Deep Metric Learning for Crowdedness Regression

Qi Wang, *Senior Member, IEEE*, Jia Wan, and Yuan Yuan, *Senior Member, IEEE*

***Abstract*— Cross-scene regression tasks, such as congestion level detection and crowd counting, are useful but challenging. There are two main problems, which limit the performance of existing algorithms. The first one is that no appropriate congestion-related feature can reflect the real density in scenes. Though deep learning has been proved to be capable of extracting high level semantic representations, it is hard to converge on regression tasks, since the label is too weak to guide the learning of parameters in practice. Thus, many approaches utilize additional information, such as a density map, to guide the learning, which increases the effort of labeling. Another problem is that most existing methods are composed of several steps, for example, feature extraction and regression. Since the steps in the pipeline are separated, these methods face the problem of complex optimization. To remedy it, a deep metric learning-based regression method is proposed to extract density related features, and learn better distance measurement simultaneously. The proposed networks trained end-to-end for better optimization can be used for crowdedness regression tasks, including congestion level detection and crowd counting. Extensive experiments confirm the effectiveness of the proposed method.**

***Index Terms*— Deep learning, metric learning, regression, congestion detection, crowd counting.**

I. Introduction

CROSS-SCENE regression tasks such as congestion detection and crowd counting [1] draw a lot of attention because of their significance in real life. Since the world's population grows faster, the congested traffic and crowd of people have become serious problems. Many accidents resulting from traffic jams or large crowd of people are causing death every year. If the traffic status and crowd number can be automatically monitored with video surveillance [2], such accidents can be reduced a lot and the travel will be more efficient.

There are many works concentrating on the two tasks. Most works on congestion detection treat it as a classification problem that distinguishes congested videos into 2-5 levels. To detect congestion, these methods utilize the attributes of moving objects in a video as features. First of all, the key points or moving blobs are detected to represent the objects. Then their speed is calculated to represent the speed of the objects. With the number of moving objects and their speed as features, the traffic jams can be detected. Crowd counting is more popular than congestion detection since there are many crowd counting datasets released for the community. Most of them have accurate labels including the size of crowd and the position of each person. These datasets promote the research of crowd counting especially deep learning based methods. Crowd counting is a typical regression problem which aims to estimate the crowd size through analyzing the features of an image. In literature, many features based on size, shape, edge, key point and texture are designed. Recently, some deep learning based counting methods have shown great potential. However, the deep networks are hard to converge without the help of additional information such as density map.

Though the two tasks have been researched for years, they are still very challenging and can't be used in real applications. Since most existing methods are composed of two separate steps: feature extraction and regression, there are two problems might limit the performance of existing systems. The first one is the design of features. It is easy to construct a feature for a detector to detect congestion level or count people in only one scene. However, it is impossible to design different detectors for different scenes. Thus, the cross scene crowdedness analysis is more practical, which highly enhanced the difficulties of feature design. For example, the holistic features like texture might have large variations among different scenes, which makes the regressors hard to distinguish different congestion levels or crowd sizes. Another problem is that the feature extraction and regression are separated which makes these models hard to optimize. It can be improved by the works based on deep learning. In these methods, the end-to-end training generates better optimized models and achieves better performance than traditional methods. However, how to efficiently train a deep network is still a challenge. Thus, most previous works utilize additional information such as the density map calculated by the position of persons to guide the training procedure. But, the labeling of all persons in a crowd scene is very time consuming. Thus, all the published datasets contain only thousands of images. That is relatively small compared to popular classification datasets which contain millions of images.

In this paper, a deep network is proposed to extract high-level semantic representations. The training of the proposed network is guided by metric learning. In literature, metric learning [3] is used to learn a better distance measurement and to improve the performance of the classification applications like person re-identification [4], image recognition [5],

Manuscript received December 22, 2016; revised March 27, 2017; accepted May 8, 2017. Date of publication May 12, 2017; date of current version October 24, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61379094, in part by the Open Research Fund of Key Laboratory of Spectral Imaging Technology, Chinese Academy of Sciences, and in part by the Fundamental Research Funds for the Central Universities under Grant 3102017AX010. This paper was recommended by Associate Editor J. Lu. *(Corresponding author: Qi Wang.)*

Q. Wang is with the School of Computer Science and the Center for OPTical IMagery Analysis and Learning, and with the Unmanned System Research Institute, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: crabwq@gmail.com).

J. Wan and Y. Yuan are with the School of Computer Science and the Center for OPTical IMagery Analysis and Learning, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: jiawan1998@gmail.com; y.yuan1.ieee@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2017.2703920

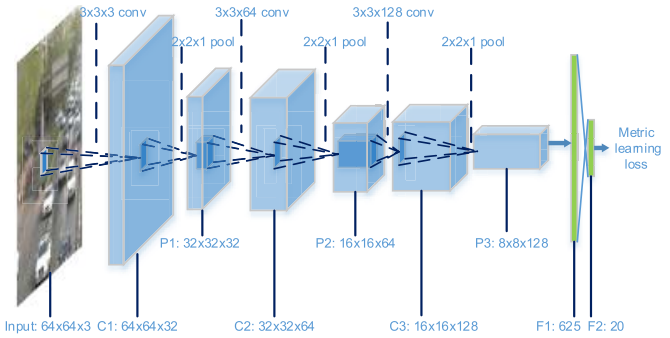

Fig. 1. The architecture of the proposed deep metric network. There are 3 convolutional layers and 2 fully connected layers. The metric learning loss is used to guide the training of the network. Note that, the training of this network is end-to-end. The learning of the feature representation and distance measurement are simultaneous.

tracking [6], cross-modal matching [7]. Since metric learning is useful to better reflect the relationships among samples, we believe that it will be effective to guide the training of deep networks with local structural information. The proposed method don't need the density map for training. The only label used for training is a number (congestion level or crowd size). Note that the training of the proposed model is end-to-end. Thus, the learning of the feature representation and distance measurement is performed at the same time. In this paper, we find that the metric learning is effective to guide the training of the deep network. And more effective distance measurement can also be learned with better features.

The contributions are summarized as follows:

1. A cross scene dataset is constructed for practical usage. This is a very challenging dataset which contains different illuminations, weathers, and road conditions. Some low resolution videos collected from real surveillance cameras are added into the dataset. That makes it closer to real scenarios.
2. A deep metric learning method is proposed for crowdedness regression tasks. The proposed model is a unified model which can be trained end-to-end for better optimization. This model can be effectively optimized through local structural embedding.
3. Extensive experiments are conducted on two crowdedness regression tasks including congestion detection and crowd counting. The experiments confirm that metric learning is effective to guide the training of the deep network. The proposed method is proved to be effective for crowdedness regression.

The reminder of this paper is as follows. Related works of congestion detection, crowd counting and metric learning are reviewed in Section II. The definition of congestion and the expanded dataset are described in Section III. The details of the proposed deep metric learning method are elaborated in Section IV. After the experimental results are reported and discussed in Section V, the conclusion and future works are presented in Section VI.

II. Related Work

In this section, the relevant works of congestion level detection, crowd counting and metric learning are reviewed.

A. Congestion Detection

The algorithms proposed for congestion detection can be divided into two classes. The first class detects congestion by analyzing the moving objects in videos. Another class is based on the density related feature extraction and classification.

The first class of algorithms are based on a simple assumption that more congested scenes have more moving objects. Hu *et al.* [8] propose a method which utilizes moving blobs to represent the moving objects. In detail, the moving blobs are first detected by background subtraction algorithm [9]. Then, the speed of moving objects is represented by the speed of the blobs which is calculated by Optical Flow [10]. Finally, the amount of blobs and their speed are utilized as features and the fuzzy logical is used for the final decision. Sobral *et al.* [11] further utilize key points to better represent the moving objects. In this method, the number of moving objects is represented by moving blobs which are detected by background subtraction and the speed is represented by the speed of key point which is calculated by Kanade-Lucas-Tomasi (KLT) algorithm [12]. These methods rely on the preprocessing algorithms like background subtraction and tracking which limit the performance.

Another class of algorithms are focus on the design of density related features [13] which can reflect congestion level efficiently. Derpanis and Wildes [14] propose a Spatial-temporal Orientation Analysis feature which encodes spatial and temporal information simultaneously. Riaz and Khan [15] propose to encode motion information through the statistics of motion vectors. Dallalzadeh *et al.* [16] propose a symbolic representation for congestion detection. These methods don't need preprocessing algorithms which work well for a specific scene. However, the accurate congestion level detection across different scenes is still challenging.

B. Crowd Counting

Crowd counting algorithms can be divided into two categories: holistic and local. Holistic approaches take the whole image as input to extract the features and then, a regression algorithm is utilized to map the feature to a crowd number. Local approaches take image patches as input and the final crowd size is summarized by the number of people in these patches.

Most holistic approaches utilize textures, foreground pixels and edges as features. Marana *et al.* [17] propose a Gray Level Cooccurrence Matrix (GLCM) based texture feature for crowd density estimation. Regazzoni *et al.* [18] propose to count crowd number through edge features. Cho *et al.* [19] propose to utilize background subtraction techniques to better model a human observer. These approaches are difficult to count the crowd size accurately due to the large variation of the crowd behavior.

The local approaches first divide the image into patches. Then, the crowd sizes of patches are estimated and accumulated as the final crowd size of the image. Chen *et al.* [20] propose to count crowd size through the feature extraction over the grid cells. Lempitsky and Zisserman [21] propose a method in which the crowd sizes of pixels are estimated

and summarized to form the crowd size of an image. Fiaschi *et al.* [22] propose to promote the regression by random forest. Chen *et al.* [20] propose to combine the local features and global features together. Recently, some deep learning based methods show huge potentials on this task. Zhang *et al.* [23] propose to count crowd size through a deep network which iteratively learns the density map and the global number. Zhang *et al.* [24] propose a multi-column convolutional neural network (MCNN) for crowd counting. However, the deep learning based methods always need additional information to guide the training of deep networks.

C. Metric Learning

Metric learning is used to learn a better distance measurement through training data. To achieve this, most algorithms is trying to minimize the distance between samples which come from the same class and maximize the distance of samples from different classes.

Most metric learning methods are applied to classification tasks such as person re-identification [25], [26], image recognition [27], [28], and image retrieval [29]. Large Margin Nearest Neighbor (LMNN) [30] is proposed to improve the performance of K nearest neighbors (KNN) classification. Information-Theoretic Metric Learning (ITML) [31] is proposed to regularize the distance matrix M to be closed to a predefined prior distribution. Neighborhood Component Analysis (NCA) [32] is a metric learning method which aims to improve the performance of leave-one-out validation. Discriminative Component Analysis (DCA) [33] is proposed to exploit the negative constraint.

Metric learning can also be used for regression. Metric Learning for Kernel Regression (MLKR) [34] is proposed to improve the performance of kernel regression. Kernel Regression with Sparse Metric Learning (KRSML) [35] is proposed to constrain the distance matrix M with a mixed (2,1)-norm. Xiao *et al.* [36] propose to estimate the human age through metric learning method.

Recently, some deep metric learning methods are proposed for classification tasks with the development of deep learning. Hu *et al.* [37] propose to compare the similarity of two face images through a deep metric learning model. Li and Tang [38] propose to learn a better distance measurement with the additional community-contributed images under the deep learning framework. Song *et al.* [39] propose a novel structured objective function for deep metric learning.

III. The Definition and Dataset

Since the congestion is not clearly defined, a unified and accurate definition of congestion is desired. In this section, the definition of congestion is first elaborated. Then, the details of the proposed dataset is presented.

A. Definition

The congestion is defined as space-time occupancy as shown in Figure 2. Generally speaking, the congestion level can be measured by two aspects: density and occupancy [40].

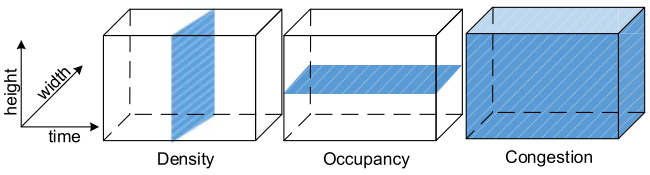

Fig. 2. The congestion is defined as space-time occupancy. The blue shadow area is where the congestion is calculated.

The density can only measure the congestion at a temporal point. The occupancy can only measure the congestion at a spatial point. The proposed congestion takes both spatial and temporal information into consideration. It has several properties:

1. The definition is accurate. Compared to the traditional methods which classify congestion videos into several rough levels, the proposed definition is more accurate, making it easier to predict the congestion status in the future.
2. The definition is universal. With the proposed definition, it is possible to compare the congestion level between different scenes. Thus, different cameras can cooperate together which makes congestion detection more practical.
3. The definition considers spatial and temporal information simultaneously. Neither density nor occupancy can reflect real traffic congestion status especially in complicated urban scenes. Consider density and occupancy together is more reasonable.

Formally, the proposed congestion is calculated as follows:

$$C_t = \frac{\sum_{x,y,\tau} f(x, y, \tau)}{w \times l \times t}, \tag{1}$$

where $C_t \in [0, 1)$ is the congestion level. w and l are the width and length of the road. (x, y) indicates a point on the road. t is a period of time and τ is a point of time in t. $f(x, y, \tau)$ indicates whether a specific point located at (x, y, τ) is occupied by a moving object. It is defined as:

$$f(x, y, \tau) = \begin{cases} 1, & \text{occupied} \\ 0, & \text{not occupied.} \end{cases} \tag{2}$$

The proposed definition can reflect the proportion of vehicles on a section of road in a period of time.

B. Dataset

Since existing datasets are not practical for real applications, a new dataset called NWPU_Congestion dataset is constructed. The dataset contains 26 different scenes including day and night, sunny and raining, and different road conditions. The resolutions of videos in this dataset vary from 352×288 to 1920×1080. Note that different directions in one video are treated as different scenes. Typical images in dataset are shown in Figure 3.

The labeling of congestion is time-consuming based on the definition since the calculation of congestion is pixel-wise.

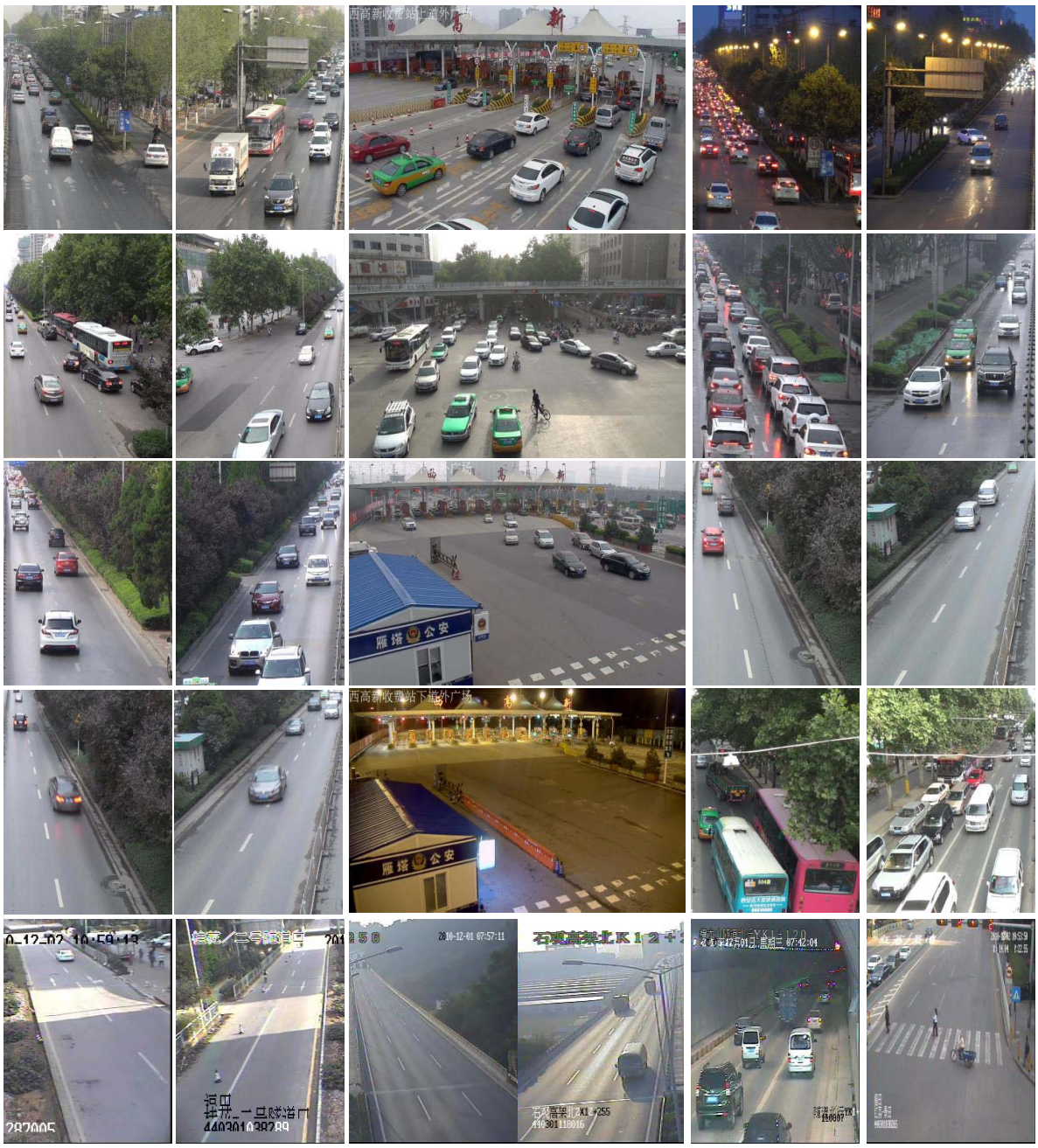

Fig. 3. Typical images of the NWPU_Congestion dataset. There are 26 different scenes in the dataset which contains different illumination and road condition. There are sunny and rainy weathers in this dataset and 3 scenes are recorded at night. Different scenes contain different lane numbers including 2 3 4 5 and complicated crossroads.

Thus, a simplified implementation is proposed with an assumption that the width of the vehicles and the width of a lane are equal. This assumption is satisfied in most cases since a lane is designated for a single line of vehicles. With this assumption, the calculation of congestion is reduced to:

$$C_t = \frac{\sum_{y,\tau} f(y,\tau)}{l \times t}. \tag{3}$$

In practice, the length of a lane is represented by a line along it and the length of vehicles can also be represented by some lines. Based on this, the perspective transformation is taken into consideration since it makes the vehicles far from the cameras smaller (shorter in length) in images. To remedy this, the vehicles far from the camera should have higher weights as shown in Figure 4 in which the color bar indicates the different weights (red indicates high weight). To calculate the weights of a scene, the width of road is employed since it is also affected by the perspective transformation. Specifically, the weights are inversely proportional to the widths of road at different positions (larger width causes smaller weight). In some particular scenes where the road boundary is not clear, the widths of vehicles at different positions are used instead of the widths of road.

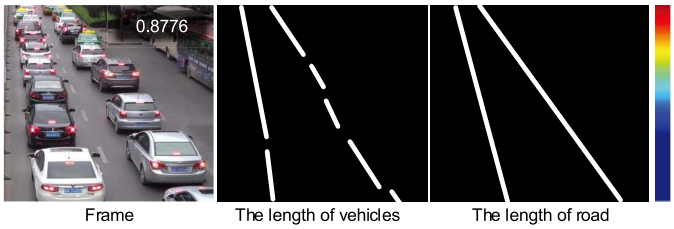

Fig. 4. The left image is a typical frame in traffic video. The number at the top-right is the congestion level. The middle image is the corresponding labeling image. In this image, the white lines are used to represent the length of vehicles. The white lines in the right image are used to represent the length of road. The color bar is the weight of the pixels in lines.

IV. Deep Metric Learning for Regression

To learn high-level density related features, the deep network is utilized for feature extraction. Though regression has been researched for years and can be used in many different applications [41], it often need careful design because it is hard to be optimized. So, the performance of deep regression model without careful design is rather poor in our experiments as shown in Section V. To remedy this, metric learning is utilized to guide the learning of the proposed deep model. The training of this network is end-to-end, and the learning of features and distance measurement are simultaneous. In this section, the basic of metric learning for regression is first presented. Then, the details of deep metric learning model are elaborated.

A. Metric Learning for Kernel Regression

Metric learning for kernel regression [34] aims to improve the performance of kernel regression by learning a better distance measurement between samples. The prediction of kernel regression can be seen as the weighted sum of training labels. The weights are decided by the distances between the test sample and other samples. The distance measurement can be learned by metric learning algorithms.

Formally, given training examples $x_1, x_2, ..., x_n$ and the corresponding labels $y_1, y_2, ..., y_n$, the prediction of x_i is written as:

$$\hat{y_i} = \frac{\sum_{j \neq i} y_i k_{ij}}{\sum_{j \neq i} k_{ij}}, \tag{4}$$

where k_{ij} is the kernel function indicating the weight between sample x_i and x_j, and is computed as:

$$k_{ij} = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{d_{ij}}{\sigma}), \tag{5}$$

where d_{ij} is the learned distance between two samples. Note that δ is set as 1 for simplification. The distance between two samples is calculated as:

$$\begin{aligned} d_{ij} &= (f(x_i) - f(x_j))^\top M (f(x_i) - f(x_j)) \\ &= (f(x_i) - f(x_j))^\top L^\top L (f(x_i) - f(x_j)) \\ &= \|L(f(x_i) - f(x_j))\|_2^2, \end{aligned} \tag{6}$$

where M is the distance matrix and L is the transformation between feature space and learned space and $f(x)$ indicates the feature of x_i. To learn the distance matrix M, the mean squared error is used as the loss function:

$$\mathcal{L} = \sum_i (y_i - \hat{y}_i)^2 + \beta(\|L\|_F^2 + \sum_i (\|W^{(i)}\|_F^2 + \|b^{(i)}\|_2^2)). \tag{7}$$

Since $f(x)$ is an nonlinear transformation and then the distance of two images is calculated after transforming the original images to an nonlinear feature space, we can treat the proposed method as an nonlinear extension of metric learning.

B. Deep Metric Learning Model

In previous works, the deep learning based methods need additional information (e.g. density map) to guide the training. However, the generation of density maps need large efforts to label the human heads. Thus, a deep metric learning model is proposed for weak labels (only congestion level or crowd size) in this paper. This model merges the feature learning and metric learning together into one network and the training is end to end. To save the memory and accelerate the training of the network, a small database is employed, and then the locality is utilized to further accelerate the training.

The proposed deep network (DeepNet) contains 3 convolutional layers and 2 fully connected layers. The first convolutional layer has 32 $3\times3\times3$ filters and the second convolutional layer has 64 $3\times3\times32$ filters. The last convolutional layer has 128 $3 \times 3 \times 64$ filters. After each convolutional layer, a max pooling layer is followed. The input of the network is 64×64 and the output of the network is a feature vector of length 20. The input image is first resized to 64×64 and then fed into the network to generate the representation.

Specifically, given an image x as an input, the output of the first layer is:

$$h^{(1)} = r(W^{(1)}x + b^{(1)}), \tag{8}$$

where $W^{(1)}$ is the projection matrix of the first layer and $b^{(1)}$ is the bias vector. $r(x)$ is the activation function defined in Equation 13. Similarly, the output of the n-th layer is:

$$h^{(n)} = r(W^{(n)}h^{(n-1)} + b^{(n)}), \tag{9}$$

where $W^{(n)}$ is the projection matrix of the n-th layer and $b^{(n)}$ is the respective bias vector. In the end, the output of the final layer (i.e., the 4-th layer) $f(x)$ is the deep feature:

$$f(x) = h^{(4)} = r(W^{(4)}h^{(3)} + b^{(4)}). \tag{10}$$

To effectively train the proposed network, we propose to guide it by metric learning. That is to say, the training of the network can be performed by minimizing the loss function defined in Equation 7.

However, it is impossible to feed all the training images to the network at the same time since the memory of GPU is limited. Thus, a subset X_{base} containing s samples is randomly selected from the training set and serves as the database. The input images are compared with the images in the database. Then, Equation 4 is reduced to:

$$\hat{y_i} = \frac{\sum_{x_j \in X_{base}} y_i k_{ij}}{\sum_{x_j \in X_{base}} k_{ij}}, \tag{11}$$

With this improvement, the training of deep networks becomes possible, and the performance does not drop through the experiments in Section V.

Another problem in kernel regression is that the exponential calculation is very time-consuming. To remedy this, k nearest neighbors instead of all the samples in the database are employed for the prediction. In particular, Equation 11 can be further reduced to:

$$\hat{y}_i = \frac{\sum_{j \in N(i)} y_i k_{ij}}{\sum_{j \in N(i)} k_{ij}}, \tag{12}$$

where $N(i) \in X_{base}$ is the set of neighbors of the sample x_i. After k nearest neighbors are selected, the number of exponential calculation is reduced from n to k where n is the training number and k is the number of neighbors. That accelerates the training of the proposed deep network since $k \ll n$. Note that, this method can be further accelerated by employing superior Nearest Neighbor Search method [42], [43]. To cope with large-scale data, graph based methods [44], [45] can be utilized to accelerate the training procedure.

The architecture of the proposed method is shown in Figure 1. This model is very effective to learn better features and distance measurement at the same time, since the metric learning can guide the network to learn better parameters.

Since the training set is relatively small, data augmentation is utilized to expand the training data. Specifically, all images are rotated -10 and 10 degrees respectively. Then, the size of generated new training set is 2 times larger than the original training set. Note that the testing set is not augmented.

C. Implementation

In this section, the details about implementations of the proposed method are elaborated. Note that, TensorFlow [46] package is employed to develop the proposed model.

1) Activation Function: There are many activation functions that can be used to define the output of a node. The rectified linear unit (ReLU) is utilized as activation function in this paper since ReLU is efficient to compute and propagate. It is defined as follows:

$$f(x) = max(0, x). \tag{13}$$

2) Initialization: The initialization is very important to the gradient decent based training. Following the setting in [47], b is initialized as 0 and w is initialized as a uniform distribution as follows:

$$w \backsim U\left[-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right], \tag{14}$$

where d is the input dimension.

3) Optimization: Given training images X_{train} and the corresponding labels Y_{train}, mini-batch gradient descent is utilized for optimization. In particular, we train the network for $n = 500$ epochs with the learning rate $\alpha = 0.001$. In each iteration, a mini batch of data X_{batch} and Y_{batch} is used for training. The calculation of $\frac{\partial \mathcal{L}}{\partial w}$, $\frac{\partial \mathcal{L}}{\partial w}$ and $\frac{\partial \mathcal{L}}{\partial w}$ is performed by symbolic differentiation provided by TensorFlow. The learning procedure of this network can be summarized in Algorithm 1.

Algorithm 1 Deep Metric Learning

Input: $X_{base} \in \Re^{n \times 64 \times 64 \times 3}$: the images in database
$Y_{base} \in \Re^{n \times 1}$: the labels of X_{base}
$X_{train} \in \Re^{n \times 64 \times 64 \times 3}$: the training images
$Y_{train} \in \Re^{n \times 1}$: the labels of X_{train}
$n \in \Re$: the number of epochs
$\alpha \in \Re$: the learning rate

Initialize $W^{(i)}$, $b^{(i)}$ according to Equation 14
// For each training epoch
for $i = 1, 2, ..., n$ **do**
  for $[X_{batch}, Y_{batch}]$ in $[X_{train}, Y_{train}]$ **do**
    // Forward propagation
    Calculate $f(X_{base})$ through Equation 8-10
    Calculate $f(X_{batch})$ through Equation 8-10
    // Compute the loss
    Calculate $\mathcal{L}$ according to Equation 7
    // Compute gradients
    Calculate gradients $\frac{\partial \mathcal{L}}{\partial L}$, $\frac{\partial \mathcal{L}}{\partial W^{(i)}}$ and $\frac{\partial \mathcal{L}}{\partial b^{(i)}}$
    // Update parameters
    $L = L - \alpha \frac{\partial \mathcal{L}}{\partial L}$
    $W^{(i)} = W^{(i)} - \alpha \frac{\partial \mathcal{L}}{\partial W^{(i)}}$
    $b^{(i)} = b^{(i)} - \alpha \frac{\partial \mathcal{L}}{\partial b^{(i)}}$
  end for
end for

Output: The deep model parameterized by $W^{(i)}, b^{(i)}$ and the learned linear transformation L.

V. Experimental Results

In this section, extensive experiments are conducted to confirm the effectiveness of the proposed method.

A. Evaluation Metrics

Following the previous works [24], the mean squared error (MSE) and the mean absolute error (MAE) are used for evaluation which are defined as:

$$MSE = \sqrt{\frac{1}{N}\sum_{1}^{N}(y_i - \hat{y}_i)^2}, \quad MAE = \frac{1}{N}\sum_{1}^{N}|y_i - \hat{y}_i|, \tag{15}$$

where N is the number of testing samples, y_i and $\hat{y}_i$ are the label and the prediction of the i-th sample.

B. Congestion Level Detection

In this section, the experimental results on congestion level detection are reported. The dataset and experimental settings are first described. Then, the compared methods are presented and the experimental results are discussed.

To demonstrate the effectiveness of the proposed method, we compare it with the following methods:

- Random Guess: Randomly select a value from $[0, 1)$ as the congestion level.
- Texture+LR: The LBP feature [48] with a linear regression.

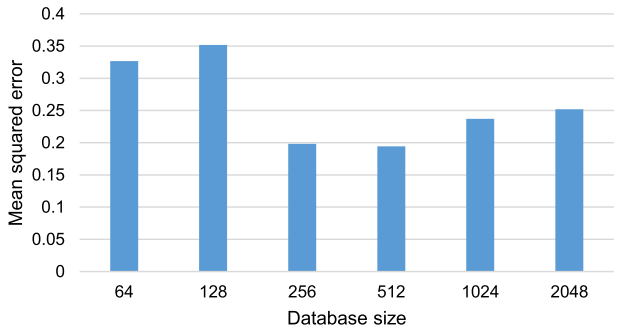

Fig. 5. The selection of the database size. The horizontal axis is the database size, and the vertical axis is the mean squared error. Note that lower error indicates better performance.

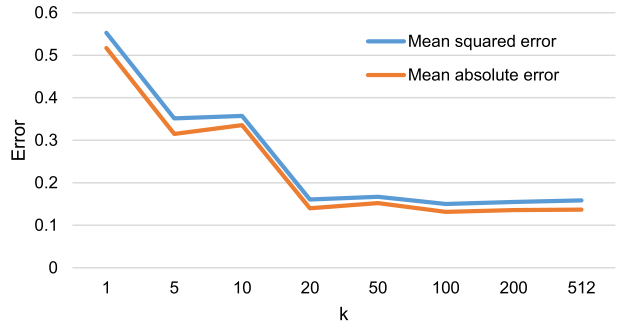

Fig. 6. The selection of k. The horizontal axis is the different k, and the vertical axis is the error. Note that lower error indicates better performance.

- Texture+ML: The LBP feature with metric learning for kernel regression.
- DeepNet+LR: The proposed deep network with a simple linear regression.
- DeepNet+LR (aug): DeepNet+LR with data augmentation.
- DML: The proposed method presented in Section IV.
- DML (aug): The proposed method with data augmentation.
- AlexNet+ML (aug): The AlexNet [49] with metric learning for regression and data augmentation.

1) NWPU_Congestion Dataset: The first dataset is the cross scene congestion level detection dataset described in Section III. The training set contains 5585 samples and the testing set contains 1372 samples.

In practice, s samples in the training set are randomly selected as the database. The rest data in the training set is split into two parts. 80% of them are used for training, and 20% of them are used for validation. Generally speaking, the bigger s is, the better the performance achieves. However, to balance the training size and the database size, s is set to 512 through our experiment as indicated in Figure 5. The number of neighbors k is set to 20 through the experiment as shown in Figure 6. In this figure, the performance is rather poor when k is too small, but when k is lager than 20, the performance becomes stable.

The experimental results are shown in Table I and Figure 7. From the results, we can obtain the following conclusions.

TABLE I

COMPARISON OF DIFFERENT METHODS FOR CONGESTION LEVEL DETECTION ON NWPU_CONGESTION DATASET

Methods	MSE	MAE
Random Guess	0.434	0.356
Texture+LR	0.387	0.315
Texture+ML	0.385	0.316
DeepNet+LR	0.430	0.395
DeepNet+LR (aug)	0.397	0.367
DML	0.169	0.127
DML (aug)	**0.145**	**0.109**
AlexNet+ML (aug)	0.318	0.272

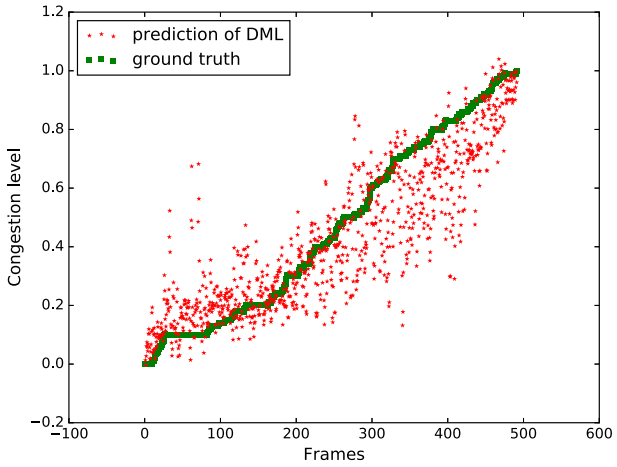

Fig. 7. The visualization of the performance of congestion level detection. The horizontal axis is frame number, and the vertical axis is the congestion level.

Deep network is hard to converge with linear regression and weak labels. Comparing Random Guess with DeepNet+LR, the performance of deep learning with a simple linear regression loss is similar to random guess. After data augmentation, the result of DeepNet+LR (aug) does not increase a lot which indicates that the linear regression with such weak labels can't guide the deep network to learn reasonable parameters. *With traditional features, the improvement of metric learning is limited.* Comparing Texture+LR with Texture+ML based on traditional texture feature, metric learning is useful to improve the performance. However, the improvement is very limited. *The proposed method is effective for congestion detection.* The proposed model with data augmentation (DML (aug)) achieves the best performance. That indicates that metric learning can effectively guide the training of deep network, and a large amount of data is essential to deep learning. *The data augmentation is essential to deep learning.* Comparing DeepNet+LR and DML to DeepNet+LR (aug) and Deep+ML (aug), the performance is increased a lot after data augmentation which indicates that data augmentation is important to deep learning. *The training of deeper network is still challenging.* However, AlexNet with metric learning and data augmentation (AlexNet+ML (aug)) does not achieve good performance. The reason might be that the training data is still not adequate.

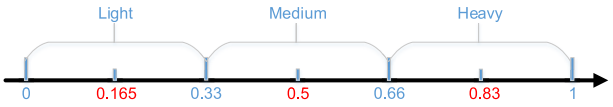

Fig. 8. Similar to the NWPU_Congestion dataset, the congestion level varies from [0, 1). We split it into three parts and the mean value of each part is used as the congestion level of the class.

TABLE II
COMPARISON OF DIFFERENT ARCHITECTURES FOR CONGESTION DETECTION ON TRAFFIC VIDEO DATASET

Methods	MSE	MAE
Random Guess	0.4190	0.3414
Texture+LR	0.0803	0.0313
Texture+ML	0.0841	0.0325
DeepNet+LR	0.2908	0.2420
DeepNet+LR(aug)	0.2810	0.1922
DML	0.0555	0.0236
DML(aug)	**0.0289**	**0.0029**
AlexNet+ML(aug)	0.2149	0.1718

2) Traffic Video Database: To test the performance of the proposed method under simple conditions, we conduct experiments on a dataset containing only one scene. The traffic video database [50] contains 254 video clips of traffic on the highway. The resolution of these videos is 320 × 240. There are different weathers and light conditions in the videos. This dataset contains only one scene (i.e., all videos are recorded by one camera with the same angle). It is designed for congestion video classification in which the videos are split into 3 levels: light, medium and heavy. To turn it into a regression task, the congestion threshold of each frame is set as 0.165, 0.5, 0.83 respectively for the light, medium and heavy class as shown in Figure 8. Similar to the experimental settings in NWPU_Congestion dataset, the size of database is set as 512. 80% of samples are used for training and 20% of samples are used for testing.

The experimental results can be seen in Table II and Figure 9. From these results, we can get the following conclusions. *The congestion detection in one scene is relatively easy.* Intuitively, the prediction is very accurate as shown in Figure 9. Quantitatively, comparing Random Guess to Texture+LR, a simple texture feature with a linear regression achieves huge improvement. *The deep network is still hard to converge with linear regression and week labels under simple conditions.* Comparing Texture+LR to DeepNet+LR, the performance of the traditional feature outperforms the deep network which indicates that the learning of deep networks is hard. *The proposed method is effective under simple conditions.* The best performance is achieved by the proposed method (DML (aug)) which confirms that the metric learning is useful to guide the training of deep network.

C. Crowd Counting

In this section, the experimental results about crowd counting are reported. The dataset and the experimental settings are first presented and the results are then discussed.

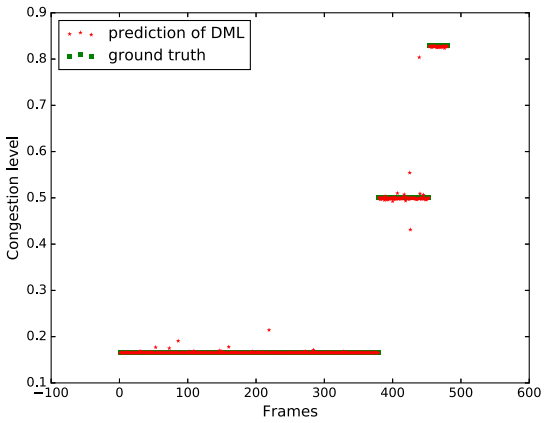

Fig. 9. The visualization of the performance of congestion level detection on traffic video dataset. The horizontal axis is the frame number, and the vertical axis is the congestion level.

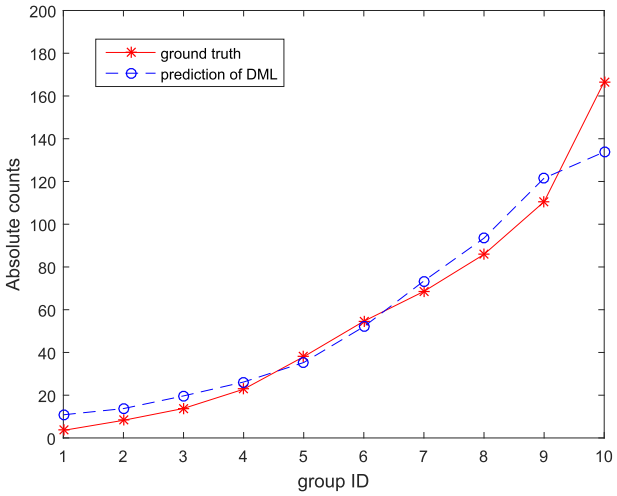

Fig. 10. The testing images are split into 10 groups according to the crowd size. The horizontal axis is the group ID and the vertical axis is the absolute counts.

1) The WorldExpo Dataset: The first dataset used for crowd counting is the WorldExpo dataset [23] which contains 3980 labeled frames. These frames come from 108 different surveillance cameras, the resolution of which is 576 × 720. In this experiment, 24 128 × 128 patches are extracted first and then resized to 64 × 64 for training. Similarly, the testing image is split into 24 patches and the summarized number of these patches is the final counting.

The first comparison method is local binary patterns (LBP) [51] and ridge regression (LBP+RR). The second method is proposed by Fiaschi *et al.* [22] in which random forest is utilized for prediction. Two deep learning based counting methods proposed by Zhang *et al.* [23] and Zhang *et al.* [24] are also included for comparison. Note that, the deep learning based methods utilize additional information to guide the training of networks.

The results of this experiment can be seen in Table III and Figure 10. From the results, we get the following conclusions. *The proposed method is effective for crowd counting.* The proposed method outperforms the other ones (LBP+RR and

TABLE III

COMPARISON OF DIFFERENT METHODS FOR CROWD COUNTING ON WORLDEXPO DATASET. NOTE THAT THE DATA AUGMENTATION IS NOT UTILIZED IN THIS EXPERIMENT AND THE MAE IS USED AS THE METRIC FOR EVALUATION. THE BEST RESULTS WITH AND WITHOUT ADDITIONAL INFORMATION ARE INDICATED IN **BOLD**

Methods	Scene 1	Scene 2	Scene 3	Scene 4	Scene5	Average	Additional info
LBP+RR	13.6	59.8	37.1	21.8	23.4	31.0	No
Fiaschi *et al.* [22]	**2.2**	87.3	22.2	16.4	**5.4**	26.7	No
DML	5.5	**27.9**	**18.3**	**13.9**	6.6	**14.5**	No
Zhang *et al.* [24]	**2.0**	29.5	**9.7**	**9.3**	**3.1**	**10.7**	Yes
Zhang *et al.* [25]	3.4	**20.6**	13.0	13.0	8.0	11.6	Yes

TABLE IV

COMPARISON OF DIFFERENT METHODS FOR CROWD COUNTING ON SHANGHAI_TECH DATASET. NOTE THAT THE DATA AUGMENTATION IS NOT UTILIZED IN THIS EXPERIMENT. THE BEST RESULTS WITH AND WITHOUT ADDITIONAL INFORMATION ARE INDICATED IN **BOLD**

	Part_A		Part_B		Average		
Methods	MAE	MSE	MAE	MSE	MAE	MSE	Additional info
LBP+RR	303.2	371.0	59.1	81.7	148.3	233.5	No
DML	**106.0**	**155.9**	**29.9**	**53.6**	**57.7**	**103.5**	No
Zhang *et al.* [24]	181.8	277.7	32.0	49.8	86.7	172.5	Yes
Zhang *et al.* [25]	**110.2**	**173.2**	**26.4**	**41.3**	**57.0**	**109.8**	Yes

Fiaschi *et al.* [22]) which doesn't use additional information for training. It confirms that the proposed method is effective to count the crowd size. *The high-level feature extracted by deep networks is effective for crowd counting.* The deep learning based methods (DML, Zhang *et al.* [23] and Zhang *et al.* [24]) outperforms traditional methods (LBP+RR and Fiaschi *et al.* [22]) which indicates that the high-level semantic features are useful for crowd counting. *The deep networks can be effectively trained with additional information.* The deep learning methods (Zhang *et al.* [23] and Zhang *et al.* [24]) with additional information (i.e., density map) achieve top performance. However, it is hard to apply these methods to large-scale data since the labeling of the additional information (human heads in crowd) is very time-consuming. *The counting of large crowd number is challenging.* As shown in Figure 10, the error of the large crowd number is higher than the small number which indicates that the crowd counting is very challenging in large crowd scenarios.

2) The Shanghai_Tech Dataset: The shanghai_Tech dataset [24] is a challenging dataset containing two parts of images. The first part (Part_A) is the crowd images collected from Internet and the second part (Part_B) is taken from the busy street in Shanghai. This dataset contains 1198 labeled images. Part_A contains 482 images and Part_B contains 716 images. The crowd density varies significantly in this dataset. Following the experimental settings in [24], 300 images in Part_A are used for training and the rest are used for testing, and 400 images in Part_B are used for training and the rest for testing. Similar to the WorldExpo dataset, the images are first split into patches for training and then the patches are accumulated as a whole image for testing.

The first comparison method is LBP feature with a ridge regressor. The other two methods are deep learning based algorithms [23], [24] which utilize additional information for training.

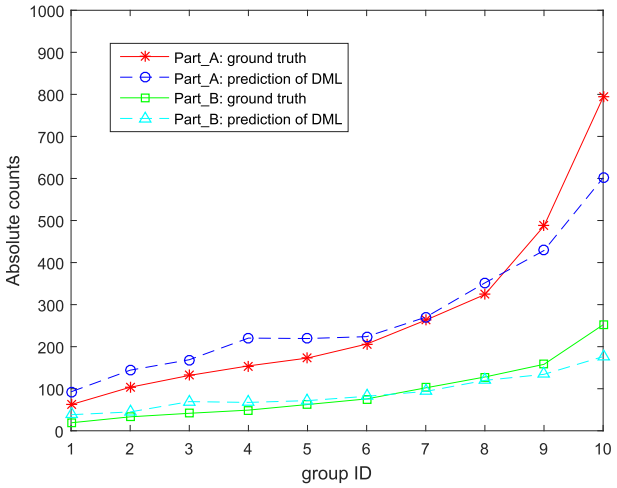

Fig. 11. The testing images of Part_A and Part_B are split into 10 groups according to the crowd size. The horizontal axis is the group ID and the vertical axis is the absolute counts.

The results can be seen in Table IV and Figure 11 which are very similar to the results on the WordExpo dataset. The proposed method (DML) outperforms the traditional method (LBP+RR) and achieves comparable result compared to other deep learning methods (Zhang *et al.* [23] and Zhang *et al.* [24]), especially in Part_A indicating that the proposed model can be effectively optimized for crowd counting. The visualization of the Part_A and Part_B is shown in Figure 11. In this figure, the prediction of Part_B is more accurate than Part_A which indicates that the background variation is a big challenge for crowd counting.

VI. Conclusion

In this paper, a deep metric learning based method is proposed for crowdedness regression tasks, such as congestion detection and crowd counting. To extract high-level semantic features, a deep network is proposed. However, the training of such a network is very challenging with weak labels. To remedy this, we propose to guide the training of the network with metric learning since it is capable of embedding local structural information. The proposed deep metric learning model is trained end-to-end for better optimization. In this procedure, the learning of better representations and distance measurement are simultaneous. The experiments confirm that the metric learning is effective to guide the training of deep network and with high-level semantic features, the metric learning shows more improvement than traditional features.

However, we find that the deeper network is still hard to train even guided with metric learning. Thus, we will further expand our dataset and try to find more effective ways to utilize deeper networks in the future.

References

[1] J. Wang, W. Fu, J. Liu, and H. Lu, "Spatiotemporal group context for pedestrian counting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1620–1630, Sep. 2014.

[2] Y. Zhang, L. Qin, R. Ji, H. Yao, and Q. Huang, "Social attribute-aware force model: Exploiting richness of interaction for abnormal crowd detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 7, pp. 1231–1245, Jul. 2015.

[3] J. Lu, G. Wang, and P. Moulin, "Localized multifeature metric learning for image-set-based face recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 3, pp. 529–540, Mar. 2016.

[4] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4766–4779, Dec. 2015.

[5] W. Zuo, D. Zhang, and K. Wang, "Bidirectional PCA with assembled matrix distance metric for image recognition," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 4, pp. 863–872, Aug. 2006.

[6] J. Hu, J. Lu, and Y.-P. Tan, "Deep metric learning for visual tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 11, pp. 2056–2068, Nov. 2016.

[7] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Deep coupled metric learning for cross-modal matching," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1234–1244, 2017.

[8] S. Hu, J. Wu, and L. Xu, "Real-time traffic congestion detection based on video analysis," *J. Inf. Comput. Sci.*, vol. 9, no. 10, pp. 2907–2914, 2012.

[9] C. Zhan, X. Duan, S. Xu, Z. Song, and M. Luo, "An improved moving object detection algorithm based on frame difference and edge detection," in *Proc. 4th Int. Conf. Image Graph.*, Aug. 2007, pp. 519–523.

[10] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, nos. 1–3, pp. 185–203, Aug. 1981.

[11] A. Sobral, L. Oliveira, L. Schnitman, and F. De Souza, "Highway traffic congestion classification using holistic properties," in *Proc. 10th Int. Conf. Signal Process., Pattern Recognit. Appl.*, 2013.

[12] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.*, 1981, pp. 674–679.

[13] Y. Yuan, J. Wan, and Q. Wang, "Congested scene classification via efficient unsupervised feature learning and density estimation," *Pattern Recognit.*, vol. 56, pp. 159–169, Aug. 2016.

[14] K. G. Derpanis and R. P. Wildes, "Classification of traffic video based on a spatiotemporal orientation analysis," in *Proc. IEEE Workshop Appl. Comput. Vis.*, Jan. 2011, pp. 606–613.

[15] A. Riaz and S. A. Khan, "Traffic congestion classification using motion vector statistical features," in *Int. Conf. Mach. Vis.*, 2013, p. 90671A.

[16] E. Dallalzadeh, D. Guru, and B. Harish, "Symbolic classification of traffic video shots," in *Advances in Computational Science, Engineering and Information Technology*. Cham, Switzerland: Springer, 2013, pp. 11–22.

[17] A. N. Marana, M. A. Cavenaghi, R. S. Ulson, and F. L. Drumond, "Real-time crowd density estimation using images," in *Proc. Adv. Int. Symp. Vis. Comput.*, 2005, pp. 355–362.

[18] C. S. Regazzoni, A. Tesei, and G. Vernazza, "A bayesian network for automatic visual crowding estimation in underground stations," in *Image Technology*, Berlin, Germany: Springer, 1996, pp. 203–230.

[19] S.-Y. Cho, T. W. S. Chow, and C.-T. Leung, "A neural-based crowd estimation by hybrid global learning algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 4, pp. 535–541, Aug. 1999.

[20] K. Chen, C. C. Loy, S. Gong, and T. Xiang, "Feature mining for localised crowd counting," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.

[21] V. S. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1324–1332.

[22] L. Fiaschi, U. Kothe, R. Nair, and F. A. Hamprecht, "Learning to count with regression forest and structured labels," in *Proc. Int. Conf. Pattern Recognit.*, 2012, pp. 2685–2688.

[23] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 833–841.

[24] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 589–597.

[25] D. Tao, L. Jin, Y. Wang, Y. Yuan, and X. Li, "Person re-identification by regularized smoothing KISS metric learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1675–1685, Oct. 2013.

[26] W. Li, Y. Wu, and J. Li, "Re-identification by neighborhood structure metric learning," *Pattern Recognit.*, vol. 61, pp. 327–338, Jan. 2017.

[27] W. Zuo, D. Zhang, and K. Wang, "An assembled matrix distance metric for 2DPCA-based image recognition," *Pattern Recognit. Lett.*, vol. 27, no. 3, pp. 210–216, 2006.

[28] Z. Huang, R. Wang, S. Shan, and X. Chen, "Face recognition on large-scale video in the wild with hybrid Euclidean-and-Riemannian metric learning," *Pattern Recognit.*, vol. 48, no. 10, pp. 3113–3124, 2015.

[29] H. Chang and D.-Y. Yeung, "Kernel-based distance metric learning for content-based image retrieval," *Image Vis. Comput.*, vol. 25, no. 5, pp. 695–703, 2007.

[30] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Feb. 2009.

[31] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon, "Information-theoretic metric learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 209–216.

[32] J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov, "Neighbourhood components analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 513–520.

[33] S. C. H. Hoi, W. Liu, M. R. Lyu, and W.-Y. Ma, "Learning distance metrics with contextual constraints for image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2006, pp. 2072–2078.

[34] K. Q. Weinberger and G. Tesauro, "Metric learning for kernel regression," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2007, pp. 612–619.

[35] R. Huang and S. Sun, "Kernel regression with sparse metric learning," *J. Intell. Fuzzy Syst.*, vol. 24, no. 4, pp. 775–787, 2013.

[36] B. Xiao, X. Yang, H. Zha, Y. Xu, and T. S. Huang, "Metric learning for regression problems and human age estimation," in *Advances in Multimedia Information Processing—PCM*. Berlin, Germany: Springer, 2009, pp. 88–99.

[37] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1875–1882.

[38] Z. Li and J. Tang, "Weakly supervised deep metric learning for community-contributed image retrieval," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 1989–1999, Nov. 2015.

[39] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2016, pp. 4004–4012.

[40] F. L. Hall, "Traffic stream characteristics," Traffic Flow Theory. U.S. Federal Highway Admin., Washington, DC, USA, 1996, ch. 2, pp. 1–38.

[41] C. Deng, J. Xu, K. Zhang, D. Tao, X. Gao, and X. Li, "Similarity constraints-based structured output regression machine: An approach to image super-resolution," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2472–2485, Dec. 2016.

[42] X. Liu, L. Huang, C. Deng, J. Lu, and B. Lang, "Multi-view complementary hash tables for nearest neighbor search," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1107–1115.
[43] X. Liu, C. Deng, B. Lang, D. Tao, and X. Li, "Query-adaptive reciprocal hash tables for nearest neighbor search," *IEEE Trans. Image Process.*, vol. 25, no. 2, pp. 907–919, Feb. 2016.
[44] M. Wang, W. Fu, S. Hao, H. Liu, and X. Wu, "Learning on big graph: Label inference and regularization with anchor hierarchy," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1101–1114, May 2017.
[45] M. Wang, W. Fu, S. Hao, D. Tao, and X. Wu, "Scalable semi-supervised learning by efficient anchor graph regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1864–1877, Jul. 2016.
[46] M. Abadi *et al.* (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, Software Available From Tensorflow.Org.* [Online]. Available: http://tensorflow.org/
[47] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[48] F. Lu and J. Huang, "An improved local binary pattern operator for texture classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2016, pp. 1308–1311.
[49] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.
[50] A. B. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2005, pp. 771–776.
[51] M. Pietikäinen, "Local binary patterns," *Scholarpedia*, vol. 5, no. 3, p. 9775, 2010.

Qi Wang received the B.E. degree in automation and the Ph.D. degree in pattern recognition and intelligent systems from the University of Science and Technology of China, Hefei, China, in 2005 and 2010, respectively. He is currently an Associate Professor with the School of Computer Science, Center for OPTical IMagery Analysis and Learning and with the Unmanned System Research Institute, Northwestern Polytechnical University, Xi'an, China. His research interests include computer vision and pattern recognition.

Jia Wan is currently pursuing the M.E. degree with the School of Computer Science and the Center for OPTical IMagery Analysis and Learning, Northwestern Polytechnical University, Xi'an, China. His current research interests include image classification and congestion analysis.

Yuan Yuan is currently a Full Professor with the School of Computer Science and the Center for OPTical IMagery Analysis and Learning, Northwestern Polytechnical University, Xi'an, China. She has authored or co-authored over 150 papers, including about 100 in reputable journals, such as the IEEE TRANSACTIONS AND PATTERN RECOGNITION, as well as conference papers in CVPR, BMVC, ICIP, and ICASSP. Her current research interests include visual information processing and image/video content analysis.