWD++ dataset [40] includes 4,372 images with various density levels in different conditions such as illumination variations and weather-based degradation. Table 1 shows that our proposed PDANet achieves the MAE of 75.6 and MSE of 284.8, and is the second best model after the CG-DRCN-CC-Res101 [40].

Overall, it can be concluded that our proposed PDANet can work well in both sparse and dense scenarios. We will also explore the results in detail at the Ablation Study section as follows.

6. Ablation Study

To further demonstrate the effectiveness of each component proposed in our PDANet model, we conduct a series of ablation studies.

In this section, we first visualize some examples of the results achieved, and then explore some of our model components and discuss their outputs to analyze the effectiveness of each component. The ablation studies are conducted on the UCF CC50 [37] and the ShanghaiTech [24] datasets.

6.1. *Density Map Visualization*

Qualitatively, we visualize the density maps generated by our proposed PDANet method on the ShanghaiTech Part A, Part B [24], and UCF CC 50 [37] datasets in comparison with the original ground truth (GT). These are shown in Figs. 7, 8 and 9. In these figures, three sample images corresponding to low, medium and high density scenes are selected from each dataset. For each sample image, we show the input image with an index of 0, and its Ground Truth (GT) density map, its estimated overall density map, and its estimated dense and sparse density maps with an index of 1 to 4, respectively.

Fig. 7 presents some sample results obtained on the ShanghaiTech Part A dataset. For this dataset, we select three images with total crowd counts of 99, 582 and 2,270, respectively, to represent input scenes of low density (the top row of Fig. 7), medium density (the middle row of Fig. 7) and high density (the bottom row of Fig. 7). As shown in this figure, the estimated counts are very close to the actual ground truth counts, demonstrating that our proposed model performs well in the scenes with various crowdedness levels. For instance, for the image in the bottom row of Fig. 7, the ground truth count is 2,270, while our prediction is 2,213, which is a reasonable estimation for such a highly crowded scene. On the other hand, for low crowdedness scenes, such as Fig. 7(a0), our proposed PDANet also produces accurate density maps. Fig. 7 also shows that our proposed model can accurately discriminate more crowded areas from less crowded ones. When looking further into the results of dense and sparse scenes, we can draw a conclusion that our model works well for extracting better information for more accurate overall density map estimation.

Fig. 8 presents results on three sample scenes from the ShanghaiTech Part B dataset. In this figure, we choose three sample images with crowd counts varying from 29 to 251, corresponding to low, medium, and high crowdedness images. These figures also demonstrate that our PDANet works well in low crowdedness areas. For instance, in Fig. 8(a0), the predicted density map and the actual density map appear to be very similar, and so are the estimated count and the ground truth count of crowd in the scene. Fig. 8 also shows that for medium and low crowdedness scenes, our proposed model produces accurate density maps.

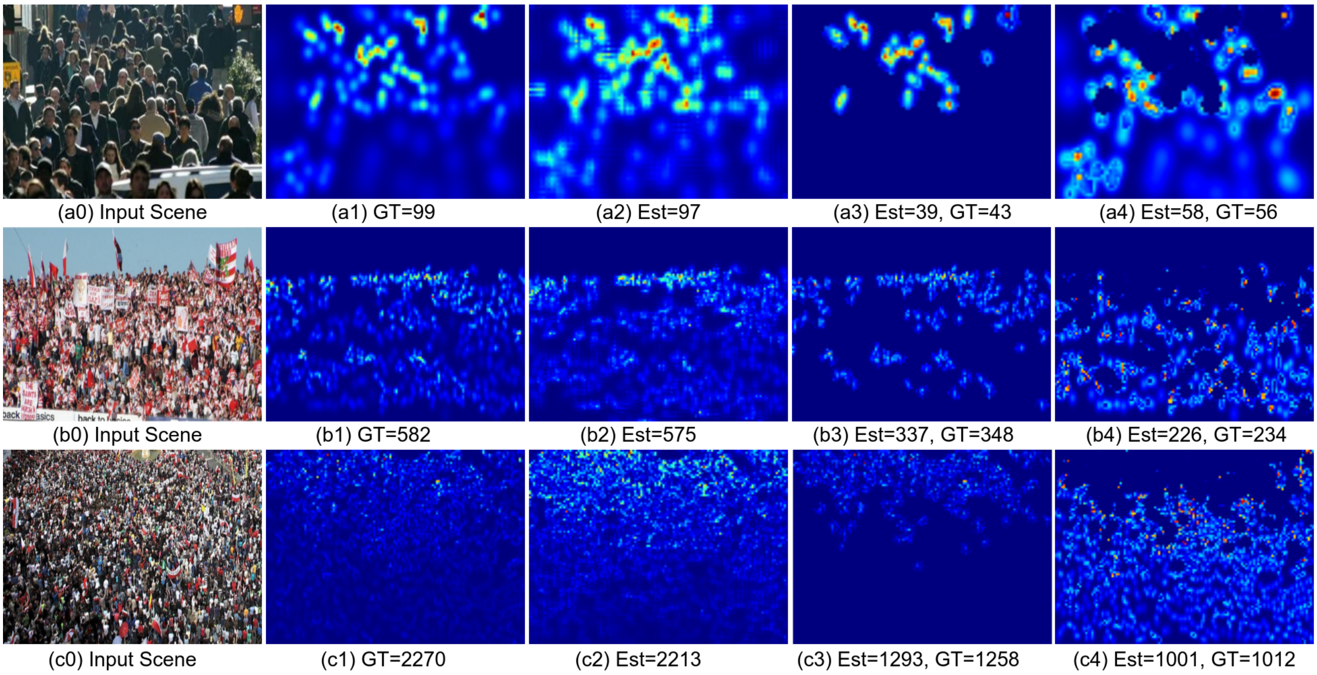
Fig. 9 illustrates the results on three sample images from the UCF CC 50 dataset [37], which is a highly crowded challenging dataset. In this figure, we choose three images with crowd counts equal to 555, 1,852, and 4,706, corresponding to low, medium, and high crowdedness scenes, respectively. We can see that our proposed PDANet works well in highly crowded images as well as low and medium crowdedness images. It is also evident that our proposed DAD model helps to localize dense and non-dense areas of the input image.

6.2. *Effectiveness of Loss Augmentation*

As discussed in Sect. 4.1 and specifically in Eq. 14, we escalate the ℓ_i^1 value by 10 times for the areas that have the higher misestimation values and the areas with the ground truth D_i^{gt} equal to zero. We have claimed that it speeds up the convergence and improves the estimation results. In this subsection, we provide the ablation study to demonstrate the effectiveness of the loss augmentation compared to the loss without the augmentation.

In term of convergence speed, according to empirical experiments on five parts of the UCF CC 50 dataset, the model converges before 100 iterations, but when we utilize the ℓ_i^1 loss without the augmentation, the model needs more than 150 iterations to converge. On the other hand, as shown in Figs. 7, 8 and 9, the model performs well in no-crowded areas, which we penalize with the augmented ℓ_i^1 loss.

In terms of performance improvement, we compare the effect of loss augmentation on the UCF CC 50 dataset. Tables 2 illustrates the results of PDANet trained with the loss augmentation (PDANet) against the PDANet trained without the augmented (ℓ_i^1) loss (PDANet_WOA). As shown in Table 2, we achieve better improvement in crowd density estimation by adopting the loss augmentation. Part0 has the highest improvement,


(a0) Input Scene (a1) GT=99 (a2) Est=97 (a3) Est=39, GT=43 (a4) Est=58, GT=56

(b0) Input Scene (b1) GT=582 (b2) Est=575 (b3) Est=337, GT=348 (b4) Est=226, GT=234

(c0) Input Scene (c1) GT=2270 (c2) Est=2213 (c3) Est=1293, GT=1258 (c4) Est=1001, GT=1012

Figure 7: Results of the estimated density maps of images from the ShanghaiTech Part A dataset. We illustrate three test images (a0, b0, c0), their actual ground truth (a1, b1, c1), our estimated overall density maps (a2, b2, c2), our estimated density maps for dense areas (a3, b3, c3), and our estimated density maps for sparse areas and their crowd counts (a4, b4, c4).

Table 2: Effect of ℓ_i^1 loss augmentation on crowd counting performance based on the UCF crowd-counting dataset [37]

	UCF CC 50						
	Metrics	Part0	Part1	Part2	Part3	Part4	AVG
PDANet_WOA	MAE	174	133	82	129	109	125.4
	MSE	217	189	99	191	132	165.6
PDANet	MAE	**157**	**128**	**80**	**126**	**108**	**119.8**
	MSE	**202**	**182**	**95**	**186**	**130**	**159**

which matches with our expectation to have better density estimation in high crowdedness areas.

6.3. Effectiveness of the PFE Module

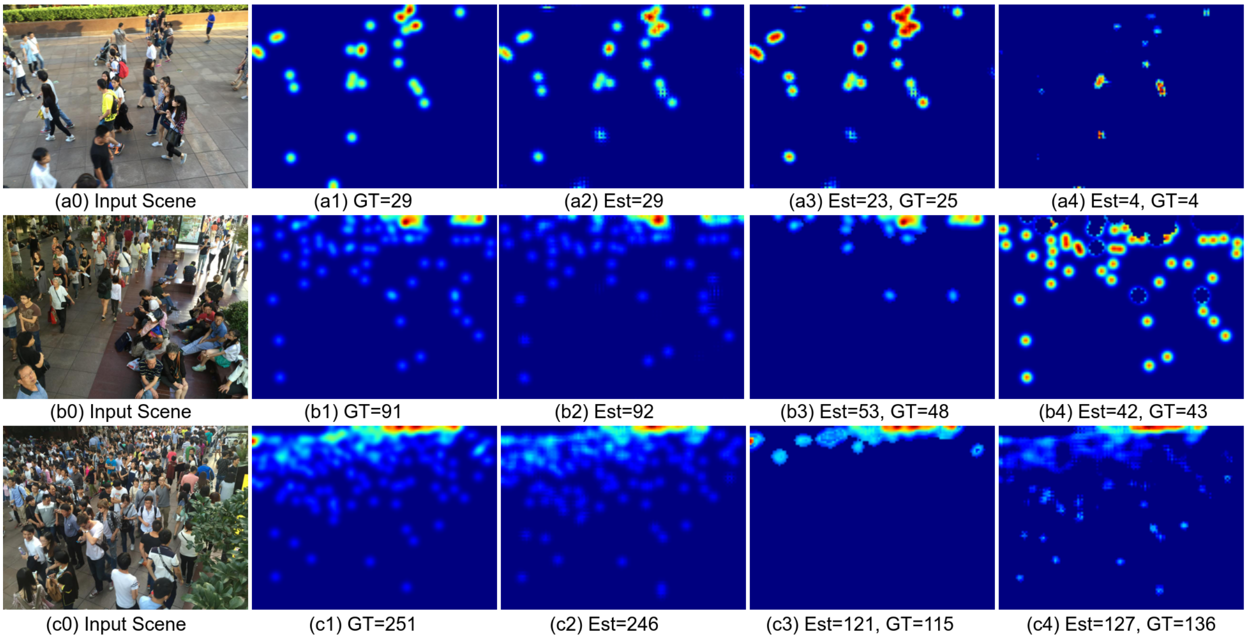
In the first experiment, we investigate the impact of different numbers of GAP modules on the baseline model (baselineAD, *i.e.*, a PDANet without the PFE module).

We test our proposed model with different numbers of GAPs from 0 GAP (baselineAD) to 10 GAPs. We obtain the GAPs of input feature maps by resizing them with divisors of 2, 4, 8, 3, 6, 10, 5, 7, and 9, respectively. For example, for the input feature map with size of $H \times W$, we apply GAP with a divisor of 2 to generate the corresponding GAPs' feature maps with the size of $\frac{H}{2} \times \frac{W}{2}$. We sort these numbers with the order of use. For example, if we aim to utilize 3 GAPs, we divide the input feature maps with the divisors of 2, 4, ..., respectively, to capture the information of various scales from the input feature maps.

Fig. 10(a) presents the results of this experiment on Part0 of the UCF CC 50 dataset. In this figure, we report the achieved MAE and MSE results for the PFE module with various GAPs. As shown in this graph, our PDANet has achieved an MAE of 157 and an MSE of 202 at three GAP settings (the third one in Fig. 10(a)), utilizing the division factors 2, 4, and 8, as three different scales of input feature maps. As it is shown, the proposed PDANet with this setting outperforms other PDANets with more or fewer GAPs modules, as well as the baselineAD model. Among the various PFE modules, PFEs with three GAPs (3GAP) and six GAPs (6GAP) provide better crowd level predictions in Part0. Fig. 10(a) also shows that the PDANet with GAPs in the worst-case still improves the estimation of the baselineAD (MAE of 202 vs. 300).

We test the effect of different numbers of pyramid GAPs on the ShanghaiTech dataset as well. The test results are shown in Fig. 10(b) and Fig. 10(c), for the ShanghaiTech PartA and partB, respectively. The results again show that our proposed PDANet (with three GAPs) outperforms the baselineAD and other PDANet models with more or fewer GAPs. The results of the other numbers of GAPs fluctuate slightly from an MAE of 58.5 to 65.7 for Part A and an MAE of 7.3 to 7.8 for Part B, which are very consistent.

In summary, the PFE with three GAPs works better in the PDANet model for crowd counting. By using the scales like the ones used in PDANet (*i.e.*, three GAPs or 3GAP), the output feature maps have more accurate scale information than those of the other PDANet models with different numbers of GAPs. On the other hand, increasing the number of GAPs will increase the number of parameters, which in turn increases the complexity of the model. Thus, the performance of the model will slightly decrease with the increase of over-fitting. Overall, our proposed PDANet (3GAP) has the most optimal number of parameters for the PFE modules. Thus, it can be trained more efficiently to capture the essential scale information.



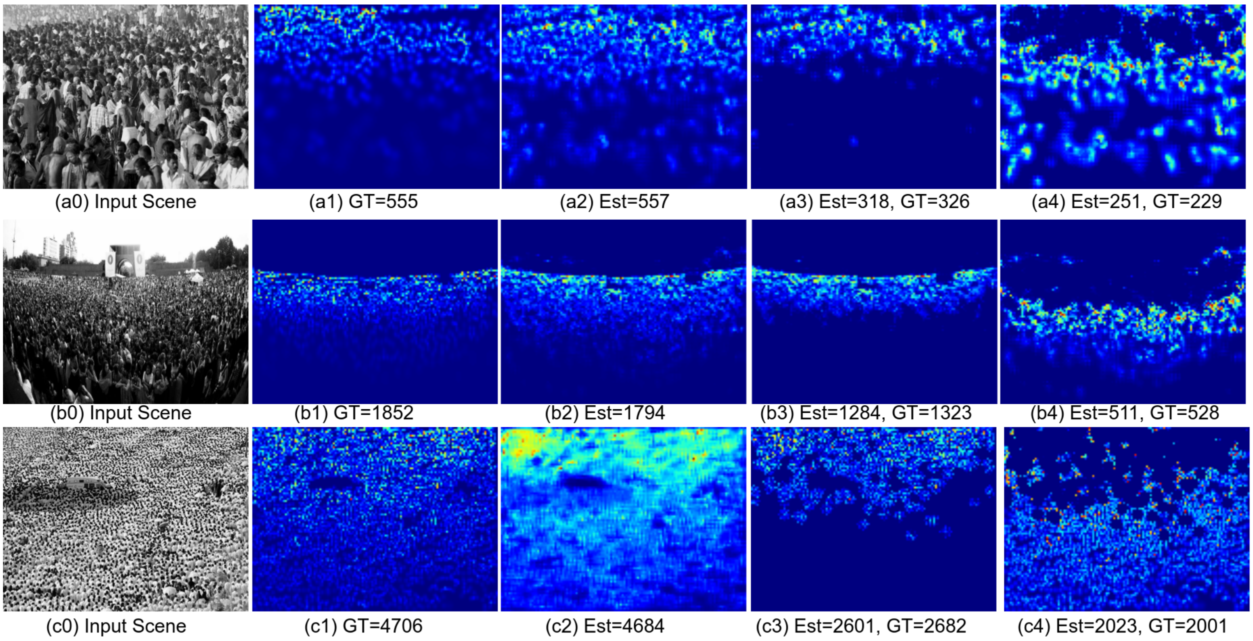
(a0) Input Scene (a1) GT=29 (a2) Est=29 (a3) Est=23, GT=25 (a4) Est=4, GT=4

(b0) Input Scene (b1) GT=91 (b2) Est=92 (b3) Est=53, GT=48 (b4) Est=42, GT=43

(c0) Input Scene (c1) GT=251 (c2) Est=246 (c3) Est=121, GT=115 (c4) Est=127, GT=136

t

Figure 8: Results of the estimated density maps of images from the ShanghaiTech Part B dataset. We illustrate three test images (a0, b0, c0), their actual ground truth (a1, b1, c1), our estimated overall density maps (a2, b2, c2), our estimated density maps for dense areas (a3, b3, c3), and our estimated density maps for sparse areas and their crowd counts (a4, b4, c4).



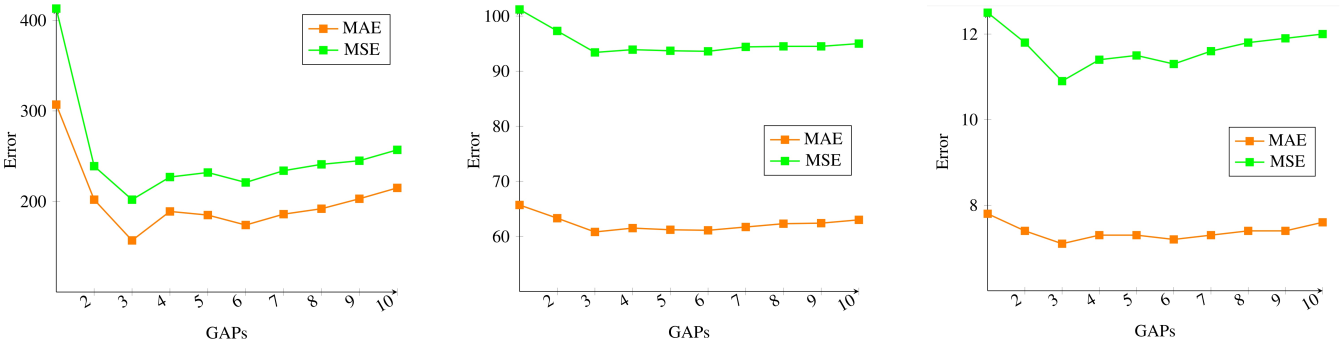
(a0) Input Scene (a1) GT=555 (a2) Est=557 (a3) Est=318, GT=326 (a4) Est=251, GT=229

(b0) Input Scene (b1) GT=1852 (b2) Est=1794 (b3) Est=1284, GT=1323 (b4) Est=511, GT=528

(c0) Input Scene (c1) GT=4706 (c2) Est=4684 (c3) Est=2601, GT=2682 (c4) Est=2023, GT=2001

t

Figure 9: Results of the estimated density maps of images from the UCF CC 50 dataset [37]. We present three test images (a0, b0, c0), their actual ground truth (a1, b1, c1), our estimated overall density maps (a2, b2, c2), our estimated density maps for dense areas (a3, b3, c3), and our estimated density maps for sparse areas and their crowd counts (a4, b4, c4).



(a) UCF CC 50 (b) ShanghaiTech PartA (c) ShanghaiTech PartB

Figure 10: Comparison of MAE and MSE results between various numbers of GAP layers on the UCF CC 50 crowd counting [37], the ShanghaiTech PartA, and the ShanghaiTech PartB datasets [24]

6.4. Effectiveness of the Attention Module

To gain an insight into the effectiveness of the Attention Module, we perform an ablation study to demonstrate the contribution of the module to the performance of the proposed model. We also compare our proposed attention modules with the two most common attention modules, *i.e.*, the Convolution Block Attention Module (CBAM) [48] and SENet [32] attention modules. We replace the proposed attention module in our PDANet model with the mentioned attentions, and name them as PDANet_CBAM and PDANet_SE, respectively. We compare the performance of our design choices with the baseline PFE and DAD module (BaselinePD), PDANet_CBAM and PDANet_SE.

Tables 3 and 4 illustrate the results obtained on the UCF CC 50 and the ShanghaiTech datasets. Part0 of UCF CC 50 dataset has the greatest improvement in terms of MAE/MSE, but the improvement on the performance of part1 to part4 is small. As shown in Table 4, we have achieved more or less the same improvement in crowd counting by adopting the proposed attention module. From these tables, we can find the proposed attention helps the PDANet perform slightly better than the PDANet_CBAM and PDANet_SE. It is expected due to that the proposed attention modules take advantages of both the spatial and channel attention with a parallel orientation to amplify the input feature map data and re-calibrate the crowded area within each input convolution layer.

Overall, we use the attention module for localizing the crowd area and improving the performance of our model. As shown in these tables, we achieve our goal by combining spatial/channel based attentions on both sparsely and densely crowded areas. Thus, these results prove the application of the attention module on improving the accuracy of the crowd counting model.

6.5. Effectiveness of the Classification and DAD Modules

To address the density variation within and between different input images, we proposed a two-branch DAD module. In this section, we aim to understand the effect of this module in our overall performance improvement. Like what was done in the previous sections, we compare the results of our PDANet with DAD and without DAD (passing the data to one branch only (Baseline PA)) on both UCF CC 50 and the ShanghaiTech datasets.

Table 3: Effect of adopting the attention module on crowd counting performance based on the UCF crowd-counting dataset [37]

	UCF CC 50						
	Metrics	Part0	Part1	Part2	Part3	Part4	AVG
BaselinePD	MAE	205	138	86	127	112	133.6
	MSE	243	198	111	189	135	175.2
PDANet_CBAM	MAE	187	128	81	126	109	126.2
	MSE	221	185	97	187	131	164.2
PDANet_SE	MAE	197	131	84	127	111	130
	MSE	235	191	106	187	133	170.4
PDANet	MAE	**157**	**128**	**80**	**126**	**108**	**119.8**
	MSE	**202**	**182**	**95**	**186**	**130**	**159**

Table 4: Effect of Attention Module on crowd counting performance based on the ShanghaiTech crowd-counting dataset [24]

	Metrics	ShanghaiTech	
		PartA	PartB
BaselinePD	MAE	62.3	7.3
	MSE	98.6	11.6
PDANet_CBAM	MAE	59.8	7.2
	MSE	95.3	11.1
PDANet_SE	MAE	60.2	7.3
	MSE	96.8	11.5
PDANet	MAE	**58.5**	**7.1**
	MSE	**93.4**	**10.9**

Table 5: Effect of classification and DAD modules on crowd counting performance based on the UCF crowd counting dataset [37]

	UCF CC50						
	Metrics	Part0	Part1	Part2	Part3	Part4	AVG
BaselinePA	MAE	217	151	116	146	114	148.8
	MSE	267	207	124	185	138	183.4
PDANet	MAE	**157**	**128**	**80**	**126**	**108**	**119.8**
	MSE	**202**	**182**	**95**	**186**	**130**	**159**

Table 6: Effectiveness of the classification and DAD modules on crowd counting performance based on the ShanghaiTech crowd-counting dataset [24]

	Metrics	Shanghai	
		PartA	PartB
BaselinePA	MAE	66.5	7.5
	MSE	104.1	12.6
PDANet	MAE	**58.5**	**7.1**
	MSE	**93.4**	**10.9**

Tables 5 and 6 show the experimental results obtained on the UCF CC 50 and the ShanghaiTech datasets, respectively. As seen from Table 5, we are able to boost the accuracy of crowd counting by about 20 percent for the UCF dataset in all subsets. With the ShanghaiTech dataset, we also achieve a noticeable improvement in accuracy with the help of the DAD module.

These results demonstrate the effectiveness of our initial idea about processing the sparsely and densely crowded feature maps separately. We believe that the DAD module helps the PDANet generate proper density maps for both high and low crowdedness areas in the images, and simultaneously, it guides the proposed model to react to the difference among input images with different crowdedness.

7. Conclusion

In this work, we have introduced a novel deep architecture called Pyramid Density-Aware Attention-based network (PDANet) for crowd counting. The PDANet incorporates pyramid features and attention modules with a density-aware decoder to address the huge density variation within the crowded scenes. The proposed PDANet has utilized a classification module for passing the pyramid features to the most suitable decoder branch to provide more accurate crowd counting with two-scale density maps. To aggregate these density maps, we took the benefit of the sigmoid function and produced a gating mask for producing the final density map. Extensive experiments on various benchmark datasets demonstrated the performance of our PDANet in terms of robustness, accuracy, and generalization. Our approach is able to achieve better performance on almost all of the major crowd counting datasets over the state-of-the-art methods, especially on the UCF CC 50 with more than 35 percent immediate improvements in the results. For the future research, we have two main suggestions. Firstly, to utilize zooming in the middle of crowd counting model to address the intra-dense areas within the input scene. Secondly, to perform patch-based processing in the middle feature maps of crowd counting model. Our initial investigation has proved that these two ideas can lead to further improvement in the accuracy of crowd counting techniques.

References

[1] W. Liu, M. Salzmann, P. Fua, Context-aware crowd counting, in: CVPR, 2019, pp. 5099–5108.

[2] M. Ling, X. Geng, Indoor crowd counting by mixture of gaussians label distribution learning, IEEE Transactions on Image Processing 28 (11) (2019) 5691–5701.

[3] J. Shao, K. Kang, C. Change Loy, X. Wang, Deeply learned attributes for crowded scene understanding, in: CVPR, 2015, pp. 4657–4666.

[4] A. B. Chan, N. Vasconcelos, Bayesian poisson regression for crowd counting, in: ICCV, IEEE, 2009, pp. 545–551.

[5] V. Lempitsky, A. Zisserman, Learning to count objects in images, in: NIPS, 2010, pp. 1324–1332.

[6] L. Liu, J. Jiang, W. Jia, S. Amirgholipour, M. Zeibots, X. He, Denet: A universal network for counting crowd with varying densities and scales, IEEE Transactions on Multimedia.

[7] H. Li, X. He, H. Wu, S. A. Kasmani, R. Wang, X. Luo, L. Lin, Structured inhomogeneous density map learning for crowd counting, arXiv.

[8] L. Liu, S. Amirgholipour, J. Jiang, W. Jia, M. Zeibots, X. He, Performance-enhancing network pruning for crowd counting, Neurocomputing.

[9] L. Zhang, Z. Shi, M.-M. Cheng, Y. Liu, J.-W. Bian, J. T. Zhou, G. Zheng, Z. Zeng, Nonlinear regression via deep negative correlation learning, TPAMI.

[10] S. Amirgholipour, X. He, W. Jia, D. Wang, M. Zeibots, A-ccnn: Adaptive ccnn for density estimation and crowd counting, in: ICIP, IEEE, 2018, pp. 948–952.

[11] X. Chen, Y. Bin, N. Sang, C. Gao, Scale pyramid network for crowd counting, in: WACV, 2019, pp. 1941–1950.

[12] D. Sam, S. Surya, R. Babu, Switching convolutional neural network for crowd counting, in: CVPR, Vol. 1/3, 2017, p. 6.

[13] D. B. Sam, N. N. Sajjan, H. Maurya, R. V. Babu, Almost unsupervised learning for dense crowd counting, in: AAAI, Vol. 27, 2019.

[14] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv.

[15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: CVPR, 2015, pp. 1–9.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: CVPR, 2016, pp. 2818–2826.

[17] D. Kang, A. Chan, Crowd counting by adaptively fusing predictions from an image pyramid, arXiv.

[18] N. Liu, Y. Long, C. Zou, Q. Niu, L. Pan, H. Wu, Adcrowdnet: An attention-injective deformable convolutional network for crowd understanding, in: CVPR, 2019, pp. 3225–3234.

[19] X. Wu, Y. Zheng, H. Ye, W. Hu, J. Yang, L. He, Adaptive scenario discovery for crowd counting, in: ICASSP, IEEE, 2019, pp. 2382–2386.

[20] Q. Wang, M. Chen, F. Nie, X. Li, Detecting coherent groups in crowd scenes by multiview clustering, IEEE transactions on pattern analysis and machine intelligence 42 (1) (2018) 46–58.

[21] Q. Wang, J. Gao, W. Lin, Y. Yuan, Pixel-wise crowd understanding via synthetic data, International Journal of Computer Vision (2020) 1–21.

[22] V. A. Sindagi, V. M. Patel, Ha-ccn: Hierarchical attention-based crowd counting network, TIP.

[23] Y. Tian, Y. Lei, J. Zhang, J. Z. Wang, Padnet: Pan-density crowd counting, IEEE Transactions on Image Processing.

[24] Y. Zhang, D. Zhou, S. Chen, S. Gao, Y. Ma, Single-image crowd counting via multi-column convolutional neural network, in: CVPR, 2016, pp. 589–597.

[25] D. Babu Sam, N. N. Sajjan, R. Venkatesh Babu, M. Srinivasan, Divide and grow: Capturing huge diversity in crowd images with incrementally growing cnn, in: CVPR, 2018, pp. 3618–3626.

[26] V. A. Sindagi, V. M. Patel, Generating high-quality crowd density maps using contextual pyramid cnns, in: ICCV, IEEE, 2017, pp. 1879–1888.

[27] D. Deb, J. Ventura, An aggregated multicolumn dilated convolution network for perspective-free counting, in: CVPR Workshops, 2018, pp. 195–204.

[28] M. Shi, Z. Yang, C. Xu, Q. Chen, Revisiting perspective information for efficient crowd counting, in: CVPR, 2019, pp. 7279–7288.

[29] J. Wan, W. Luo, B. Wu, A. B. Chan, W. Liu, Residual regression with semantic prior for crowd counting, in: CVPR, 2019, pp. 4036–4045.

[30] R. R. Varior, B. Shuai, J. Tighe, D. Modolo, Scale-aware attention network for crowd counting, arXiv.

[31] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: CVPR, 2017, pp. 2881–2890.

[32] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: CVPR, 2018, pp. 7132–7141.

[33] A. G. Roy, N. Navab, C. Wachinger, Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks, in: MICCAI, Springer, 2018, pp. 421–429.

[34] Y. Li, X. Zhang, D. Chen, Csrnet: Dilated convolutional neural networks for understanding the highly congested scenes, in: CVPR, 2018, pp. 1091–1100.

[35] X. Cao, Z. Wang, Y. Zhao, F. Su, Scale aggregation network for accurate and efficient crowd counting, in: ECCV, 2018, pp. 734–750.

[36] C. Zhang, H. Li, X. Wang, X. Yang, Cross-scene crowd counting via deep convolutional neural networks, in: CVPR, 2015, pp. 833–841.

[37] H. Idrees, I. Saleemi, C. Seibert, M. Shah, Multi-source multi-scale counting in extremely dense crowd images, in: CVPR, 2013, pp. 2547–2554.

[38] A. Chan, Z. Liang, N. Vasconcelos, Privacy preserving crowd monitoring: Counting people without people models or tracking, in: CVPR, IEEE, 2008, pp. 1–7.

[39] Q. Wang, J. Gao, W. Lin, X. Li, Nwpu-crowd: A large-scale benchmark for crowd counting and localization, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[40] V. Sindagi, R. Yasarla, V. M. Patel, Jhu-crowd++: Large-scale crowd counting dataset and a benchmark method, IEEE Transactions on Pattern Analysis and Machine Intelligence.

[41] S. Huang, X. Li, Z. Zhang, F. Wu, S. Gao, R. Ji, J. Han, Body structure aware deep crowd counting, TIP 27 (3) (2017) 1049–1059.

[42] Z. Shen, Y. Xu, B. Ni, M. Wang, J. Hu, X. Yang, Crowd counting via adversarial cross-scale consistency pursuit, in: CVPR, 2018, pp. 5245–5254.

[43] L. Liu, H. Wang, G. Li, W. Ouyang, L. Lin, Crowd counting using deep recurrent spatial-aware network, arXiv.

[44] V. Ranjan, H. Le, M. Hoai, Iterative crowd counting, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 270–285.

[45] Q. Wang, J. Gao, W. Lin, Y. Yuan, Learning from synthetic data for crowd counting in the wild, in: CVPR, 2019, pp. 8198–8207.

[46] L. Fiaschi, U. Köthe, R. Nair, F. Hamprecht, Learning to count with regression forest and structured labels, in: ICPR, Vol. 21, IEEE, 2012, pp. 2685–2688.

[47] B. Wang, H. Liu, D. Samaras, M. Hoai, Distribution matching for crowd counting, arXiv preprint arXiv:2009.13077.

[48] S. Woo, J. Park, J.-Y. Lee, I. S. Kweon, Cbam: Convolutional block attention module, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 3–19.

Saeed Amirgholipour received his master's degree in computing sciences from Isfahan university in 2009. Between 2011 to 2017 He was lecturer at Azad University. Now, he is a PhD student in University of Technology Sydney, Australia. His research interest includes computer vision, deep learning, crowd counting, video analytic, and image sentiment analysis.

Wenjing Jia received her PhD degree in Computing Sciences from UTS in 2007. She is currently a Senior Lecturer at the Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney (UTS). Her research falls in the fields of image processing and analysis, computer vision and pattern recognition.

Lei Liu received his dual PhD degree in Computing Sciences from University of Technology Sydney and Beihang University (BUAA), China in 2019, and his B.Sc and M.Sc Degrees in University of Science and Technology Beijing (USTB) in 2006 and 2014, respectively. His research interests are deep learning and computer vision.

Xiaochen Fan received his B.S. degree in computer science from the Beijing Institute of Technology, China, in 2013. He is currently a Ph.D. candidate at the School of Electrical and Data Engineering, University of Technology Sydney, Australia. His research interests include mobile/pervasive computing, deep learning, and IoT.

Dadong Wang is an adjunct Professor at the University of Technology, Sydney (UTS) and a Conjoint Associate Professor at the University of New South Wales (UNSW), a Principal Research Scientist & the leader of the CSIRO Quantitative Imaging Research Team, part of the CSIRO Data61. His main research interests include image analysis, computer vision, artificial intelligence, signal processing and software engineering. He has been developing automated image analysis solutions for scientific and industrial applications, with the aim of increasing both quality and quantity of information extracted from multidimensional image data.

Xiangjian He is a professor and the Leader of Computer Vision and Pattern Recognition Laboratory at the Global Big Data Technologies Centre (GB-DTC) at the University of Technology Sydney (UTS). He received his PhD degree in UTS in 1999 and has been with UTS since then. His research interests include image processing, network security, pattern recognition, computer vision and machine learning. He has received many competitive national or regional grants and he has many high quality publications in prestigious journals.