Deep Understanding of Urban Mobility from Cityscape Webcams

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Department of Electrical and Computer Engineering

Shanghang Zhang

M.S. Electronics Engineering and Computer Science,
Peking University
B.E. Electronic Science, Southeast University

Carnegie Mellon University
Pittsburgh, PA

May, 2018

This thesis is dedicated to my family and my advisors.

Abstract

Deep understanding of urban mobility is of great significance for many real-world applications, such as urban traffic management and autonomous driving. This thesis develops deep learning methodologies to extract vehicle counts from streaming real-time video captured by multiple low resolution web cameras and construct maps of traffic density in a city environment; in particular, we focus on cameras installed in the Manhattan borough of NYC. The large-scale videos from these web cameras have low spatial and temporal resolution, high occlusion, large perspective, and variable environment conditions, making most existing methods to lose their efficacy. To overcome these challenges, the thesis develops several techniques: 1. a block-level regression model with a rank constraint to map the dense image feature into vehicle densities; 2. a deep multi-task learning framework based on fully convolutional neural networks to jointly learn vehicle density and vehicle count; 3. deep spatio-temporal networks for vehicle counting to incorporate temporal information of the traffic flow; and 4. multi-source domain adaptation mechanisms with adversarial learning to adapt the deep counting model to multiple cameras. To train and validate the proposed system, we have collected a large-scale webcam traffic dataset CityCam that contains 60 million frames from 212 webcams installed in key intersections of NYC. Of there, $60,000$ frames have been annotated with rich information, leading to about $900,000$ annotated objects. To the best of our knowledge, it is the first and largest webcam traffic dataset with such large number of elaborate annotations. The proposed methods are integrated into the CityScapeEye system that has been extensively evaluated and compared to existing techniques on different counting tasks and datasets, with experimental results demonstrating the effectiveness and robustness of CityScapeEye.

Acknowledgments

The pursuit of Ph.D. has been the most precious and memorable experience for me. It builds me up into a better myself, in both academic arena and personality. I have sincere appreciations to acknowledge many people who support and help me along the way. First and foremost, I am indebted to my Ph.D. advisors, Prof. José M. F. Moura and Prof. João P. Costeira. It is their untiring mentorship that leads me to the fruitful PhD experience. Prof. Moura has built a great role model for me, both in research and in life. He has taught me how to think creatively, research thoroughly, and write clearly. I found myself inspired with effective ideas each time when interacting with him. Prof. Moura generously provided the resources that enabled me to carry out my research, patiently guided me through the winding path of seeking for truth and knowledge, and warmly supported me with consistent help and valuable opportunities. I would also like to give my great appreciation to my co-advisor Prof. João P. Costeira. Not only do I benefit from his invaluable input to my research, but also from his readily available help all along my work. He acts as a supportive advisor, brilliant researcher, sincere friend, and considerate father. I have been greatly inspired by his insight and enthusiasm of conducting research. It is his wisdom that unveils the true joy of the Ph.D. journey. I feel so fortunate to have Prof. Moura and Prof. Costeira as my advisors, and will cherish all the memorable experience to work with them.

I am also grateful to the other members of my thesis committee: Prof. Ragunathan Rajkumar, Prof. Katerina Fragkiadak, Dr. André F. T. Martins, and Dr. Ricardo Silveira Cabral for serving on my defense. They elaborately provided me valuable comments and helped make the final stretch of my Ph.D. successful. I appreciate their time devoted to reviewing my research projects, prospectus proposal, and thesis. Their inspiring and valuable suggestions contributed a lot to the improvement of the thesis. I also sincerely appreciate all the support from the Ph.D. Academic Program Advisor, Nathan Snizaski. His professional and considerate advice is of great help for the successful fulfillment of my Ph.D. study.

I would like to give my great gratitude to my wonderful colleagues, Guanhang Wu, Han Zhao, Jian Du, Rajshekhar Das, Evgeny Toropov, Liangyan Gui, Yuxiong Wang, Qiwei Han, Jayakorn Vongkulbhisal, Zhiding Yu, Pengtao Xie, Xin Li, Satwik Kottur, Jian Wang, Siheng Chen, Jonathan Mei, Dong Huang, Wenbo Liu, and Hongyang Zhang, for their warm support, inspiring suggestions, kind help, and sincere friendship. I would like to especially thank my close collaborators, Guanhang Wu, Han Zhao, Jian Du, and Rajshekhar Das, for their strong capability, responsibility, wisdom, and supports. They always fight with me against the tough problems and give me strength. We are not only collaborators, but also great friends. One of the most beautiful memory of my Ph.D. experience is to work with these excellent colleagues. I would also like to thank my internship mentors, who provided the guidance to make my work successful: Dr. Xiaohui Shen, Dr. Zhe Lin, and Dr. Radomír Měch. It was enjoyable and fruitful time to work with them. Their experienced advice improved my research capability and better prepared me for the future career. I would like to acknowledge the Adobe Academic Collaboration Funding provided by Adobe Research, and the financial support by the FCT from the Carnegie Mellon Portugal Program under Grant [SFRH/BD/113729/2015].

Finally, I would like to thank my family for their enormous support and warm companion. My father Yue, mother Yuling, and brother Shangduo, have been the pillar of support and the epitome of love, strength, and sacrifice. They always stand by me wherever I go, and whatever I do. Their love strongly motivates me to become better myself and believe in a beautiful future. Without their support, I would not become who I am. I would like to give my special thanks to my brother Shangduo. He is a lovely angel who brings great happiness to my family. He can always make me laugh whenever I talk with him, and his smile is the most beautiful thing in the world. He often amazes me with his wisdom, kindness, and rich knowledge. I know there is a long road for him to go, but I will always stand by him. Together, we embrace a beautiful future. I would also like to take the chance to give my thanks and yearning to my grandmother Guizhi. She is the most kindhearted person and gave me all her unconditional love. I am sorry that she has not lived to see me finish my Ph.D, but she is always living in my heart. I wish her a peaceful and happy life in the heaven. In the end, I would like to thank my beloved friends, Xukun Li, Ruosi Wang, Xiling Sun, Nanjing Yu, Meng Song, and Keyu Jin, for our lifelong friendship. Even though we are living in the different places of the world, our hearts are always beating together. With their understanding and support, I am full of strength to chase my dream.

Contents

List of Tables

List of Figures

Part I

In this part we explain the problem definition, thesis contribution, related work, and dataset collection.

- Chapter 1 Introduction
- Chapter 2 Large Scale City Wide Web Camera Data

Chapter 1

Introduction

1.1 Problem Definition

Many cities are being instrumented with hundreds of surveillance cameras mounted on streets and intersections [55, 95, 96]. They capture traffic 24 hours a day, 7 days a week, generating large scale video data. Web camera (Webcam) videos can be regarded as highly versatile, being an untapped potential to develop many vision-based techniques for applications like traffic flow analysis and crowd counting. This thesis aims to extract vehicle counts from streaming real-time video captured by low resolution webcams and provide tools to construct maps of traffic density in a city environment. We focus on NYC traffic cameras in the Manhattan borough. Traffic density is the number of vehicles per unit area of a road [56], which together with average vehicle speed, enables traffic flow analysis [86]. To estimate the traffic density from each webcam, we select a region of fixed length (yellow dotted rectangle) in a video frame and count the number of vehicles in that region, as shown in Figure 1.1.

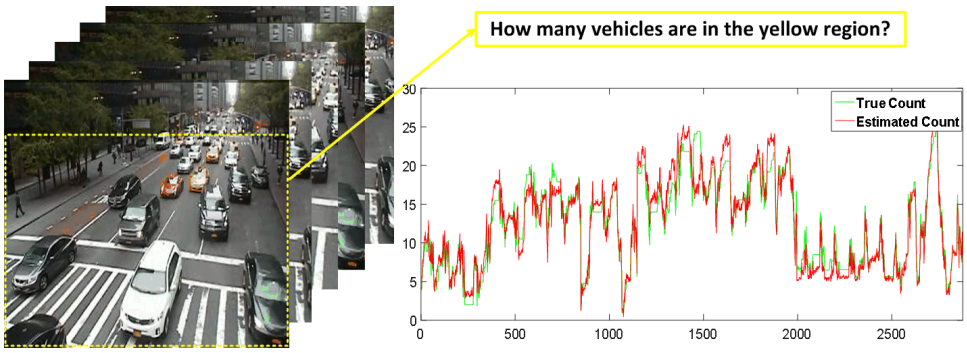


Figure 1.1: Problem Statement. Traffic density is estimated by counting the number of vehicles in a given region.

Understanding traffic density is of great importance for many real-world applications, such as urban traffic management and autonomous driving. Traffic congestion is a serious problem that significantly impacts city life quality. According to the annual INRIX Traffic Scorecard, commuters in NYC spend an average of 53 hours per year sitting behind the wheel and suffer a 20 percent increase in travel time due to bad traffic. Traffic density estimation is a fundamental building block of traffic management to mitigate traffic congestion. Autonomous vehicles for urban driving have seen rapid progress in the past 30 years [84]. Most of the effort has been in processing sensing data from the driverless car itself [16, 18, 63, 102]. However, an autonomous driverless car should be capable of not only planning its local trajectory safely but also determining the optimal global route to reach its destination effectively. To achieve this goal, the system we develop here, that we refer to as CityScapeEye, will provide a means to process simultaneously the video streaming from multiple city traffic cameras.

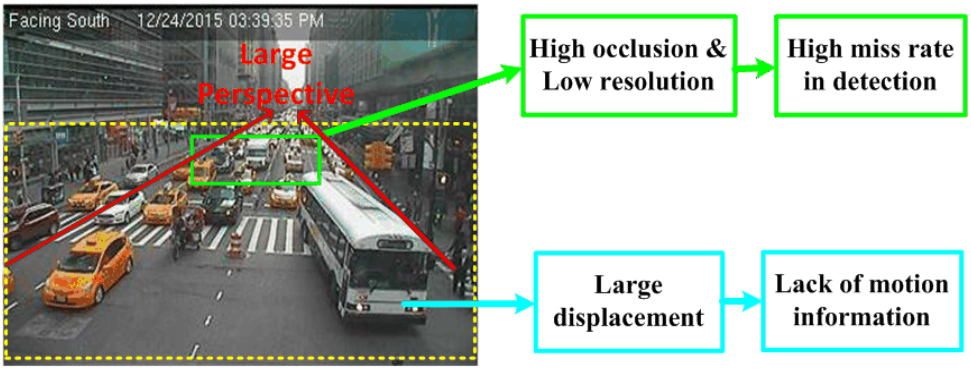


Figure 1.2: Traffic counting from webcam videos is challenged by low frame rate, low resolution, high occlusion, and large perspective.

Important as it is, understanding urban traffic from large scale city cameras is a difficult problem faced with severe challenges (illustrated in Figure 1.2) due to network bandwidth limitations, lack of persistent storage, and privacy concerns. Publically available webcam video is limited by:

1. Low frame rate. The time interval between two sequential frames of a webcam video typically ranges from 1s to 3s, resulting in large vehicle displacement, with some vehicles only appearing in one frame of the video sequence.
2. Low resolution. Webcam video resolutions vary, for example, 352×240, 320×240 or 704×480 pixels. The vehicle at the top of a frame can be as small as 5×5 pixels. Further, image compression also induces artifacts.

3. High occlusion. Cameras installed at urban intersections often capture videos with high traffic congestion, especially during rush hours, resulting in vehicle high occlusion rate.
4. Large perspective. Cameras are installed at the top of tall poles with a high point of view to be able to capture more video content, resulting in videos with large perspective differences. Vehicle scales vary dramatically based on their distance to the camera.
5. Variable environmental conditions. Different cameras have different positions, scenes, and perspectives. Even for the same camera, weather and illumination change significantly over time.
6. Different traffic patterns. Throughout the day or night, weeks and months, different seasons, as well as different streets, traffic patterns change significantly from sparse to heavy.

All these challenges make traffic density estimation from webcam data very difficult. These difficulties preclude existing approaches that we group into five categories: frame differencing [28, 98], detection [97, 124], motion based counting [23, 24, 73, 93], density estimation [103], and deep learning [5, 79, 117, 122, 123] methods. Frame differencing, detection, and motion based methods are sensitive to environment conditions and tend to fail in high congestion, low resolution, and low frame rate videos. While density estimation approaches are less dependent on environment conditions, they perform poorly in videos with large perspective and severe occlusion. Further, most existing methods fail to account for domain adaptation and exhibit large errors when estimating vehicle counts [25, 36, 50, 115]. To overcome these limitations, we develop a system for urban traffic from large scale city cameras that solves the following crucial problems that emerge in city wide traffic flow analysis:

1. Jointly estimate vehicle density and vehicle count;
2. Accurately estimate traffic density for long-term and non-interrupted video sequences; and
3. Estimate traffic density for multiple cameras with different environmental, weather, light, or traffic conditions.

1.2 Contributions

In this thesis, we develop CityScapeEye, a system to estimate traffic density that is scalable, robust, and able to generalize. To train and validate CityScapeEye, we first collected and labelled a large-scale

webcam traffic dataset that contains 60 million frames from 212 webcams installed in key intersections of NYC. We have annotated 60,000 frames with vehicle bounding box, orientation, re-identification, speed, category, traffic flow direction, weather, and time. We estimate that our database includes around 900,000 total annotations. Unlike the existing car data set KITTI[37] and Detrac[112], which focus on vehicle models, our dataset emphasizes the real world traffic network in a large metropolis. To the best of our knowledge, it is the first and largest webcam traffic dataset with elaborate annotations. We refer to our annotated dataset as CityCam and introduce it in Chapter 2.

In Chapter 3, we develop a block-level regression model with a rank constraint (OPT-RC) that embeds road geometry in the weight matrix and significantly reduces error induced by camera perspective. The regression model learns different weights for different blocks to increase the number of degrees of freedom of the weights and relax the rank one constraint. It avoids detecting or tracking individual vehicles, enabling the counting model to perform accurately when the traffic video has low resolution and high occlusion. Experimental evaluation showed promising performance, especially in reducing large errors induced by camera perspective.

To further improve the model scalability and robustness of CityScapeEye, in Chapter 4, we propose a novel deep multi-task learning model based on fully convolutional regression networks (FCN-MT) to jointly estimate vehicle density and vehicle count with end-to-end dense prediction, allowing for arbitrary input image resolution and adapting to different vehicle scale and perspective. FCN-MT replaces background subtraction, feature extraction, and the linear regression model previously used in OPT-RC with convolutional neural networks. To avoid large errors induced by vehicle scale variations, we jointly perform image-level vehicle count regression and pixel-level vehicle density estimation. The global count regression takes the estimated density map as input feature. Instead of directly regressing the global vehicle count from the last feature map, we develop a residual learning framework. This reformulates global count regression as learning residual functions with reference to the sum of densities in each frame. This design avoids learning unreferenced functions and eases the training of the network. Experimental evaluation verifies the efficacy of the deep model and shows performance that is better than the performance of state-of-the-art alternatives. The mean absolute error (MAE) on

testing data that measures the average vehicles miscounts per frame is reduced from 4.56 obtained with OPT-RC to 2.74.

To further explore the temporal correlation of traffic flow, we develop in Chapter 5 a deep spatio-temporal network architecture to sequentially estimate vehicle count by integrating Long Short-Term Memory (LSTM) networks with the FCN-MT systems (FCN-rLSTM). Such design leverages the strengths of FCN for pixel-level prediction and the strengths of LSTM for learning complex temporal dynamics. The residual learning connection between FCN and LSTM reformulates the vehicle count regression as learning residual functions with reference to the sum of densities in each frame, which significantly accelerates the training of the networks. To preserve feature map resolution, we propose a Hyper-Atrous combination to integrate atrous convolution in FCN and combine feature maps of different convolution layers. FCN-rLSTM enables refined feature representation and a novel end-to-end trainable mapping from pixels to vehicle count. We extensively evaluated the proposed method on different counting tasks with three datasets, with experimental results demonstrating their effectiveness and robustness. In particular, FCN-rLSTM reduces the MAE from 2.74 to 1.53 on the CityCam Dataset. The training process is accelerated by 5 times on average.

There are hundreds of cameras in NYC. It is too expensive to label an abundant number of frames for each camera. To adapt the deep counting model to different cameras and enviromental conditions, in Chapter 6, we develop a multi-source domain adaptation mechanism with adversarial learning. We first propose a new generalization bound for multi-camera vehicle counting, when there are multiple cameras with labeled instances and one target camera with unlabeled instances. Interestingly, our theory also leads to an efficient learning strategy using adversarial neural networks: we show how to interpret it as learning feature representations that are invariant to the multiple camera shifts while still being discriminative for the learning task. To this end, we propose two models: the first model optimizes directly our bound, while the second model is a smoothed approximation of the first one, leading to a more data-efficient and task-adaptive model. The optimization tasks of both models are minimax saddle point problems that can be optimized by adversarial training. Experimental results demonstrate the effectiveness and robustness of the proposed method.

To summarize, in Chapter 2, we introduce our web camera dataset CityCam. Chapter 3 describes our optimization based approach to traffic density estimation and experimental results. Chapter 4 explains our proposed FCN-MT. Chapter 5 presents the FCN-rLSTM architecture for counting vehicles from traffic videos. Chapter 6 introduces the multiple camera domain adaptation with adversarial learning. Finally Chapter 7 concludes the thesis and introduces the future work. Along each Chapter, we benchmark the results with state-of-the-art methods.

1.3 Related Work

In this section, we provide a general review of related work on traffic density estimation, categorized into two groups: non-vision based methods and vision-based methods. We will present separately literature reviews for some of the specific techniques as these are introduced in the following Chapters.

1.3.1 Non-vision Based Methods

Traditionally road traffic density has been estimated with roadside magnetic loop detectors, wireless vehicle sensors, roadside radar, and infra-red counters [64, 90, 106, 116]. These techniques suffer from high installation costs and disruption during maintenance. These non-camera devices can not capture information such as vehicle type and scale. Reference [106] estimates vehicle density on a road segment based on traffic data collected from two wireless vehicle sensors placed at both ends of a road segment, using a modified extended Kalman filter. Reference [57] develops Vehicular Ad Hoc Networks (VANETs), which require appropriate hardware to be installed on every vehicle.

1.3.2 Vision-based Methods

While magnetic loop detectors and infra-red sensors [29] count vehicles with high accuracy, they provide limited traffic information and require separate systems for traffic surveillance. Alternative vision-based approaches deal with camera data, which have low installation costs, bring little traffic disruption during maintenance, and provide wider coverage and more detailed understanding of traffic

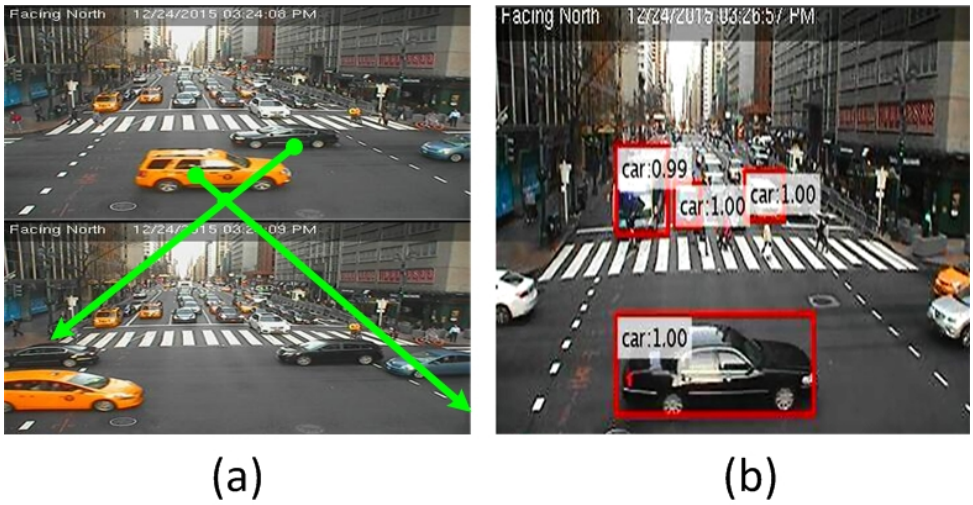


Figure 1.3: Failure cases of existing methods. (a) Tracking-based methods are precluded due to large vehicle displacement. (b) High missing rate from Faster RCNN detection results.

flow patterns [29, 54, 75]. Vision-based methods can be further divided into five categories:

1. **Frame differencing.** Frame differencing methods count vehicles based on the difference between sequential frames and are easy to implement. They cannot deal with noise, abrupt illumination changes, and background changes [28, 98]. Background subtraction techniques can alleviate such problems and are widely used in the literature, but sophisticated background subtraction methods, e.g., Hidden Markov Models and Neural Networks [68], which might deal with various environmental variations, incur high computational costs and require large amounts of labeled data [91].

2. **Detection based methods.** These methods [97, 124] try to identify and localize vehicles in each frame. They perform poorly in low resolution and high occlusion videos. Figure 1.3 shows detection results by Faster RCNN [83]. Even though trained on our collected and annotated webcam dataset CityCam, it still exhibits a very high miss rate.

3. **Motion based methods.** Several methods [23, 24, 73, 93] combine vehicle detection and tracking to estimate traffic flow. The general procedure consists of three steps: (i) motion detection, (ii) tracking of individual vehicles, and (iii) combining trajectories' information to derive an overall description of traffic flow. These methods tend to fail because of the webcam video's low frame rate and lack of motion information. Figure 1.3 shows a large displacement of a vehicle (black car) in successive frames due to the low frame rate. Some vehicles only appear once in the video and their trajectory

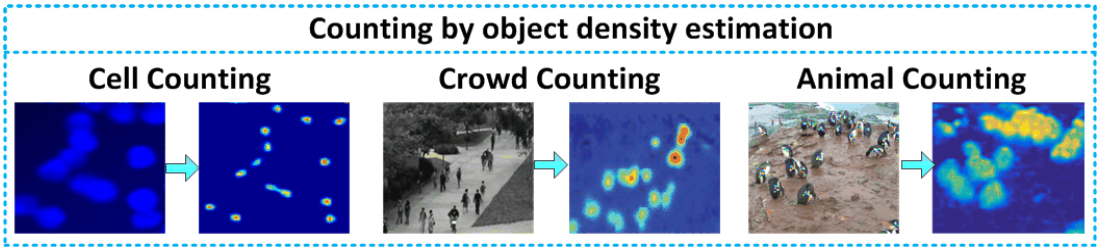


Figure 1.4: Examples of density estimation based methods.

cannot be well estimated.

4. **Density estimation based methods.** To deal with the limitations of detection and motion based methods, recently several density estimation approaches have been proposed to count objects in the literature. These techniques map the dense (pixel-level) image feature into object densities, thereby avoiding detection or tracking of each object, as shown in Figure 1.4. Reference [103] casts the counting problem as estimating an image density whose integral over any image region gives the count of objects within that region. It formulates object density as a linear transformation of each pixel local features, with a uniform weight vector for the whole image. This method suffers from low accuracy when the camera has large perspective and oversized vehicles occur.

5. **Deep learning based methods.** More recently, several deep learning based counting models have been developed [5, 79, 117, 122, 123], which have significantly improved counting performance. The network in [117] outputs a 1D feature vector and fits a ridge regressor to perform the final density estimation. This technique cannot perform pixel-wise dense prediction and loses spatial information. Reference [79] is based on fully convolutional neural networks; it does not have deconvolutional or upsampling layers, resulting in the output density map being much smaller than the input image. Reference [5] jointly estimates the object density map and performs foreground segmentation, but it does not solve for the large perspective and object scale variation problems.

In summary, frame differencing, detection, and motion based methods are sensitive to environment conditions and tend to fail in high congestion, low resolution, and low frame rate videos. While density estimation approaches are less dependent on environment conditions, they perform poorly in videos with large perspective and severe occlusion. In addition, most of the existing methods are not capable of estimating exact vehicle counts [25, 36, 50, 115].

Chapter 2

CityCam: Large-Scale City Webcam Data

This thesis aims to understand traffic density and traffic flow from low quality videos captured by large scale web cameras in a city environment. We focus on NYC traffic cameras. NYC has been instrumented with 564 surveillance cameras mounted on streets and intersections. These cameras capture traffic 24 hours a day, 7 days a week, generating large scale video data. Different cameras have different positions, scenes, and perspectives. Even for the same camera, weather and illumination change significantly over time. Figure 2.1 shows the cameras installed in the Manhattan borough and images from 9 sampled cameras.

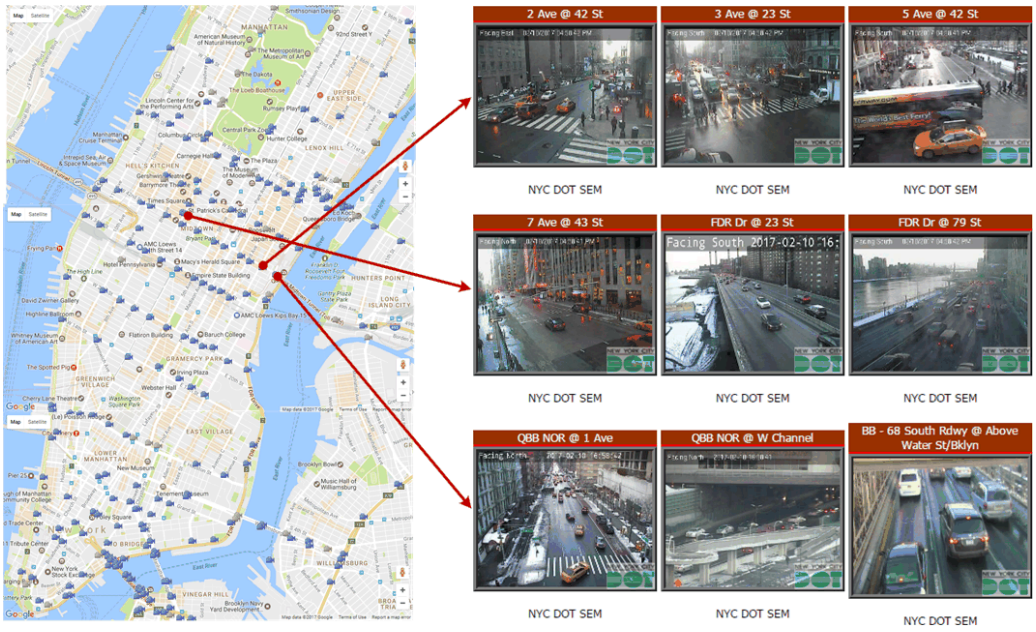


Figure 2.1: Cameras installed in Manhattan and 9 sampled cameras.

	Dataset	GlobalCam	KITTI	TRANCOS	CompCars	DETRAC	CityCam
Train	# of Frames	-	7.48k	823	50k	84k	32k
	Boxes	-	40.6k	-	-	578k	480k
Test	# of Frames	125M	7.52k	421	-	56k	28k
	Boxes	-	39.7k	-	-	632k	420k
Properties	Traffic Net	✓	X	X	X	X	✓
	Sensor	WebCam	Cam	WebCam	WebCam	Cam	WebCam
	Task	Count	Det	Count	Det	Count	Det, Count
	Vehicle type	X	✓	X	✓	✓	✓
	Vehicle Re-ID	X	X	X	X	X	✓
	Orientation	X	X	X	✓	✓	✓
	Multi. Wea.	X	X	✓	X	✓	✓
	Frame rate	1f/60s	10f/s	X	X	25f/s	0.5f/s ∼1f/s
	Resolution	-	1382 x 512	640 x 480	-	960 x 540	352 x 240
	Publication	2011	2014	2016	2015	2015	2016

Table 2.1: Comparison of existing vehicle datasets. In the eighth row: "Count" is the vehicle counting task; "Det" is the detection task.

As there is no existing labeled real-world webcam traffic dataset, in order to evaluate our proposed method, we utilized existing traffic web cameras to collect continuous streams of street images and annotate the rich information. Different from currently existing traffic datasets, webcam data are challenging for analysis due to the low frame rate, low resolution, high occlusion, and large perspective. We select 212 representative web cameras from Manhattan, covering different locations, camera perspective, and traffic states. For each camera, we downloaded videos for four time intervals each day (7am-8am, 12pm-1pm; 3pm-4pm; 6pm-7pm). These cameras have frame rates around 1 frame/second and resolution 352×240. Collecting such data for 4 weeks generated 1.4 Terabytes of video data consisting of 60 million frames. To the best of our knowledge, CityCam is the first and largest annotated webcam traffic dataset to date. See Table 2.1 for a detailed comparison with other benchmark datasets.

Among these 60 million frames, we carefully select $60,000$ frames from 16 cameras to label them with rich information. To study the correlations among different cameras, we select representative cameras spreading over the Manhattan borough to label. Some of the cameras are close to each other, such as on the same street, while some of the cameras are quite far from each other. For each selected camera, we also carefully select the frames to cover different time of the day and different weather condition. The rich annotations include:

1. Bounding box: closest rectangle around each vehicle.
2. Vehicle type: ten types including taxi, black sedan, other cars, little truck, middle truck, big truck, van, middle bus, big bus, other vehicles.
3. Orientation: each vehicle orientation into four categories: $0°$, $90°$, $180°$, and $270°$.
4. Vehicle density: the number of vehicles in the target region of each frame.
5. Re-identification: we match the same car in sequential frames.
6. Weather: five types of weather, including sunny, cloudy, rainy, snowy, and intensive sunshine.

The results of the annotation for two successive frames is shown in Figure 2.2. The dataset is divided into training and testing sets, with 32,000 and 28,000 frames, respectively. To reduce the chance of overfitting, as well as to ensure generalization from training to testing phases, we deliberately select training videos that are taken at different locations from the testing videos, but ensure the training and testing videos share similar traffic conditions and attributes.

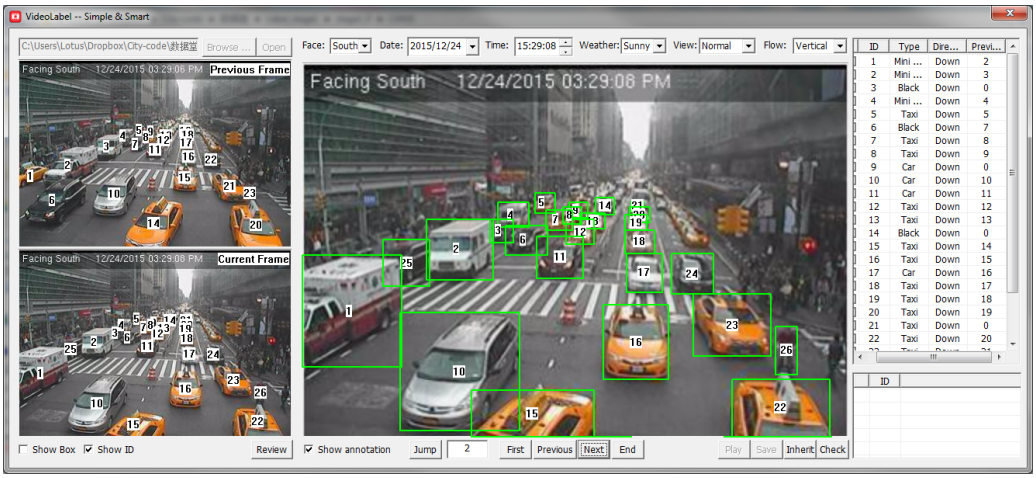


Figure 2.2: Annotation Instance.

CityCam serves as an appropriate dataset to evaluate our proposed method. Unlike the car data set KITTI[37] and Detrac[112] that focus on vehicle models, our dataset emphasizes real world traffic network analysis in a large metropolis. This dataset has three benefits: 1. It motivates research on vision based traffic flow analysis, posing new challenges for state-of-the-art algorithms. 2. With various street scenes, it can serve as benchmark for transfer learning and domain adaptation. 3. With large amount of labeled data, it provides training set for various learning based models, especially for deep learning techniques. Further details are provided in Appendix B.3.

Part II

In this part we describe three methodologies for vehicle counting for cameras with fully annotated data.

- Chapter 3 Optimization Based Vehicle Density Estimation with Rank Constraint
- Chapter 4 FCN Based Multi-Task Learning for Vehicle Counting
- Chapter 5 Deep Spatio-Temporal Neural Networks for Vehicle Counting

Chapter 3

Optimization Based Vehicle Density Estimation with Rank Constraints

3.1 Introduction

This Chapter considers our initial approach to the problem. To avoid detecting or tracking individual vehicles, we proposed a block-level regression model with a rank constraint, which maps the dense (pixel-level or block-level) image feature into vehicle densities. It learns different weights for different blocks of the image, increasing the number of degrees of freedom on the weights, and embeds perspective information. The framework is shown in Figure 3.1. We first divide the selected target region into blocks, extract features for each block, and perform background subtraction. Then, we linearly map each block's feature $\mathbf{x}_b$ into vehicle density $\text{Den}_b = \mathbf{w}_b^\top \mathbf{x}_b$, as illustrated in Figure 3.2. To avoid large errors induced by large perspective, we build one regressor per block with different weight $\mathbf{w}_b$, and learn the optimal weights. We stack all the weight vectors into a single weight matrix $\mathbf{W} = \left[\mathbf{w}_1^\top;\ \ \mathbf{w}_2^\top;\ \ ...\ \ \mathbf{w}_B^\top\right]$. To handle high dimensionality and capture the correlations among weight vectors of different blocks, we impose a low rank constraint on the weight matrix $\mathbf{W}$. Figure 3.2 illustrates the approach. Due to large perspective, vehicles have different scales in block A and block C, and their corresponding vehicle densities are also different. To reflect this fact, we assign different weights $\mathbf{w}_A$ and $\mathbf{w}_C$ to blocks A and C. Yet for blocks A and B, as they are at the same

distance from the camera, the vehicle scales are similar in these blocks and their corresponding vehicle densities are also similar. We build the weight matrix $\mathbf{W}$ to reflect both the diversity and the correlation among weight vectors.

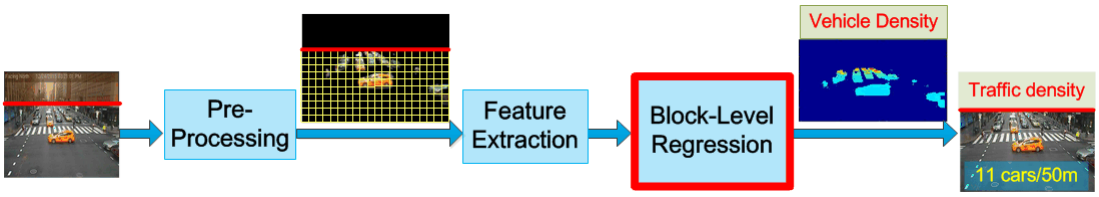


Figure 3.1: Block diagram for OPT-RC.

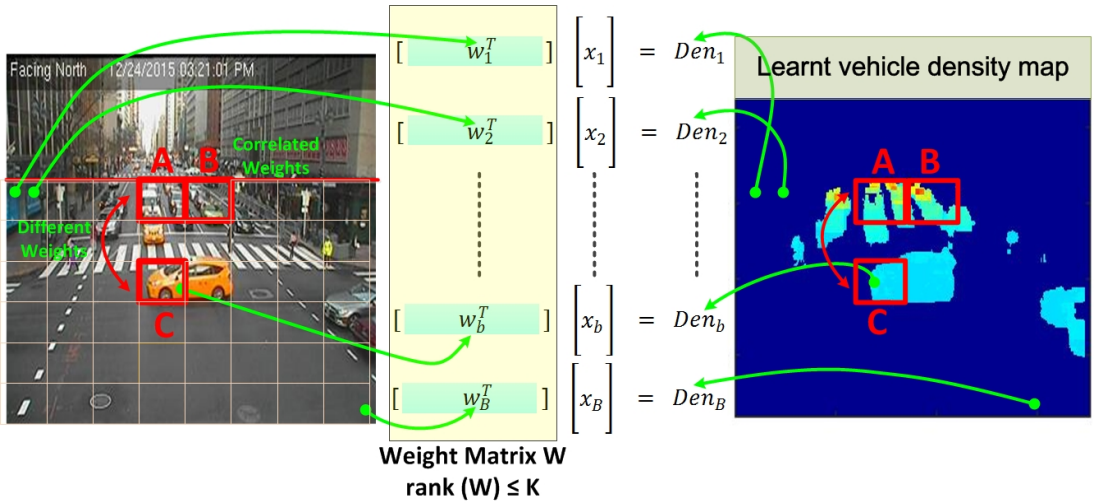


Figure 3.2: Intuition for OPT-RC. To reflect differences in perspective, different blocks have different weights. A rank constraint is added on the stacked weight matrix.

3.2 Problem Formulation and Optimization

We assume a set of N training images $I_1, \cdots, I_N$ is given. We first perform foreground/background segmentation based on GrabCut [15]. We automate the segmentation process by generating a pure background image, and initialize the background and foreground based on the difference between the input frame and the pure background frame. For a stream of images, we select a region of interest and divide it into J blocks. The block size can vary from 16×16 to 1×1, depending on the width of the lane and the dimensions of the smallest vehicle. A block $B_j^{(i)}$ in each image I_i is represented by a feature vector $\mathbf{x}_j^{(i)} \in \mathbb{R}^K$. Examples of particular choices of features are given in the experimental section. To

generate the ground truth density map, vehicle types, and bounding boxes, each training image I_i is annotated with a set of 2D bounding boxes, centered at pixels $P_i = \{P_1, \cdots, P_k, \cdots, P_{c(i)}\}$, where $c(i)$ is the total number of annotated vehicles in the i-th image. Each bounding box k is annotated with a vehicle type T_k. For a training image I_i, we calculate the ground truth density based on the labeled bounding boxes (shown in Figure 2.2). The pixel p covered by a set of bounding boxes $O(p)$ has density $D(p)$ defined as

$$D_i(p) = \sum_{o \in O(p)} \frac{1}{A(o)} \tag{3.1}$$

where $A(o)$ denotes the area of bounding box o. Then, we define the density $D(B)$ of a block as

$$D(B) = \sum_{p \in B} D(p) \tag{3.2}$$

Given a set of training images with their ground truth densities, for each block B_j, we learn a block-specific linear regression model that predicts block-level density $\widehat{D}(B_j)$ given its feature $\mathbf{x}_j$ by

$$\widehat{D}(B_j) = \mathbf{w}_j^\top \mathbf{x}_j \tag{3.3}$$

where $\mathbf{w}_j \in \mathbb{R}^K$ is the coefficient vector of the linear regression model to be learned for block j. We assign different weights to different blocks. To capture the correlations and commonalities among the regression weight vectors at different blocks, we encourage these vectors to share a low-rank structure. To avoid overfitting, we add ℓ_2-regularization to these weight vectors. To encourage sparsity of the weights, ℓ_1-regularization is imposed.

Let $\mathbf{W} \in \mathbb{R}^{K \times J}$, where the j-th column vector $\mathbf{w}_j$ denotes the regression coefficient vector of block j. To this end, we define the following regularized linear regression model with low-rank constraint,

$$\begin{aligned} \min_{\mathbf{W}} \ & \tfrac{1}{2N} \sum_{i=1}^{N} \sum_{j=1}^{J} \left(\mathbf{w}_j^\top \mathbf{x}_j^{(i)} - D\left(B_j^{(i)}\right) \right)^2 + \alpha \|\mathbf{W}\|_F^2 + \beta |W|_1 \\ \text{s.t.} \ & \ \text{rank}(\mathbf{W}) \leq r \end{aligned} \tag{3.4}$$

To solve this rank-constrained problem that has a non-smooth objective function, we develop an accelerated projected subgradient descent (APSD) algorithm outlined in Algorithm 1, which iteratively performs subgradient descent, rank projection, and acceleration. Two sequences of variables $\{\mathbf{A}_k\}$ and $\{\mathbf{W}_k\}$ are maintained for the purpose of performing acceleration.

Subgradient Descent. Subgradient descent is performed over variable $\mathbf{A}_k$. We first compute the subgradient $\triangle\mathbf{A}_k$ of the non-smooth objective function $\frac{1}{2N}\sum_{i=1}^{N}\sum_{j=1}^{J}\left(\mathbf{a}_j^\top \mathbf{x}_j^{(i)} - D\left(B_j^{(i)}\right)\right)^2 + \alpha\|\mathbf{A}\|_F^2 + \beta|\mathbf{A}|_1$, where the first and second terms are smooth, hence their subgradients are simply gradients. For the third term $|\mathbf{A}|_1$ which is non-smooth, its subgradient $\partial\mathbf{A}$ can be computed as

$$\partial A_{ij} = \begin{cases} +1 & \text{if } A_{ij} \geq 0 \\ -1 & \text{if } A_{ij} < 0 \end{cases} \tag{3.5}$$

Adding the subgradients of the three terms, we obtain the subgradient $\triangle\mathbf{A}_k$ of the overall objective function. Then $\mathbf{A}_k$ is updated by

$$\mathbf{A}_k \leftarrow \mathbf{A}_k - t_k \triangle \mathbf{A}_k \tag{3.6}$$

Rank Projection. In lines 4-5 of Algorithm 1, we project the newly obtained $\mathbf{A}_k$ to the feasible set $\{\mathbf{W}|\text{rank}(\mathbf{W}) \leq r\}$, which amounts to solving the following problem

$$\begin{aligned} \underset{\mathbf{W}}{min} \quad & \|\mathbf{A}_k - \mathbf{W}\|_F^2 \\ \text{s.t.} \quad & \text{rank}(\mathbf{W}) \leq r \end{aligned} \tag{3.7}$$

According to [32], the optimal solution $\mathbf{W}^*$ can be obtained by first computing the largest r singular values and singular vectors of $\mathbf{A}_k$: $\mathbf{U}_r$, $\mathbf{\Sigma}_r$, $\mathbf{V}_r$, then setting $\mathbf{W}^*$ to $\mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^H$.

Acceleration. In lines 6-7 of Algorithm 1, acceleration is performed by updating the step size according to the following rule: $t_{k+1} \leftarrow \frac{1}{2}(1 + \sqrt{1 + 4t_k^2})$, and adding a scaled difference between consecutive $\mathbf{W}$s to $\mathbf{A}$.

It is worth noting that this optimization problem is not convex and the APSD algorithm may lead to local optimal. It helps to run the algorithm multiple times with different random initializations.

3.3 Experimental Results

We performed extensive experiments on different datasets and tasks to verify the efficacy of the proposed method: 1. We first evaluate and compare OPT-RC with state-of-the-art methods on the

Algorithm 1 Accelerated Projected Subgradient Descent

Input: Data $\mathcal{D}$, rank r, regularization parameters α, β

while not converged **do**
  Compute gradient $\triangle \mathbf{A}_k$
  $\mathbf{A}_k \leftarrow \mathbf{A}_k - t_k \triangle \mathbf{A}_k$
  Compute top r singular values and vectors of $\mathbf{A}_k$: $\mathbf{U}_r$, $\mathbf{\Sigma}_r$, $\mathbf{V}_r$
  $\mathbf{W}_{k+1} \leftarrow \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^\top$
  $t_{k+1} \leftarrow \frac{1}{2}(1 + \sqrt{1 + 4t_k^2})$
  $\mathbf{A}_{k+1} \leftarrow \mathbf{A}_k \frac{t_{k-1}-1}{t_k}(\mathbf{W}_k - \mathbf{W}_{k-1})$
end while

Output: $\boldsymbol{W} \leftarrow \boldsymbol{W}_{k+1}$

CityCam; 2. We then evaluate OPT-RC on the public dataset TRANCOS[79]; and, finally, 3. to verify the robustness and generalization of our model, we evaluate our method on the public pedestrian counting dataset UCSD [17].

1) Quantitative Evaluations on CityCam

To evaluate our method, we selected 48 videos from 6 cameras in CityCam, containing 14,150 frames and covering different scenes, congestion states, camera perspectives, weather and time of the day. Each video has 352×240 resolution, and frame rate around 1 frame/sec. The training set contains 20,000 frames from different videos with the same resolution. Two metrics are employed for evaluation: 1. Mean absolute error (MAE); and 2. Mean square error(MSE). To compare OPT-RC with the baseline methods, we extract SIFT features [67] from each block. Other features can also be exploited in the proposed framework. The block size is determined by the width of the lane and the length of the smallest vehicle. Parameters in (3) are selected by cross-validation.

Baseline approaches: We compare our method with two methods: *Baseline 1: Learning to count* [103]. This work maps each pixel's feature into object density with uniform weight for the whole image. For comparison, we extract dense SIFT features [67] for each pixel using VLFeat [108]. The ground truth density is computed as a normalized 2D Gaussian kernel based on the center of each labeled bounding box. *Baseline 2: Hydra-3s*[79]. It stacks feature maps generated by different networks and combine them to generate the final prediction of the density map. We train Hydra-3s on the same training set as FCN-MT.

Experimental Results: Below, we compare the errors of the proposed and baseline approaches given

Method	MAE	
	Downtown	Parkway
Baseline 1	5.91	5.19
Baseline 2	4.42	4.21
OPT-RC	4.56	4.24

Table 3.1: Accuracy comparison on CityCam

in Table 3.1. From these results, we conclude that OPT-RC outperforms the non-deep learning based Baseline 1 and shows comparable results with Baseline 2, but require much less training data. As the testing data cover different congestion states, camera perspectives, weather conditions and time of the day, these results verify the generalization and robustness of OPT-RC. It can learn a smooth density map with geometry information. Figure 3.3 shows the original image (a), learned density map from Baseline1 (b), and learned density map from OPT-RC (c). From Figure 3.3, we see that the density map from Baseline 1 can not handle well the large perspective present in the video, while the density map from our method captures well the camera perspective.

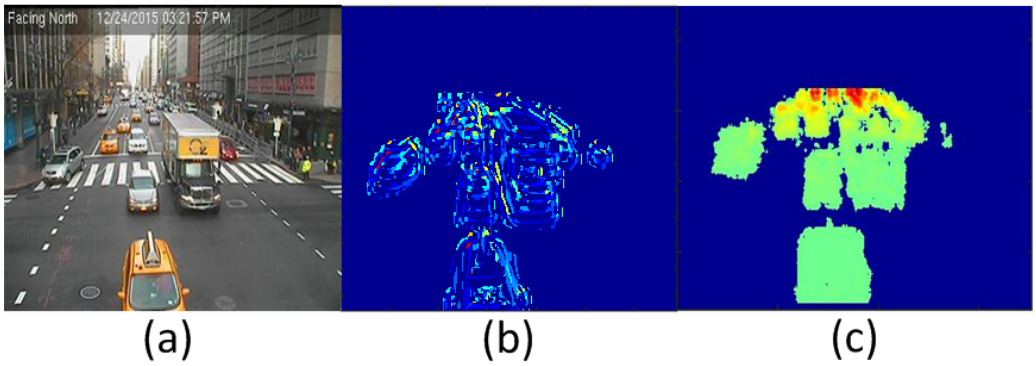


Figure 3.3: Comparison of OPT-RC and Baseline 1. (a) Original image, (b) learned density map from Baseline1, and (c) learned density map from OPT-RC.

2) Quantitative Evaluations on TRANCOS

To verify the efficacy of our OPT-RC method, we also evaluate and compare it with baselines on a public dataset, TRANCOS[79]. TRANCOS provides a collection of 1244 images of different traffic scenes, obtained from real video surveillance cameras, with a total of 46796 annotated vehicles. The objects have been manually annotated using dots. It also provides a region of interest (ROI) per image, defining the region considered for the evaluation. This database provides images from very different

Method	Baseline 1	Baseline 2-CCNN	Baseline 2-Hydra	OPT-RC
MAE	13.76	12.49	10.99	12.41

Table 3.2: Results comparison on TRANCOS dataset

scenarios but no perspective maps are provided. The ground truth object density maps are generated by placing a Gaussian Kernel in the center of each annotated object[44].

We compare our OPT-RC with baselines in Table 3.2. Baseline 1 and OPT-RC have the same settings as evaluated in CityCam. Baseline 2-CCNN is a basic version of the network in [79], and Baseline 2-Hydra augments the performance by learning a multiscale non-linear regression model with a pyramid of image patches extracted at multiple scales to perform the final density prediction. Baseline 2-CCNN, and Baseline 2-Hydra are trained on 823 images and tested on 421 images following the separation in [79]. From Table 3.2, we conclude that OPT-RC outperforms Baseline 1 and obtains results comparable to Baseline 2.

3) Quantitative Evaluations on the UCSD Dataset

To verify the generalization and robustness of our OPT-RC in different counting tasks, we also evaluate and compare it with baselines on the pedestrian counting dataset UCSD[17]. This dataset contains 2000 frames chosen from one surveillance camera. The average number of people in each frame is around 25. The dataset provides the ROI for each video frame.

By following the same setting as in [17], we use frames from 601 to 1400 as training data, and the remaining 1200 frames as test data. OPT-RC has the same setting as evaluated in CityCam. Table 3.3 shows the results of our methods and existing methods. From the results we can see that OPT-RC outperforms non-deep learning based methods, but it is less accurate than the deep learning-based methods in[117]. These results show that although our methods are developed to solve the challenges in webcam video data, they perform well with other type of counting tasks.

Method	MAE	MSE
Kernel Ridge Regression [2]	2.16	7.45
Ridge Regression [19]	2.25	7.82
Gaussian Process Regression [17]	2.24	7.97
Cumulative Attribute Regression [20]	2.07	6.86
Cross-scene DNN[117]	1.6	3.31
OPT-RC	2.03	5.97

Table 3.3: Results comparison on UCSD dataset

From the extensive evaluations on different datasets and counting tasks, OPT-RC shows promising performance on vehicle density estimation, especially in reducing large errors induced by camera perspective. But it also suffers from limitations. In the next Chapter, we will propose several techniques to overcome these limitations.

Chapter 4

FCN Based Multi-Task Learning for Vehicle Counting

4.1 Introduction

The block-level optimization method OPT-RC described in Section 3 avoids individual vehicle detection and tracking by mapping dense image features into vehicle densities. It embeds road geometry in the weight matrix and achieves promising results. The evaluation of OPT-RC led to the following insights:

1. It verifies the efficacy of mapping the dense features into vehicle density for vehicle counting;
2. Accounting for camera perspective is important to reduce the counting error; and
3. It reveals the necessity of considering correlation of nearby pixels.

To further improve the system scalability and robustness, we propose to pursue a more generalized model that can address the following challenging and crucial problems:

1. Extract both representative and discriminative dense features;
2. Understand urban traffic with additional rich information, such as vehicle count and type;
3. Incorporate temporal information of the traffic videos;
4. Adapt the model to multiple cameras and different environmental conditions; and
5. Understand different camera perspectives.

With the hierarchical feature learning and state-of-the-art performance exhibited by deep neural

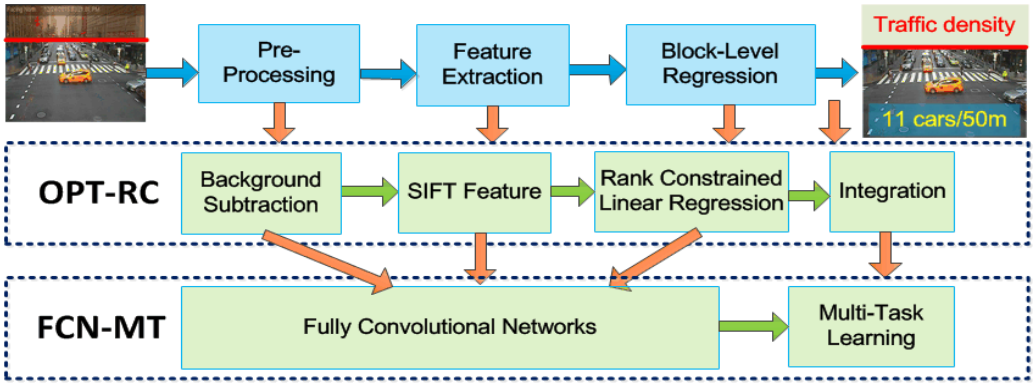


Figure 4.1: Comparison of OPT-RC and FCN-MT. Both methods share the idea of mapping dense features into vehicle densities. FCN-MT replaces the background subtraction, feature extraction, and linear regression with fully convolution networks.

networks (as shown in the experiments of Chapter 3), we propose a deep multi-task model to replace the linear regression model with fully convolutional regression networks (illustrated in Figure. 4.1). To address the above problems, we developed CityScapeEye:

1. a deep multi-task learning framework based on fully convolutional neural networks (FCN-MT) to jointly learn vehicle density and vehicle counts;
2. deep spatio-temporal networks to incorporate temporal information of traffic flow;
3. a multi-domain adversarial training mechanism to adapt the counting model to different cameras.

Each proposed technique will be explained in this and the following chapters. In this Chapter, we mainly introduce the deep multi-task learning model based on fully convolutional regression networks. FCN-MT jointly estimates vehicle density and vehicle count with end-to-end dense prediction, allowing for arbitrary input image resolution. To avoid large errors induced by vehicle scale variations, we jointly perform image-level vehicle count regression and pixel-level vehicle density estimation. The global count regression takes the estimated density map as input feature, but instead of directly regressing the global vehicle count from the last feature map, we develop a residual learning framework to reformulate global count regression as learning residual functions with reference to the sum of densities in each frame. Such design avoids learning unreferenced functions and eases the training of the network. Evaluation verifies the efficacy of the deep model and shows state-of-the-art performance. The MAE on the testing data is reduced from 4.56 to 2.74, compared with OPT-RC.

4.2 Deep Multi-Task Learning of Vehicle Density with FCN

To overcome the limitations of OPT-RC in Chapter 3), we propose a deep multi-task model to jointly learn vehicle density and vehicle counts, as illustrated in Fig. 4.2. It replaces the background subtraction, feature extraction, and linear regression in OPT-RC with deep neural networks. To produce density maps that have the same size as the input image, we design a FCN [65] to perform pixel-wise prediction whole-image-at-a-time by dense feedforward computation and backpropagation. Global vehicle count is regressed in a residual learning manner from the estimated density map. The overall structure of our proposed FCN-MT is illustrated in Figure 4.2, which contains a convolution network, deconvolution network, feature combination and selection, and multi-task learning.

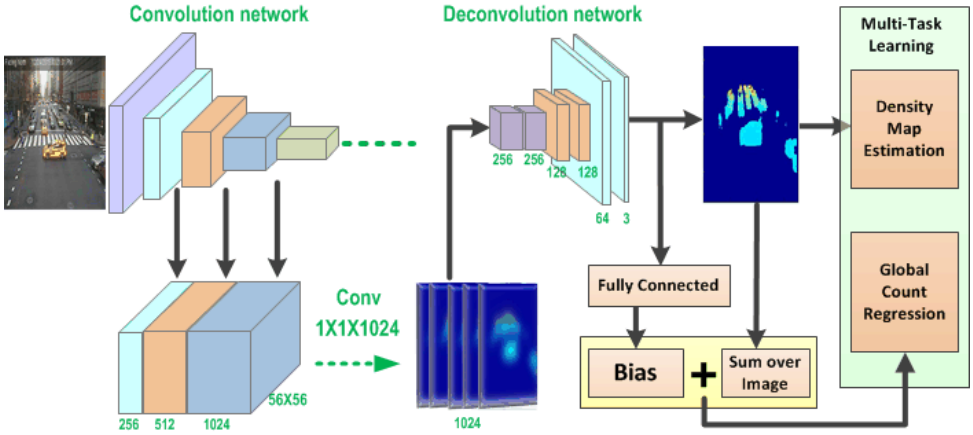


Figure 4.2: Framework of FCN-MT. Jointly learn vehicle density and count.

FCN-based vehicle counting and detection faces two challenges: 1. variation of vehicle scales; and 2. reduced feature resolution[21]. To avoid large errors induced by scale variations, we jointly perform global count regression and density estimation. Inspired by ResNets [45], instead of directly regressing the global vehicle count from the last feature map, we develop a residual learning framework to reformulate global count regression as learning residual functions with reference to the sum of densities. Such design avoids learning unreferenced functions and eases the training of the network. The second challenge is caused by the repeated combination of max-pooling and striding for the DCNNs that are originally designed for image classification[61],[88],[94]. This results in feature maps with significantly reduced spatial resolution. To solve this problem, we produce denser feature maps by combining appearance features from shallow layers with semantic features from deep layers. We

then add a convolution layer after the combined feature volume with 1×1 kernels to perform feature selection. The selected features can better distinguish foreground and background. Thus the whole network is able to accurately estimate vehicle density without foreground segmentation.

4.2.1 Network Architecture

The overall structure of our proposed FCN-MT is illustrated in Figure 4.2, which contains a convolution network, deconvolution network, feature combination and selection, and multi-task learning. The convolution network is based on pre-trained ResNets[45]. The accuracy of our approach can be improved by 8% using ResNets compared with using VGG-net[88]. The pixel-wise density estimation requires high feature resolution, yet the pooling and striding can reduce feature resolution significantly. To solve this problem, we rescale and combine the features from the $2a$, $3a$, and $4a$ layers of ResNets. We then add a convolution layer after the combined feature volume with 1×1 kernels to perform feature selection. By learning the parameters of this layer, the selected features can better distinguish foreground and background pixels. We input the combined and selected feature volume into the deconvolution network, which contains five deconvolution layers. Inspired by the deep VGG-net[88], we apply small kernels of 3×3 in the deconvolution layers. The feature is mapped back to the image size by deconvolution layers, whose parameters can be learned from the training process[78]. A drawback of deconvolution layer is that it may have uneven overlap when the kernel size is not divisible by the stride. We add one convolution layer with 3×3 kernels to smooth the checkerboard artifacts and alleviate this problem. Then, one convolution layer with $1 \times 1 \times 1$ kernels is added to map the feature maps into density map; one convolution layer with $1 \times 1 \times 4$ kernels is added to map the feature maps into predicted location offsets; and another convolution layer with $1 \times 1 \times 1$ kernels is added to map the feature maps into the vehicle type confidence score.

4.2.2 Multi-Task Learning Building Blocks

Vehicle Density Estimation: At the last stage of the network, we jointly learn vehicle density, vehicle count, and vehicle type. The vehicle density is predicted by the last convolution 1×1 layer from the feature map. We adopt Euclidean distance to measure the difference between the estimated density and

the ground truth. The loss function for density map estimation is defined as follows:

$$L_D(\Theta) = \frac{1}{2N} \sum_{i=1}^{N} \sum_{p=1}^{P} \|F\left(X_i(p); \Theta\right) - F_i(p)\|_2^2 \tag{4.1}$$

where Θ is the parameter vector of the FCN-MT model, N is the number of training images, and $F_i(p)$ is the ground truth density of pixel p.

Global Count Regression: We reformulate the second task, global count regression, as learning residual functions with reference to the sum of densities, which consists of two parts: 1. base count: the integration of the density map over the whole image; and 2. offset count: predicted by two fully connected layers from the feature map after the convolution 3×3 layer of the deconvolution network. We sum the two parts together to get the estimated vehicle count, as shown in the following equation:

$$C(i) = B(D(i); \gamma) + \sum_{p=1}^{P} D(i, p) \tag{4.2}$$

where γ is the learnable parameters of the two fully connected layers, $B(D(i); \gamma)$ is the learned bias, and $D(i, p)$ indicates the density of each pixel p in image i. We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. Considering that the vehicle count for some frames may have very large value, we adopt a Huber loss to measure the difference between the estimated count and the ground truth count. The count loss for one frame is defined as follows:

$$L_\delta(i) = \begin{cases} \frac{1}{2}\left(C(i) - C_t(i)\right)^2 & \text{for } |C(i) - C_t(i)| \leq \delta \\ \delta\,|C(i) - C_t(i)| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases} \tag{4.3}$$

where $C_t(i)$ is the ground truth vehicle count of frame i, $C(i)$ is the estimated loss of frame i, and δ is the threshold to control the outlier in the training sets.

Then, the overall loss function for the network is defined as:

$$L = L_D + \lambda \sum_{i=1}^{N} L_\delta(i) \tag{4.4}$$

where λ is the weight of global count loss. By simultaneously learning the three related tasks, each task can be better trained with fewer parameters.

Method	MAE	
	Downtown	Parkway
Baseline 1	5.91	5.19
Baseline 2	4.42	4.21
OPT-RC	4.56	4.24
FCN-MT	2.74	2.52

Table 4.1: Accuracy comparison on CityCam

4.3 Experiments

Experiments on different datasets and tasks verify the efficacy of the proposed methods: 1. We evaluate and compare the preliminary implementation of FCN-MT with state-of-the-art methods on dataset CityCam. 2. We evaluate the preliminary implementation of FCN-MT on the public dataset TRANCOS[79]. 3. To verify the generalization of the proposed model, we evaluate our methods on the pedestrian counting dataset UCSD [17].

Results on CityCam: The experimental settings and baseline methods are the same as in Cahpter 3. For FCN-MT, we divide the training data into two groups: downtown and parkway. Then in each group, we balance the training frames with less than 15 vehicles and the frames with more than 15 vehicles. FCN-MT is tested on all the test data of CityCam. The network architecture is shown in Figure 4.2. The weight of the vehicle count loss is 0.01. We compare the errors of the proposed and baseline approaches in Table 4.1. From these results, we see that FCN-MT outperforms the baseline approaches and OPT-RC of Cahpter 3 for all the tests. As the testing data cover different congestion states, camera perspectives, weather conditions; and time of the day, these results verify the generalization and robustness of FCN-MT. Figure 4.3 shows the density map learned from FCN-MT. Without foreground segmentation, the learned density map can still distinguish background from foreground in sunny and cloudy, dense and sparse scenes. Yet due to the uneven overlaps of the deconvolution layers, checkerboard artifacts may be created on the learned density map.

Results on the TRANCOS Dataset: We evaluate and compare the preliminary counting results of FCN-MT with baselines on a public dataset, TRANCOS[79]. The dataset and experimental settings are the same as in Chapter 3. From the results shown in Table 4.2, we see that the preliminary FCN-MT

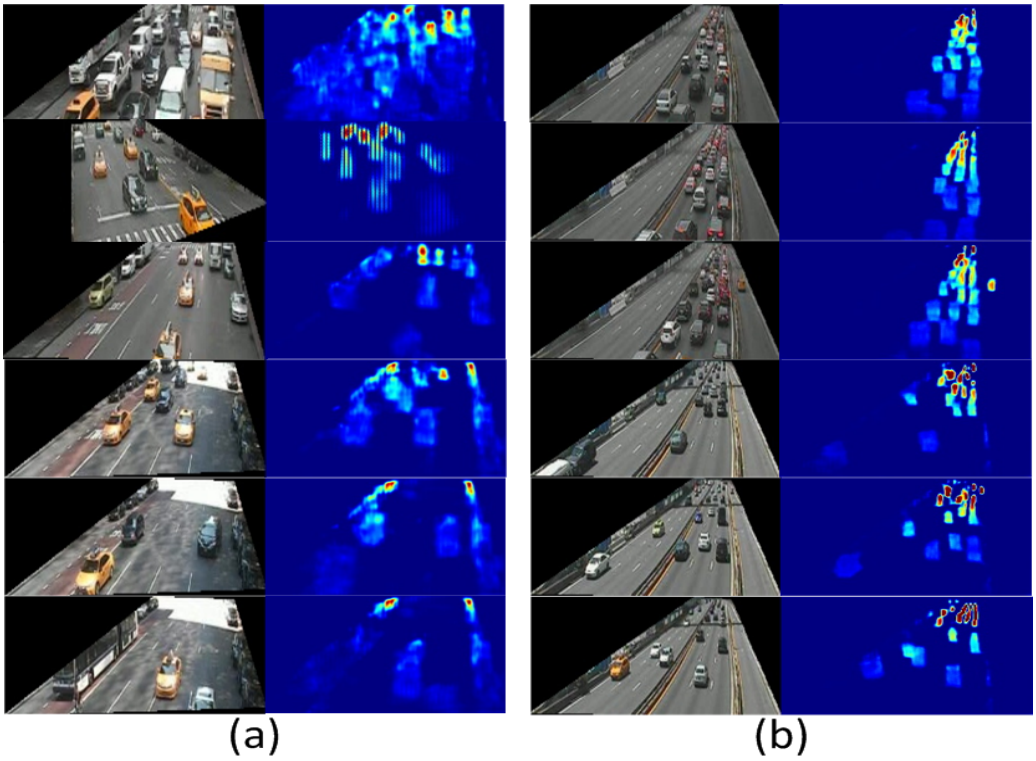


Figure 4.3: Density map from FCN-MT: (a) Downtown; (b) Parkway. Top three rows are cloudy days; Bottom three rows are sunny days.

Method	Baseline 1	Baseline 2-CCNN	Baseline 2-Hydra	OPT-RC	FCN-MT
MAE	13.76	12.49	10.99	12.41	5.31

Table 4.2: Results on TRANCOS dataset

significantly improves the MAE from 10.99 to 5.31 compared with Baseline 2.

Results on UCSD Dataset: To verify the generalization of the proposed methods on different counting tasks, we also evaluate the preliminary FCN-MT on the crowd counting dataset UCSD[17]. The dataset and experimental settings are the same as in Chapter 3. Table 4.3 shows the results of FCN-MT and existing methods. FCN-MT outperforms all the non-deep learning based methods and gets accuracy similar to that of the deep learning-based methods in[117]. These results show the robustness of our methods to other type of tasks.

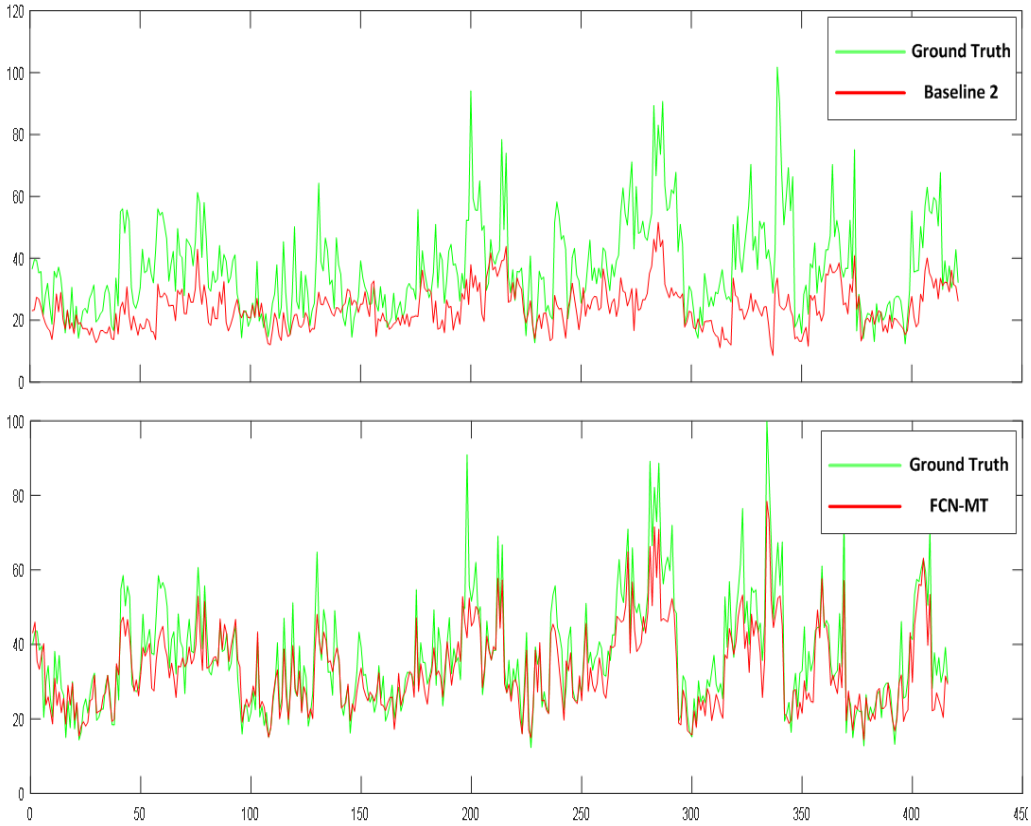


Figure 4.4: Comparing FCN-MT and Baseline 2. Please zoom to view better.

Method	MAE	MSE
Kernel Ridge Regression [2]	2.16	7.45
Ridge Regression [19]	2.25	7.82
Gaussian Process Regression [17]	2.24	7.97
Cumulative Attribute Regression [20]	2.07	6.86
Cross-scene DNN[117]	1.6	3.31
OPT-RC	2.03	5.97
FCN-MT	1.67	3.41

Table 4.3: Results on UCSD dataset

Chapter 5

Deep Spatio-Temporal Neural Networks for Vehicle Counting

5.1 Introduction

From the experiments we see that the FCN-MT in Chapter 4 achieves promising performance. To explore the temporal correlation of traffic flow, we design a novel FCN-rLSTM network to jointly estimate vehicle density and vehicle count by connecting FCN with long short term memory networks (LSTM) in a residual learning fashion. Such design leverages the strengths of FCN for pixel-level prediction and the strengths of LSTM for learning complex temporal dynamics. The residual learning connection reformulates the vehicle count regression as learning residual functions with reference to the sum of densities in each frame, which significantly accelerates the training of networks. To preserve feature map resolution, we propose a Hyper-Atrous combination to integrate atrous convolution in FCN and combine feature maps of different convolution layers. FCN-rLSTM enables refined feature representation and a novel end-to-end trainable mapping from pixels to vehicle count. We extensively evaluated the proposed method on different counting tasks with three datasets, with experimental results demonstrating their effectiveness and robustness. In particular, FCN-rLSTM reduces the mean absolute error (MAE) from 5.31 to 4.21 on TRANCOS; and it reduces the MAE from 2.74 to 1.53 on CityCam. The training process is accelerated by 5 times on average.

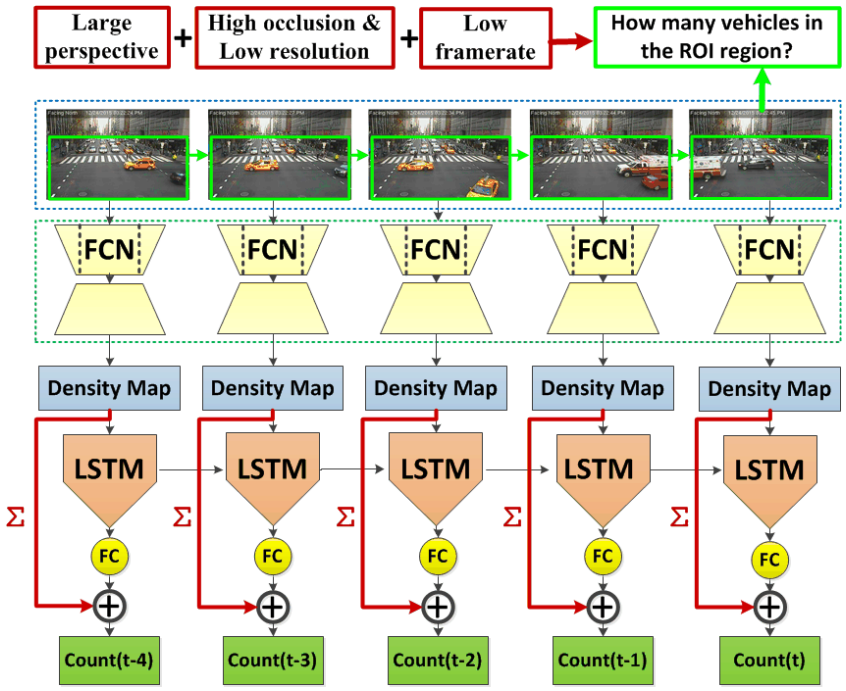


Figure 5.1: FCN-rLSTM network to count vehicles in traffic videos captured by city cameras. Video frames are input into FCN, and the output density maps are fed into a stack of LSTMs to learn residual functions with reference to the sum of densities in each frame. The global vehicle count is finally generated by summing the learned residual and the densities.

The challenges of webcam videos preclude existing approaches to vehicle counting, which can be grouped into five categories: frame differencing based [28, 98], detection based [97, 124], motion based [23, 24, 73, 93], density estimation based [103], and deep learning based [5, 49, 79, 89, 117, 122, 123] methods. The first three groups of methods are sensitive to environment conditions and tend to fail in high occlusion, low resolution, and low frame rate videos. While density estimation approaches avoid detecting or tracking individual vehicles, they perform poorly in videos with large perspective and oversized vehicles. Though the low frame rate webcam video lacks motion information, vehicle counts of sequential frames are still correlated. Existing methods fail to account for such temporal correlation [25, 36, 50, 79, 115, 121]. Work [5] and our own work [121] achieve state-of-the-art performance on animal counting and traffic counting, respectively, yet they fail to model the temporal correlation as an intrinsic feature of the surveillance video. In recent years, several works attempt to combine CNN with recurrent neural networks (RNN) [7] to model the spatio-temporal information of visual tasks, such as action recognition [6, 31], video description [31], caption generation [92], and multi-label

classification [111]. However, no existing work models the spatio-temporal correlation for object counting, especially by combining CNN/FCN with RNN/LSTM. Some work [76] explores a new design of the internal LSTM architecture, but none of the existing works combined FCN with LSTM in a residual learning fashion. Work [118] regards the crowd flow map in a city as an image and builds spatio-temporal networks to predict crowd flow. It does not apply RNN or LSTM networks to learn the temporal information; instead, it aggregates the output of three residual neural networks to model temporal dynamics. Thus such work is essentially multiple convolutional neural networks, rather than the combination of CNN and LSTM.

To overcome these limitations, we propose a deep spatio-temporal network architecture to sequentially estimate vehicle count by combining FCN [65] with LSTM [47] in a residual learning framework (FCN-rLSTM). The FCN maps pixel-level features into vehicle density to avoid individual vehicle detection or tracking. LSTM layers learn complex temporal dynamics by incorporating nonlinearities into the network state updates. The residual connection of FCN and LSTM reformulates global count regression as learning residual functions with reference to the sum of densities in each frame, avoiding learning unreferenced functions and significantly accelerating the network training. FCN-rLSTM enables refined feature representation and a novel end-to-end optimizable mapping from image pixels to vehicle count. The framework is shown in Figure 5.1. Video frames are input into FCN, and the output density maps are fed into LSTMs to learn the vehicle count residual for each frame. The global vehicle count is finally generated by summing the learned residual and the densities.

The proposed FCN-rLSTM has the following novelties and contributions:

1. FCN-rLSTM is a novel spatio-temporal network architecture for object counting, such as crowd counting [117], vehicle counting [79], and penguin counting [5]. It leverages the strength of FCN for dense visual prediction and the strengths of LSTM for learning temporal dynamics. Such network can learn from more information. To the best of our knowledge, FCN-rLSTM is the first spatio-temporal network architecture with residual connection for object counting.
2. The residual connection between FCN and LSTM is novel and significantly accelerates the training process by 5 times on average, as shown by our experiments. Though some recent work

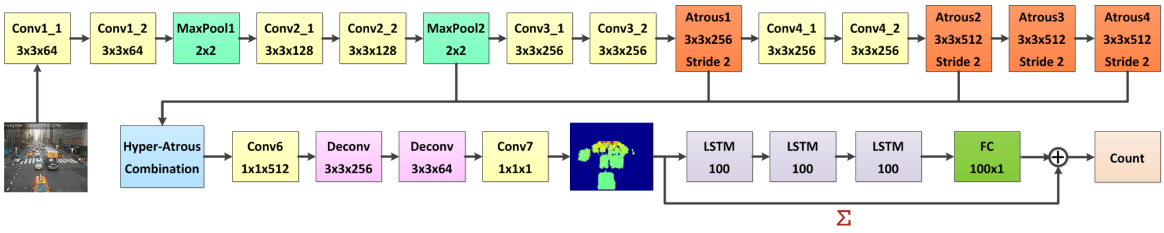


Figure 5.2: Network architecture and parameters of FCN-rLSTM.

on other visual tasks [31, 111] also explored spatio-temporal networks, none of them combines FCN with LSTM, or has the residual connection between CNN and LSTM. FCN-rLSTM can be potentially applied to other visual tasks that both require dense prediction and exhibit temporal correlation.

3. One challenge for FCN based visual tasks is the reduced feature resolution [21] caused by the repeated max-pooling and striding. To solve this problem, we propose a Hyper-Atrous combination to integrate atrous convolution [21] in the FCN and to combine feature maps of different atrous convolution layers. We then add a convolution layer after the combined feature volume with 1×1 kernels to perform feature re-weighting. The selected features both preserve feature resolution and distinguish better foreground from background. Thus, the whole network accurately estimates vehicle density without foreground segmentation.
4. We jointly learn vehicle density and vehicle count from end-to-end trainable networks improving the accuracy of both tasks. Recent object counting literature [5, 79, 122, 123] estimates directly the object density map and sum the densities over the whole image to get the object count. But such methods suffer from large error when videos have large perspective and oversized vehicles (big bus or big truck). Our proposed multi-task framework pursues different but related objectives to achieve better local optimum, to provide more supervised information (both vehicle density and vehicle count) in the training process, and to learn better feature representation.
5. We present comprehensive experiments on three datasets covering different counting tasks, such as vehicle counting and crowd counting, to show generalization and substantially higher accuracy of FCN-rLSTM. On the TRANCOS dataset [79], we improve over state-of-the-art baseline methods, reducing the MAE from 5.31 to 4.21.

5.2 FCN-rLSTM Model and Network Architecture

As the low spatial and temporal resolution and high occlusion of webcam videos preclude existing detection or motion based methods for vehicle counting, we propose to apply FCN [65] to map the dense (pixel-level) feature into vehicle density and to avoid detecting or tracking individual vehicles. FCN based density estimation allows arbitrary input resolution and outputs vehicle density maps that are of the same size as the input image. Existing object counting literature [5, 79, 122, 123] estimates the object density map and directly sums the density over the whole image to get the object count. But such methods suffer from large error when the video has large perspective and oversized vehicles (big bus or big truck). Thus we propose the FCN-rLSTM network to jointly estimate vehicle density and vehicle count by connecting FCN with LSTM in a residual learning fashion. Such design leverages the strengths of FCN for pixel-level prediction and the strengths of LSTM for learning complex temporal dynamics. Counting accuracy is significantly improved by taking the temporal correlation of vehicle counts into account. However, it is not easy to train the combined FCN and LSTM networks. We further propose the residual connection of FCN and LSTM to accelerate the training process. The resulting end-to-end trainable network has high convergence rate and further improves the counting accuracy. In the following subsections, we will explain the proposed network architecture and highlight additional details.

The network architecture with detailed parameters is shown in Figure 5.2, which contains convolution network, deconvolution network, hyper-atrous feature combination, and LSTM layers. Inspired by the VGG-net [88], small kernels of size 3×3 are applied to both convolution layers and deconvolution layers. The number of filter channels in the higher layers are increased to compensate for the loss of spatial information caused by max pooling.

To preserve feature map resolution, we develop a hyper-atrous combination, where atrous convolution [21] is integrated into the convolution networks, and the feature maps after the second max-pooling layer and the atrous convolution layers are combined together into a deeper feature volume. Atrous convolution is proposed by [21]; it amounts to filter upsampling by inserting holes between nonzero filter taps. It computes feature maps more densely, followed by simple bilinear interpolation of the

feature responses back to the original image size. Compared to regular convolution, atrous convolution effectively enlarges the field of view of filters without increasing the number of parameters. After several atrous convolution layers, we combine the features from the second max-pooling layer and the atrous convolution layers. And then, after the combined feature volume, we add a convolution layer with 1×1 kernels to perform feature re-weighting to encourage the re-weighted feature volume to distinguish better foreground and background pixels. The combined and re-weighted feature volume is input of the deconvolution network that contains two deconvolution layers. At the top of the FCN, a convolution layer with 1×1 kernel acts as a regressor to map the features into vehicle density.

To incorporate the temporal correlation of vehicle counts from sequential frames, we combine LSTM with FCN to jointly learn vehicle density and count. RNN maintains internal hidden states to model the dynamic temporal behavior of sequences. LSTM extends RNN by adding three gates to an RNN neuron: a forget gate f_t; an input gate i_t; and an output gate o_t. These gates enable LSTM to learn long-term dependencies in a sequence, and make it easier to be optimized. LSTM effectively deals with the gradient vanishing/exploding issues that commonly appear during RNN training [81]. It also contains a cell activation vector c_t and a hidden output vector h_t. We reshape the output density map of FCN into a 1D vector x_t and feed this vector into three LSTM layers. Each LSTM layer has 100 hidden units and is unrolled for a window of 5 frames. The gates apply sigmoid nonlinearities σ_i, σ_f, σ_o, and tanh nonlinearities σ_c, and σ_h with weight parameters W_{hi}, W_{hf}, W_{ho}, W_{xi}, W_{xf}, and W_{xo}, which connect different inputs and gates with the memory cells, outputs, and biases b_i, b_f, and b_o. We define the commonly-used update equations [43]:

$$
\begin{aligned}
i_t &= \sigma_i(x_t W_{xi} + h_{t-1} W_{hi} + w_{ci} \odot c_{t-1} + b_i) \\
f_t &= \sigma_f(x_t W_{xf} + h_{t-1} W_{hf} + w_{cf} \odot c_{t-1} + b_f) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \sigma_c(x_t W_{xc} + h_{t-1} W_{hc} + b_c) \\
o_t &= \sigma_o(x_t W_{xo} + h_{t-1} W_{ho} + w_{co} \odot c_t + b_o) \\
h_t &= \sigma_t \odot \sigma_h(c_t)
\end{aligned}
\tag{5.1}
$$

To accelerate training, FCN and LSTM are connected in a residual learning fashion as illustrated in

Figure 5.3. We take the sum of the learned density map over each frame as a base count, and feed the output hidden vector of the last LSTM layer into one fully connected layer to learn the residual between base count and final estimated count. Compared to the direct connection of FCN and LSTM, the residual connection eases the training process and increases counting accuracy.

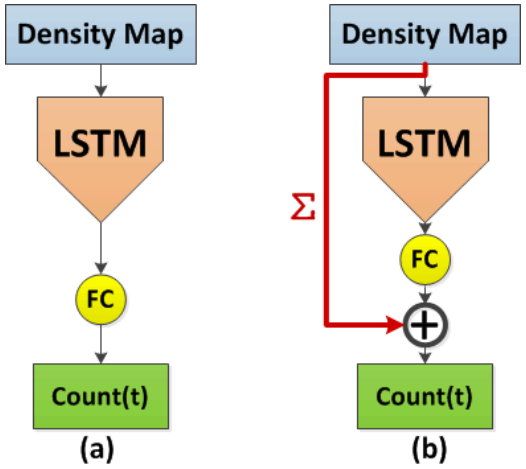


Figure 5.3: Comparison of (a) Direct connection of FCN and LSTM (FCN-dLSTM); (b) Residual connection of FCN and LSTM.

5.3 Spatio-Temporal Multi-Task Learning

The ground truth supervision for FCN-rLSTM includes two types of information: the pixel-level density map and the global vehicle count for each frame. Generation of this supervision depends on how the objects are annotated. If the center of each object is labeled as a dot d, the ground truth vehicle count for frame i is the total number of labeled dots. The ground truth density $F_i^0(p)$ for each pixel p in image i is defined as the sum of 2D Gaussian kernels centered at each dot annotation covering pixel p:

$$F_i^0(p) = \sum_{d \in D_i} N(p; d, \delta) \tag{5.2}$$

where D_i is the set of the dot annotations, d is each annotation dot, and δ of the Gaussian kernel is decided by the perspective map. If each object is annotated by a bounding box $B = (x_1, y_1, x_2, y_2)$, where (x_1, y_1) are the coordinates of the left top point and (x_2, y_2) are the coordinates of the right bottom point, the ground truth vehicle count for frame i is the total number of bounding boxes in frame

i. The center o of each bounding box B is: $o_x = \frac{1}{2}(x_1 + x_2), o_y = \frac{1}{2}(y_1 + y_2)$. Then, the ground truth density $F_i^0(p)$ for each pixel p in image i is defined as:

$$F_i^0(p) = \sum_{o \in O_i} N(p; o, \delta) \tag{5.3}$$

where the parameter O_i is the set of bounding box centers in frame i. The δ of the Gaussian kernel is decided by the width of the bounding box.

The FCN task is to estimate the pixel-level density map, and the LSTM task is to estimate the global vehicle count for each frame. These two tasks are jointly achieved by training the whole FCN-rLSTM network end-to-end. The vehicle density is predicted from the feature map by the last convolution 1×1 layer of the FCN. We adopt Euclidean distance to measure the difference between the estimated density and the ground truth. The loss function for the density map estimation is defined as follows:

$$L_D = \frac{1}{2N} \sum_{i=1}^{N} \sum_{p=1}^{P} \left\| F_i(p; \Theta) - F_i^0(p)) \right\|_2^2 \tag{5.4}$$

where N is the batch size and $F_i(p)$ is the estimated vehicle density for pixel p in image i, and Θ is the parameter of the FCN.

The second task, global count regression, is learned from the LSTM layers including two parts: (i) base count: the integration of the density map over the whole image; and (ii) residual count: learned by the LSTM layers. We sum the two to get the estimated vehicle count:

$$C_i = G(F_i(p; \Theta); \Gamma, \Phi) + \sum_{p=1}^{P} F_i(p; \Theta) \tag{5.5}$$

where $G(F_i; \Gamma, \Phi)$ is the estimated residual count, F_i is the estimated density map for frame i, Γ is the learnable parameters of the LSTM, and Φ is the learnable parameters of the fully connected layers. We hypothesize that it is easier to optimize the residual mapping than to optimize the original mapping.

The loss of the global count estimation is:

$$L_C = \frac{1}{2N}\sum_{i=1}^{N}(C_i - C_i^0)^2 \tag{5.6}$$

where C_i^0 is the ground truth vehicle count of frame i, C_i is the estimated count of frame i. Then overall loss function for the network is defined as:

$$L = L_D + \lambda L_C \tag{5.7}$$

where λ is the weight of the global count loss, and it should be tuned to achieve best accuracy. By simultaneously learning the two related tasks, each task can be better trained with much fewer parameters.

The loss function is optimized via batch-based Adam [59] and backpropagation. Algorithm 2 outlines the FCN-rLSTM training process. As FCN-rLSTM can adapt to different input image resolutions and variation of vehicle scales and perspectives, it is robust to different scenes.

Algorithm 2 FCN-rLSTM Training Algorithm

Input: Images: $\{I_{11}, ..., I_{nm}\}$, wherer n is the number of sequences and m is the number of unrolled frames. Label: Density Maps: $\{F_{11}^0, ..., F_{nm}^0\}$

Output: Parameters of FCN, LSTM, and FC: Θ, Γ, Φ

for i = 1 to max_iteration **do**
  for j = 1 to unroll_number **do**
    F_{ij} = FCN($I_{ij}; \Theta$)
    L_{Dj} = $L_2(F_{ij}, F_{ij}^0)$
    C_{residual} = FC(LSTM($F_{ij}; \Gamma$); Φ)
    C_{ij} = $\sum F_{ij}$ + C_{residual}
    L_{Cj} = $L_2(\sum F_{ij}^0, C_{ij})$
  end for
  L = $\sum L_{Dj}$ + $\lambda \sum L_{Cj}$
  $\Theta, \Gamma, \Phi \leftarrow$ Adam(L, Θ, Γ, Φ)
end for

5.4 Experiments

In this session, we discuss experiments and results: 1. We first evaluate and compare the proposed methods with state-of-the-art methods on the CityCam Dataset[121]; 2. we evaluate the proposed methods on the public dataset TRANCOS [79]; and 3. to verify the robustness and generalization of our model, we evaluate our methods on the public crowd counting dataset UCSD. [17].

Quantitative Evaluations on CityCam

CityCam is a public dataset for large-scale city camera videos, which have low resolution (352×240), low frame rate (1 frame/second), and high occlusion as described in Chapter 2. Both bounding box and vehicle counts are available for $60,000$ frames. The dataset is divided into training and testing sets, with 45,850 and 14,150 frames, respectively, covering multiple cameras and different weather conditions.

Following the same settings as in [121], we evaluate our method on the 14,150 test frames of CityCam, which contains 61 videos from 8 cameras. These videos cover different scenes, congestion states, camera perspectives, weather conditions, and time of the day. The training set contains 45,850 frames with the same resolution, but from different videos. Both training and testing sets are divided into two groups: downtown cameras and parkway cameras. Mean absolute error (MAE) is employed for evaluation.

For FCN-rLSTM, the network architecture and parameters are shown in Figure 5.2. The weight of the vehicle count loss is 0.01. The learning rate is initialized by 0.0001 and adjusted by the first and second order momentum in the training process. To test the efficacy of the proposed Hyper-Atrous combination, combination of FCN and LSTM, and the residual connection, we evaluate different configurations of FCN-rLSTM as shown in Table 5.1. Atrous indicates the atrous convolution; Hyper indicates hypercolumn combination of the feature maps; Direct connect indicates combining FCN with LSTM directly; Residual connect indicates connecting FCN with LSTM in residual fashion. The input images of FCN-rLSTM and its different configurations are resized into 128×128.

Data augmentation. To make the model more robust to various cameras and weather conditions,

Configuration	Atrous	Hyper	Direct connect	Residual connect
FCN-A	✓	X	X	X
FCN-H	X	✓	X	X
FCN-HA	✓	✓	X	X
FCN-dLSTM	✓	✓	✓	X
FCN-rLSTM	✓	✓	X	✓

Table 5.1: Different configurations of FCN-rLSTM

several data augmentation techniques are applied to the training images: 1. horizontal flip, 2. random crop, 3. random brightness, 4. and random contrast.

Baseline approaches. We compare our method with three methods: *Baseline 1: Learning to count* [103]. This work maps each pixel's feature into object density with uniform weight for the whole image. For comparison, we extract dense SIFT features [67] for each pixel using VLFeat [108] and learn the visual words. *Baseline 2: Hydra*[79]. It learns a multi-scale non-linear regression model that uses a pyramid of image patches extracted at multiple scales to perform final density prediction. We train Hydra 3s model on the same training set as FCN-rLSTM. *Baseline 3: FCN* [121]. It develops a deep multi-task model to jointly estimate vehicle density and vehicle count based on FCN. We train FCN on the same training set as FCN-rLSTM.

Experimental Results. We compare the error of the proposed and baseline approaches in Table 5.2. From the results, we see that FCN-rLSTM outperforms all the baseline approaches and all the other configurations. As the testing data cover different congestion states, camera perspectives, weather conditions, and time of the day, these results verify the efficacy and robustness of FCN-rLSTM. To do ablation analysis of the proposed techniques, we also evaluate the performance of different configurations as shown in Table 5.2. With the Hyper-Atrous combination, FCN-HA itself already outperforms all the baseline methods and achieves better accuracy than FCN-A and FCN-H, which verifies the efficacy of the Hyper-Atrous combination. FCN-rLSTM has higher accuracy than FCN-HA and FCN-dLSTM, which verifies the efficacy of the residual connection of FCN and LSTM. When the input images are resized into 256×256, the MAE of FCN-rLSTM is reduced to 1.11 for downtown cameras, and 1.07 for parkway cameras. Figure 5.4 and Figure 5.5 compare the counting results of

Method	Downtown	Parkway
Baseline 1	5.91	5.19
Baseline 2	3.55	3.64
Baseline 3	2.74	2.52
FCN-A	3.07	2.75
FCN-H	2.48	2.30
FCN-HA	2.04	2.04
FCN-dLSTM	1.80	1.82
FCN-rLSTM	**1.53**	**1.63**

Table 5.2: Results comparison on CityCam

FCN-HA and FCN-rLSTM, from which we conclude that FCN-rLSTM estimates better the vehicle count (blue dashed circles) and reduces large counting error induced by oversized vehicles (purple dashed circles). Figure 5.6 and Figure 5.7 shows the density map learned from FCN-rLSTM. Without foreground segmentation, the learned density map can still distinguish background from foreground in both sunny, rainy and cloudy, dense and sparse scenes. Figure 5.8 shows the counting results for six different cameras from downtown and parkway. The camera positions are shown in the map of Figure 5.9. From the counting curves, we see that the proposed FCN-rLSTM accurately counts the vehicles for multiple cameras and long time sequences. Besides the high accuracy achieved by FCN-rLSTM, the convergence of the proposed approach is also improved significantly. As shown in Figure 5.10, the FCN-rLSTM converges much faster than FCN alone networks (FCN-HA). The residual connection of FCN and LSTM also enables faster convergence than the direct connection.

Quantitative Evaluations on TRANCOS

We evaluate FCN-rLSTM on a dataset TRANCOS [79] to verify its efficacy. TRANCOS is a collection of 1244 images of different traffic scenes from surveillance camera videos. It has 46796 annotated vehicles in total and provides a region of interest (ROI) for each image. Images of TRANCOS are from very different scenarios and no perspective maps are provided. The ground truth vehicle density maps are generated by the 2D Gaussian Kernel in the center of each annotated vehicle [44].

MAE of the proposed methods and baselines are compared in Table 5.3. Baseline 2-CCNN is a basic version of the network in [79], and Baseline 2-Hydra augments the performance by learning a

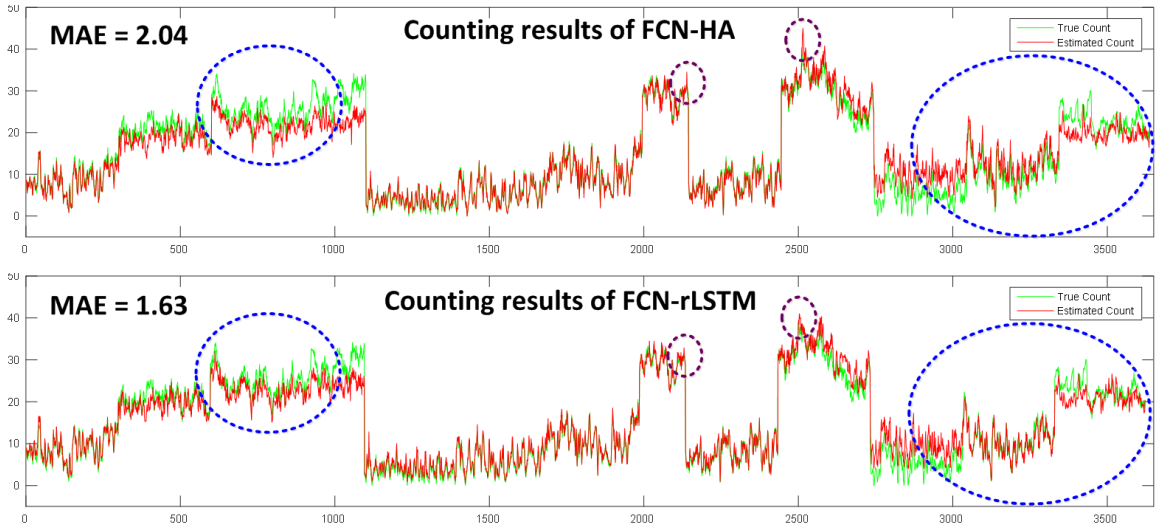


Figure 5.4: Counting results comparison of FCN-HA and FCN-rLSTM on parkway cameras. X axis-frames; Y axis-Counts.

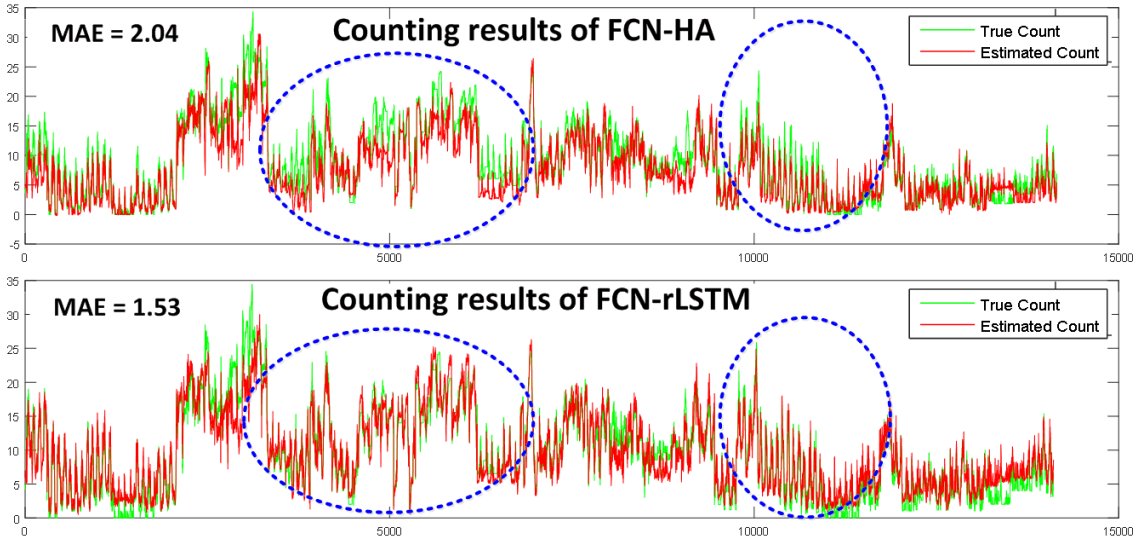


Figure 5.5: Counting results comparison of FCN-HA and FCN-rLSTM on downtown cameras. X axis-frames; Y axis-Counts.

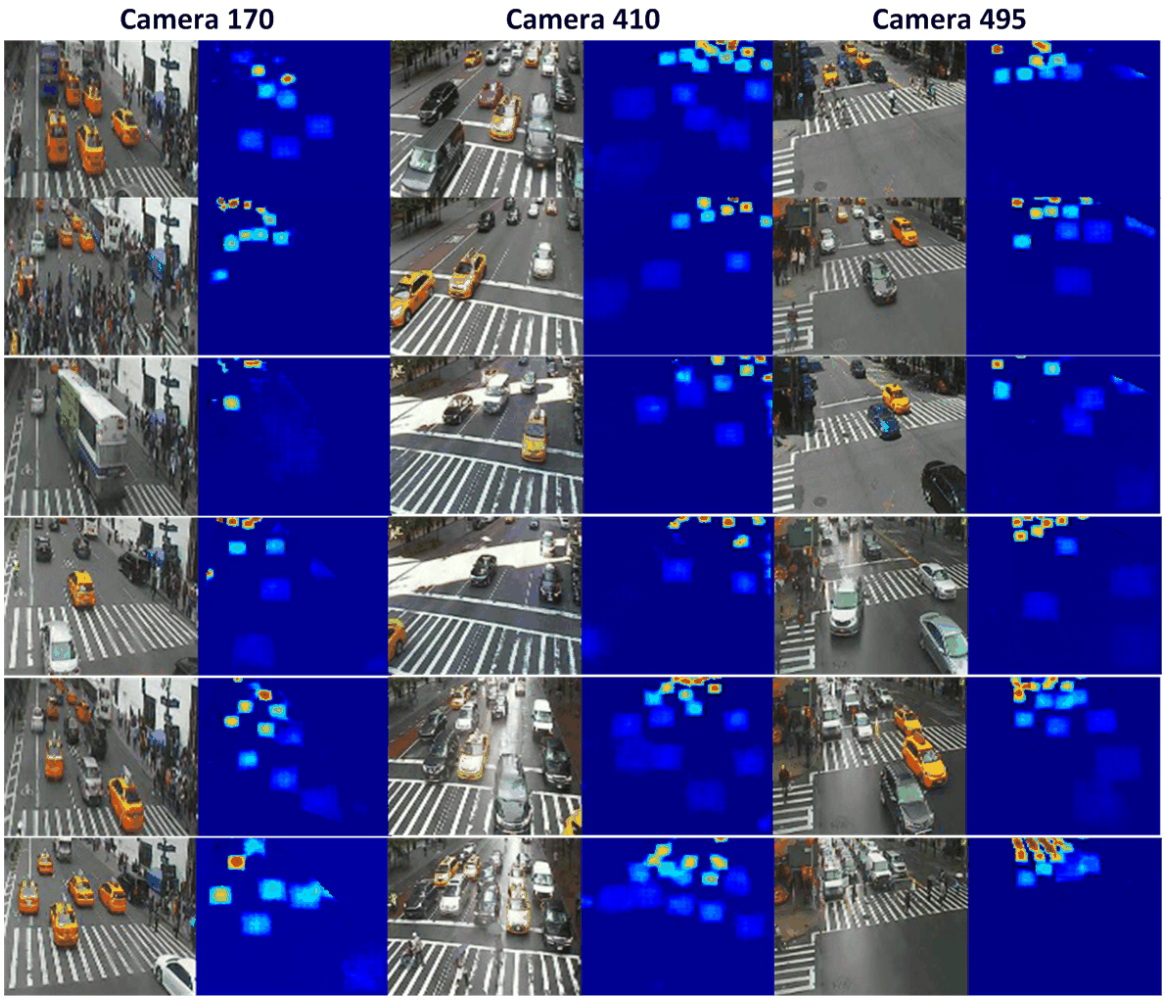


Figure 5.6: Estimated density map for three cameras. Column-wise: The three cameras are from downtown. Row-wise: The first two rows are estimated density maps for cloudy frames; the middle two rows are for sunny frames; and the last two rows are for rainy frames. Better view in color. Some density values may be too small to be clearly seen.

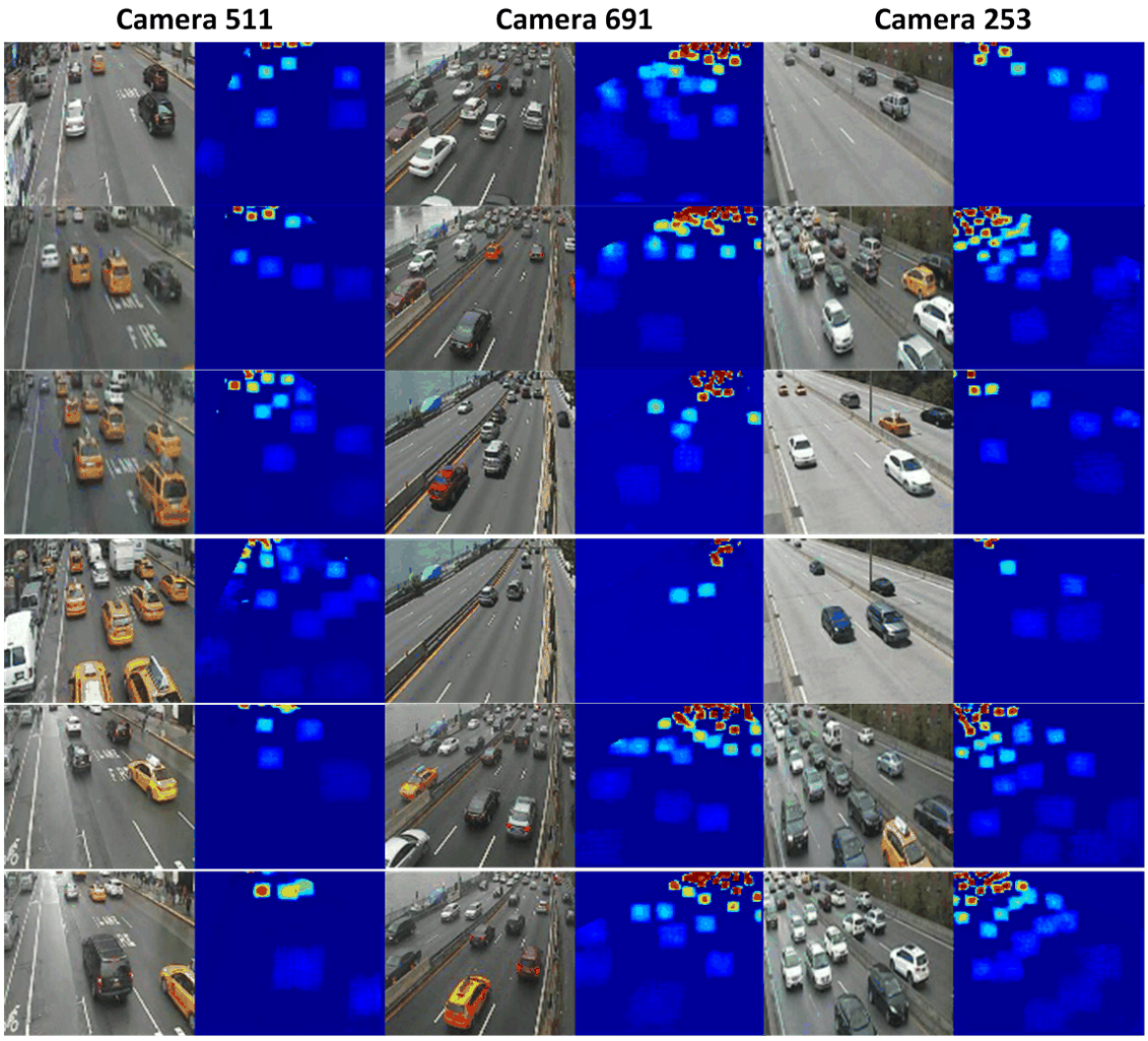


Figure 5.7: Estimated density map for another three cameras. Column-wise: The first camera is from downtown, and the last two cameras are from parkway. Row-wise: The first two rows are estimated density maps for cloudy frames; the middle two rows are for sunny frames; and the last two rows are for rainy frames. Better view in color. Some density values may be too small to be clearly seen.

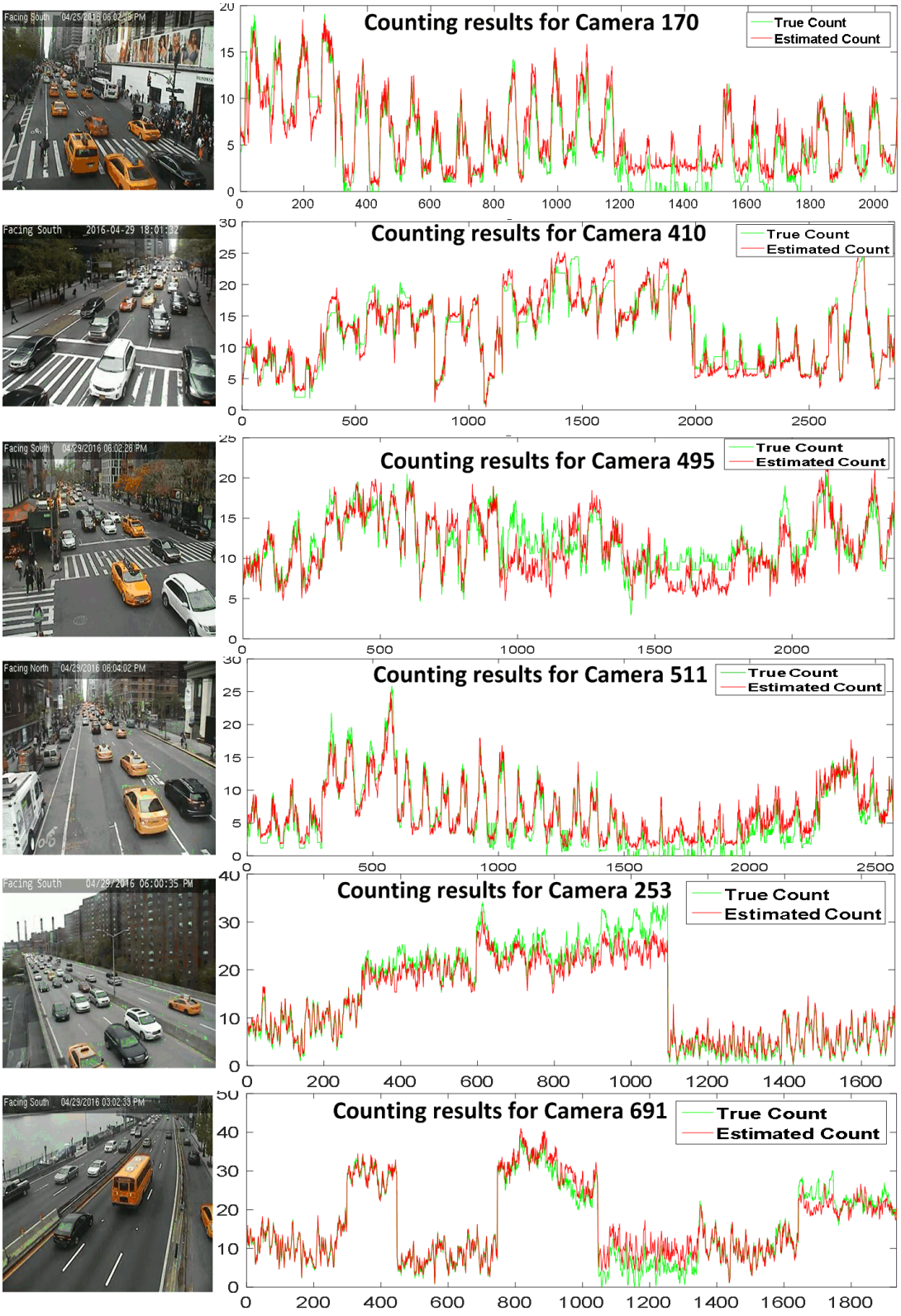


Figure 5.8: Counting results for multiple cameras.

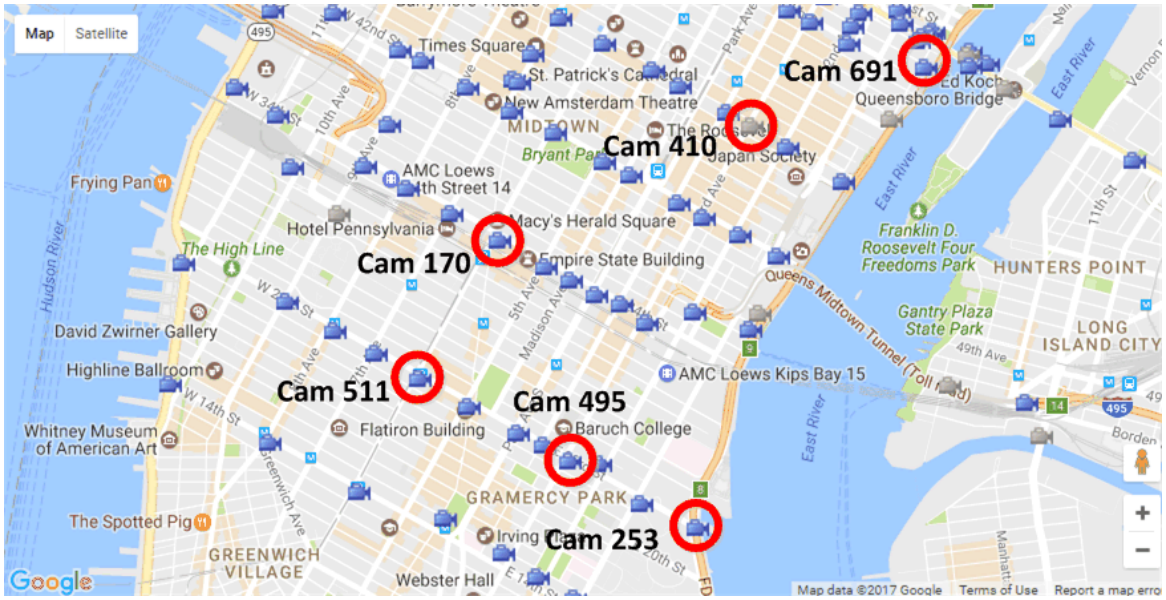


Figure 5.9: Test cameras in the urban area

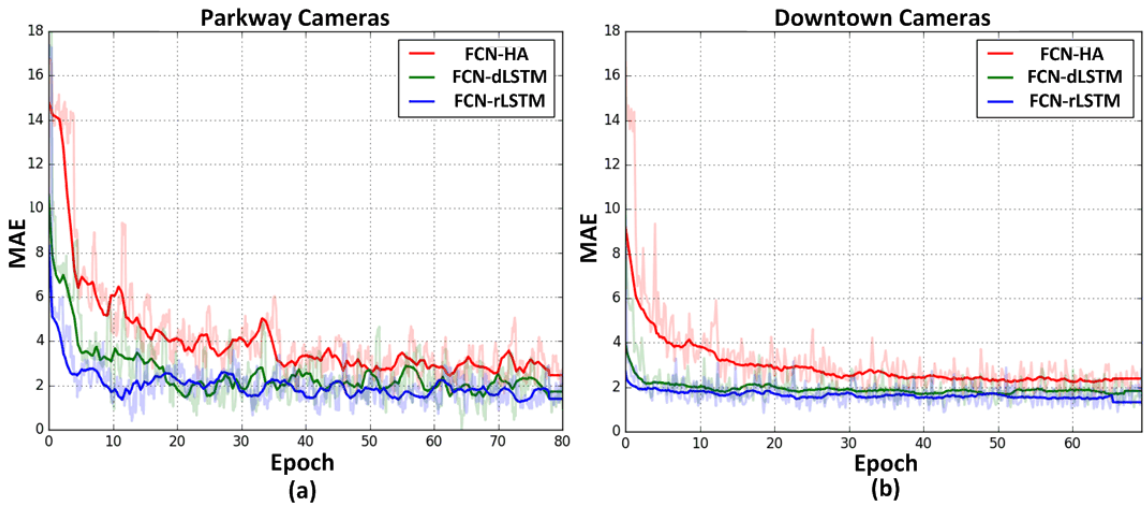


Figure 5.10: Convergence of FCN-HA, FCN-dLSTM, and FCN-rLSTM for: (a) Parkway Cameras (b) Downtown Cameras. Shading shows the MAE over Epochs and dark lines indicate the smoothed trend.

Method	MAE	Method	MAE
Baseline 1	13.76	Baseline 3	5.31
Baseline 2-CCNN	12.49	FCN-HA	**4.21**
Baseline 2-Hydra	10.99	FCN-rLSTM	4.38

Table 5.3: Results comparison on TRANCOS dataset

multiscale regression model with a pyramid of image patches to perform the final density prediction. All the baselines and proposed methods are trained on 823 images and tested on 421 frames following the separation in [79]. From the results, we see that FCN-HA significantly decreases the MAE from 10.99 to 4.21 compared with Baseline 2-Hydra, and decreases the MAE from 5.31 to 4.21 compared with Baseline 3. As the training and testing images of TRANCOS are random samples from different cameras and videos, they lack consistent temporal information. Thus FCN-rLSTM cannot learn temporal patterns from the training data. The performance of FCN-rLSTM is not as good as FCN-HA, but it already outperforms all the baseline methods. When applying our proposed model to other datasets, we can choose the FCN-rLSTM configuration for datasets that have temporal correlation and choose the FCN-HA configuration for datasets that do not have temporal correlation. Figure 5.11 compares the estimated vehicle counts. The estimated counts of the proposed methods are evidently more accurate than those of the baseline methods. FCN-rLSTM and FCN-HA have comparable estimation accuracy of vehicle counts.

Quantitative Evaluations on the UCSD Dataset

To verify the generalization and robustness of our proposed methods in different counting tasks, we also evaluate and compare our methods with baselines on the pedestrian counting dataset UCSD [17]. This dataset contains 2000 frames chosen from one surveillance camera. The frame size is 158×238 and frame rate is 10fps. Average number of people in each frame is around 25. The dataset provides the ROI for each video frame. By following the same setting in [17], we use frames from 601 to 1400 as training data, and the remaining 1200 frames as test data. Table 5.4 shows the results of our methods and existing methods, from which we can see that FCN-rLSTM outperforms all the baseline methods and the FCN-HA configuration. These results show our proposed methods are robust to other type of counting tasks.

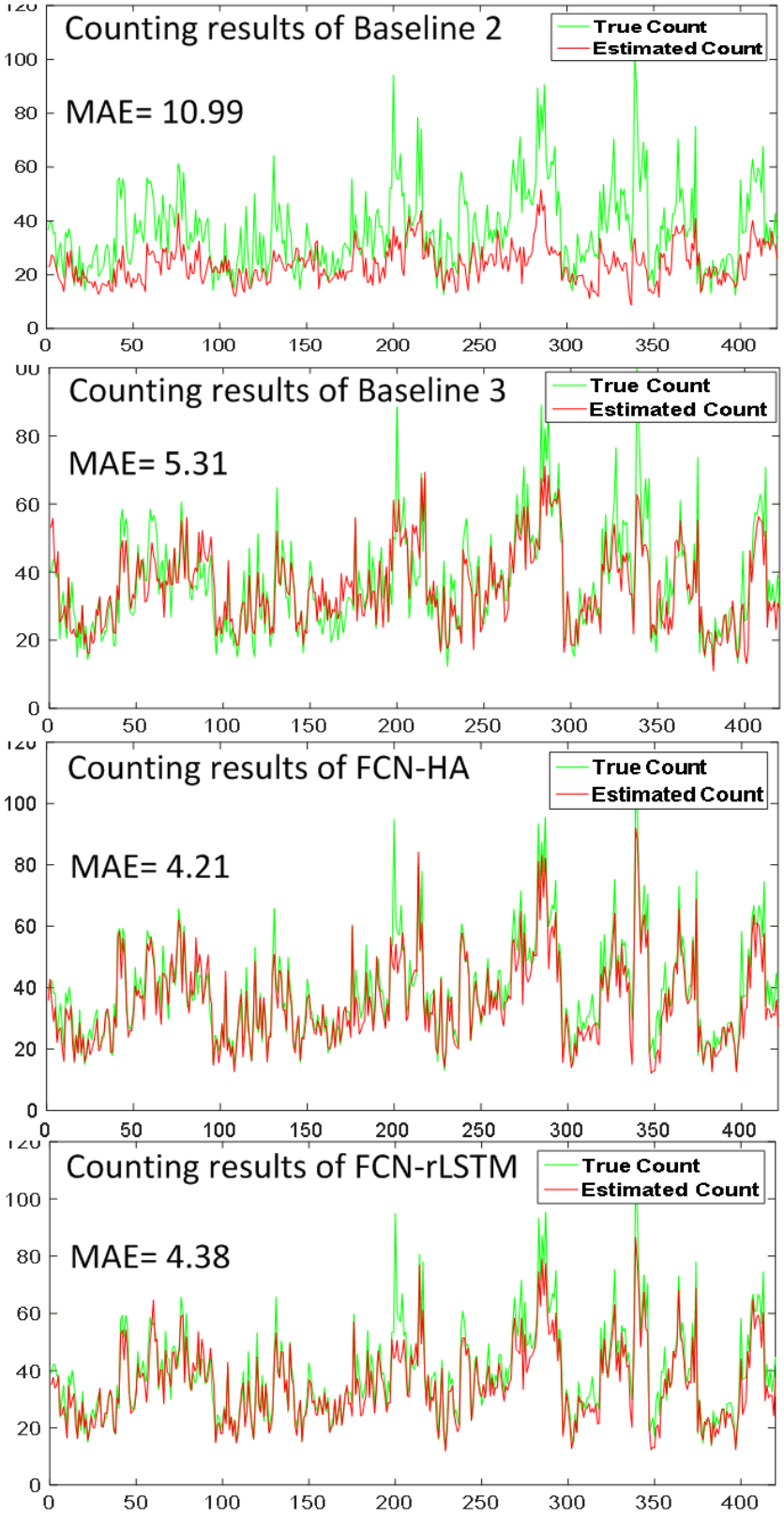


Figure 5.11: Results comparison on TRANCOS dataset.

Method	MAE	MSE
Kernel Ridge Regression [2]	2.16	7.45
Ridge Regression [19]	2.25	7.82
Gaussian Process Regression [17]	2.24	7.97
Cumulative Attribute Regression [20]	2.07	6.86
Cross-scene DNN[117]	1.6	3.31
Baseline 3	1.67	3.41
FCN-HA	1.65	3.37
FCN-rLSTM	**1.54**	**3.02**

Table 5.4: Results comparison on UCSD dataset

5.5 Discussion and Conclusion

Vehicle counting is of great significance for many real world applications, such as traffic management, optimal route planning, and environment quality monitoring. Counting vehicles from webcams is very challenging as videos from webcams have low spatial and temporal resolution, and high occlusion. To overcome these challenges, we develop a novel FCN-rLSTM network architecture to jointly estimate vehicle density and vehicle count by connecting FCN with LSTM in a residual learning fashion. FCN maps pixel-level dense feature into vehicle density to avoid individual vehicle detection and tracking. LSTM layers are added at the end of FCN to learn long-term dependencies and model complex temporal dynamics by incorporating nonlinearities into the network state updates. The residual connection reformulates global count regression as learning residual functions with reference to the sum of densities in each frame. Such design avoids learning unreferenced functions and accelerates the training of the network. Extensive evaluations on different counting tasks and three datasets demonstrate the effectiveness and the robustness of the proposed methods. One limitation of the FCN-rLSTM is that the window size of the unrolled sequential frames is restricted by the available memory capacity. Thus we cannot learn very long term temporal information from the current FCN-rLSTM architecture. This limitation does not significantly affect the counting performance, for small window sizes are capable of learning the smoothness of vehicle count.

Part III

In this part we describe the methodology for vehicle counting for cameras with limited or no annotations.

We also include the thesis conclusion and future work in this part.

- Chapter 6 Multiple Camera Counting with Adversarial Learning
- Chapter 7 Conclusions

Chapter 6

Multiple Camera Counting with Adversarial Learning

6.1 Introduction

6.1.1 Motivation

The OPT-RC presented in Chapter 3, FCN-MT designed in Chapter 4, and FCN-rLSTM developed in Chapter 5 demonstrate increasingly superior performance under the supervised settings for vehicle counting. Nowadays, many cities are instrumented with hundreds of city cameras. These cameras monitor dissimilar scenes with varied backgrounds and very different perspectives. Even for the same camera, weather and illumination change over time. All these factors result in different data domains. On the other hand, it is difficult and expensive to label a large number of training images for all these cameras and data domains. We define cameras with abundant labeled images as source cameras, while cameras with no label information are target cameras, as shown in Fig. 6.1. A crucial issue here is how to adapt the deep model trained on source cameras to data domains from target cameras. As we have hundreds of cameras to deal with, resulting in multiple source and target camera domains, we develop a multi-camera domain adaptation method with adversarial learning (MDAN) that adapts the deep neural networks trained on multiple source cameras to different target cameras.

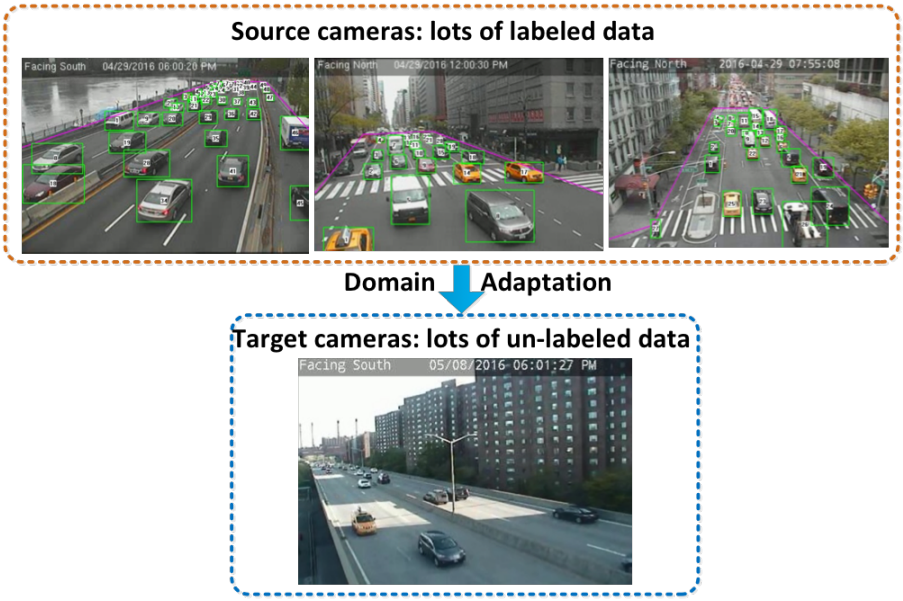


Figure 6.1: Motivation for multi-camera domain adaptation.

While domain adaptation has been actively researched in recent years, most theoretical results and algorithms focus on the single-source-single-target adaptation setting. Naive application of such algorithms to the multiple source domain adaptation problem may lead to suboptimal solutions. To develop our solution, we first propose a new generalization bound for domain adaptation when there are multiple source domains with labeled instances and one target domain with unlabeled instances. Compared with existing bounds [11, 69, 71], the bound we present in this Chapter does not require expert knowledge about the target distribution, nor does it require the optimal combination rule for multisource domains. Our theory also leads to an efficient learning strategy using adversarial neural networks: we show how to interpret it as learning feature representations that are invariant to the multiple domain shifts while still being discriminative for the desired learning task.

To this end, we propose two models, both of which we call multisource domain adversarial network (MDAN): the first model optimizes directly our bound, while the second model is a smoothed approximation of the first one, leading to a more data-efficient and task-adaptive model. The optimization task with both models is a minimax saddle point problem that can be optimized by adversarial training. To demonstrate the effectiveness of these two MDANs, we conduct extensive experiments showing superior adaptation performance on three real-world datasets: Amazon review sentiment analysis [22], digit classification [35], and CityCam vehicle counting, the main problem in this thesis.

6.1.2 Contributions

We summarize here the main results of this chapter. We analyze theoretically the multiple source domain adaptation problem and propose an adversarial learning strategy based on these theoretical results. Specifically, we prove a new generalization bound for domain adaptation when there are multiple source domains with labeled instances and one target domain with unlabeled instances. Our theoretical results build on the seminal theoretical model for domain adaptation introduced by [11], where a divergence measure, known as the $\mathcal{H}$-divergence, was proposed to measure the distance between two distributions based on a given hypothesis space $\mathcal{H}$. Our new result generalizes the bound [11, Thm. 2] to the case when there are multiple source domains. The new bound has an interesting interpretation and reduces to [11, Thm. 2] when there is only one source domain. Technically, we derive our bound by first proposing a generalized $\mathcal{H}$-divergence measure between two sets of distributions from multi-domains. We then prove a PAC bound [104] for the target risk by bounding it from empirical source risks, using tools from concentration inequalities and the VC theory [107]. Compared with existing bounds, the new bound does not require expert knowledge about the target domain distribution [71], nor the optimal combination rule for multiple source domains [11]. Our results also imply that it is not always beneficial to naively incorporate more source domains into training, which we verify to be true in our experiments.

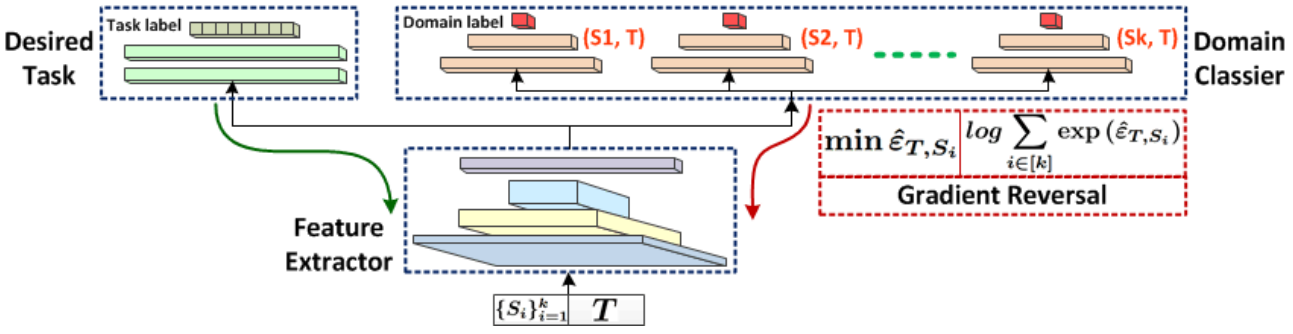


Figure 6.2: MDANs Network architecture. Feature extractor, domain classifier, and task learning are combined in one training process. Hard version: the source that achieves the minimum domain classification error is backpropagated with gradient reversal; Smooth version: all the domain classification risks over k source domains are combined and backpropagated adaptively with gradient reversal.

Interestingly, our bound also leads to an efficient implementation using adversarial neural networks.

This implementation learns both domain invariant and task discriminative feature representations under multiple domains. Specifically, we propose two models (both named MDAN) by using neural networks as rich function approximators to instantiate the generalization bound we derive (Fig. 6.2). After proper transformations, both models can be viewed as computationally efficient approximations of our generalization bound, so that the goal is to optimize the parameters of the networks in order to minimize the bound. The first model optimizes directly our generalization bound, while the second is a smoothed approximation of the first, leading to a more data-efficient and task-adaptive model. The optimization problem for each model is a minimax saddle point problem, which can be interpreted as a zero-sum game with two participants competing against each other to learn invariant features. Both models combine feature extraction, domain classification, and task learning in one training process. MDANs generalize the popular domain adversarial neural network (DANN) [35] and reduce to it when there is only one source domain. We propose to use stochastic optimization with simultaneous updates to optimize the parameters in each iteration. To demonstrate the effectiveness of MDANs as well as the relevance of our theoretical results, we conduct extensive experiments on real-world datasets. We achieve superior adaptation performances on all the tasks, validating the effectiveness of our models.

MDAN learns the features that are discriminative for the main learning task on the source cameras and indiscriminate when shifting between source cameras and target cameras. Fig. 6.2 illustrates the main architecture of MDAN. It includes a deep feature extractor and a deep label predictor for the desired task. Unsupervised multi-camera domain adaptation is achieved by adding a multi-domain classifier connected to the feature extractor via a gradient reversal layer that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds in standard fashion and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the learned feature distributions are similar over different domains (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features. MDAN can be achieved in almost any feed-forward model by augmenting it with a few standard layers and a new gradient reversal layer. The resulting augmented architecture can be trained using standard backpropagation and stochastic gradient descent, and can

thus be easily integrated into the FCN-MT, or the FCN-rLSTM model. As most of the existing domain adaptation work focuses on classification and single source/target domain adaptation for deep neural networks, the proposed MDAN is the first attempt to adapt fully convolutional networks from multiple source domains to different target domains for the regression task.

6.1.3 Related Work for Domain Adaptation

The success of machine learning algorithms has been partially attributed to rich datasets with abundant annotations [46, 61, 85]. Unfortunately, collecting and annotating such large-scale training data is prohibitively expensive and time-consuming. To solve these limitations, different labeled datasets can be combined to build a larger one, or synthetic training data can be generated with explicit yet inexpensive annotations [87]. However, due to the possible shift between training and test samples, learning algorithms based on these cheaper datasets still suffer from high generalization error. Domain adaptation (DA) focuses on such problems by establishing knowledge transfer from a labeled source domain to an unlabeled target domain, and by exploring domain-invariant structures and representations to bridge the gap [80]. Both theoretical results [11, 42, 69, 70, 113] and algorithms [1, 9, 39, 48, 53] for DA have been proposed. Recently, DA algorithms based on deep neural networks have produced breakthrough performance by learning more transferable features [14, 30, 40, 66, 114]. Most theoretical results and algorithms with respect to DA focus on the single-source-single-target adaptation setting [35, 100, 101]. In many application scenarios, the labeled data available may come from multiple domains with different distributions. As a result, naive application of the single-source-single-target DA algorithms may lead to suboptimal solutions. Such problem calls for an efficient technique for multiple source DA. Reference [34, 48, 119] explore multisource DA methods, but they are based on non-deep architectures and their performance leaves much margin to be improved.

From a theoretical point of views, several results have been derived in the form of upper bounds to the generalization target error by learning from source data. A keypoint of the theoretical frameworks is estimating the distribution shift between source and target. Reference [58] proposed the $\mathcal{H}$-divergence to measure the similarity between two domains. It derived a bound on the generalization target error

using the empirical error on the source domain and the $\mathcal{H}$-divergence between the source and the target. This idea has later been extended to multisource domain adaptation [13] and the corresponding bound on the generalization target error has been developed as well. Reference [11] provides a generalization bound for domain adaptation on the target risk that generalizes the standard bound on the source risk. This work formalizes a natural intuition regarding DA: reducing the divergence of two distributions while ensuring a low error on the source domain and justifies many DA algorithms. Based on this work, [70] introduces a new divergence measure: discrepancy distance, whose empirical estimate is based on the Rademacher complexity [60] (rather than the VC-dim). Other theoretical works include [69] that derives the bound on the generalization target error by making use of the robustness properties introduced in [113]. See [27, 70, 72] for further details.

Following these theoretical developments, many DA algorithms have been proposed, such as instance-based methods [99]; feature-based methods [9]; and parameter-based methods [33]. The general approach for domain adaptation starts from algorithms that focus on a linear hypothesis class [12, 26, 38]. The linear assumption can be relaxed and extended to a non-linear setting using the kernel trick, leading to a reweighting scheme that can be efficiently solved via quadratic programming [41, 51]. Recently, due to the availability of rich data and powerful computational resources, non-linear representations and hypothesis classes have been increasingly explored [1, 8, 22, 35, 40]. This line of work focuses on building common and robust feature representations among multiple domains, using either supervised neural networks [40], or unsupervised pretraining using denoising auto-encoders [109, 110].

Recent studies have shown that deep neural networks can learn more transferable features for DA [30, 40, 114]. Reference [14] develops domain separation networks to extract image representations that are partitioned into two subspaces: domain private component and cross-domain shared component. The partitioned representation is utilized to reconstruct images from both domains, improving the DA performance. Reference [66] enables classifier adaptation by learning the residual function with reference to the target classifier. The main-task of this work is limited to the classification problem. Reference [35] proposes a domain-adversarial neural network to learn the domain indiscriminate

but main-task discriminative features. These works generally outperform non-deep learning based methods, but they focus only on the single-source-single-target DA problem, and most of the work is on empirical design with no statistical guarantees. Reference [48] presents a domain transform mixture model for multisource DA, which is based on non-deep architectures and is difficult to scale up.

Adversarial training techniques that aim to build feature representations that are indistinguishable between source and target domains have been proposed in the last few years [1, 35]. Specifically, one of the central ideas is to use neural networks, which are powerful function approximators, to approximate the $\mathcal{H}$-divergence between two domains [10, 11, 58]. The overall algorithm can be viewed as a zero-sum two-player game: one network tries to learn feature representations that can fool the other network, whose goal is to distinguish between representations generated from the source domain from those generated from the target domain. The goal of the algorithm is to find a Nash-equilibrium for the game, or the stationary point of the min-max saddle point problem. Ideally, at such equilibrium state, feature representations from the source domain will share the same distribution as those from the target domain, and, as a result, better generalization on the target domain can be expected by training models using only labeled instances from the source domain.

6.2 Definitions

We first introduce notation used in this Chapter and review a theoretical model for domain adaptation when there is only one source and one target domain [10, 11, 13, 58]. The key idea is the $\mathcal{H}$-divergence to measure the discrepancy between two distributions. Other theoretical models for DA exist [26, 27, 70, 72]; we choose to work with the $\mathcal{H}$-divergence because this distance measure has a particularly natural interpretation and can be well approximated using samples from both domains.

Notation We use *domain* to represent a distribution $\mathcal{D}$ on the input space $\mathcal{X}$ and a labeling function $f : \mathcal{X} \to [0, 1]$. In the setting of one source one target domain adaptation, we use $\langle \mathcal{D}_S, f_S \rangle$ and $\langle \mathcal{D}_T, f_T \rangle$ to denote the source and target domains, respectively. A *hypothesis* is a binary classification function $h : \mathcal{X} \to \{0, 1\}$. The *error* of a hypothesis h w.r.t. a labeling function f under distribution

$\mathcal{D}_S$ is defined as: $\varepsilon_S(h, f) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_S}[|h(\mathbf{x}) - f(\mathbf{x})|]$. When f is also a hypothesis, then this definition reduces to the probability that h disagrees with f under

$$\mathcal{D}_S : \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_S}[|h(\mathbf{x}) - f(\mathbf{x})|] = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_S}[\mathbb{I}(f(\mathbf{x}) \neq h(\mathbf{x}))] = \Pr_{\mathbf{x}\sim\mathcal{D}_S}(f(\mathbf{x}) \neq h(\mathbf{x})),$$

where $\mathbb{I}(\cdot)$ is the indicator function.

We define the *risk* of hypothesis h as the error of h w.r.t. a true labeling function under domain $\mathcal{D}_S$, i.e., $\varepsilon_S(h) := \varepsilon_S(h, f_S)$. Following common notation in computational learning theory, we use $\widehat{\varepsilon}_S(h)$ to denote the empirical risk of h on the source domain. Similarly, we use $\varepsilon_T(h)$ and $\widehat{\varepsilon}_T(h)$ to mean the true risk and the empirical risk on the target domain. The $\mathcal{H}$-divergence is defined as follows:

Definition 6.2.1. *Let $\mathcal{H}$ be a hypothesis class for instance space $\mathcal{X}$, and $\mathcal{A}_\mathcal{H}$ be the collection of subsets of $\mathcal{X}$ that are the support of some hypothesis h in $\mathcal{H}$, i.e., $\mathcal{A}_\mathcal{H} := \{h^{-1}(\{1\}) \mid h \in \mathcal{H}\}$. The distance between two distributions $\mathcal{D}$ and $\mathcal{D}'$ based on $\mathcal{H}$ is:*

$$d_\mathcal{H}(\mathcal{D}, \mathcal{D}') := 2 \sup_{A\in\mathcal{A}_\mathcal{H}} |\Pr_\mathcal{D}(A) - \Pr_{\mathcal{D}'}(A)|$$

When the hypothesis class $\mathcal{H}$ contains all the possible measurable functions over $\mathcal{X}$, $d_\mathcal{H}(\mathcal{D}, \mathcal{D}')$ reduces to the familiar total variation. Given a hypothesis class $\mathcal{H}$, we define its symmetric difference w.r.t. itself as:

$$\mathcal{H}\Delta\mathcal{H} = \{h(\mathbf{x}) \oplus h'(\mathbf{x}) \mid h, h' \in \mathcal{H}\},$$

where $\oplus$ is the xor operation. Let h^* be the optimal hypothesis that achieves the minimum combined risk on both the source and the target domains:

$$h^* := \arg\min_{h\in\mathcal{H}} \varepsilon_S(h) + \varepsilon_T(h)$$

and use λ to denote the combined risk of the optimal hypothesis h^*:

$$\lambda := \varepsilon_S(h^*) + \varepsilon_T(h^*)$$

Reference [10] and [13] proved the following generalization bound on the target risk in terms of the source risk and the discrepancy between the source domain and the target domain:

Theorem 6.2.1 ([13]). *Let $\mathcal{H}$ be a hypothesis space of VC-dimension d and $\mathcal{U}_S$, $\mathcal{U}_T$ be unlabeled samples of size m each, drawn from $\mathcal{D}_S$ and $\mathcal{D}_T$, respectively. Let $\widehat{d}_{\mathcal{H}\Delta\mathcal{H}}$ be the empirical distance on $\mathcal{U}_S$ and $\mathcal{U}_T$; then with probability at least $1-\delta$ over the choice of samples, for each $h \in \mathcal{H}$,*

$$\varepsilon_T(h) \leq \varepsilon_S(h) + \frac{1}{2}\widehat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_S, \mathcal{U}_T) + 4\sqrt{\frac{2d\log(2m) + \log(4/\delta)}{m}} + \lambda \tag{6.1}$$

The generalization bound depends on λ, the optimal combined risk that can be achieved by hypothesis in $\mathcal{H}$. The intuition is that if λ is large, then we cannot hope for a successful domain adaptation. One notable feature of this bound is that the empirical discrepancy distance between two samples $\mathcal{U}_S$ and $\mathcal{U}_T$ can usually be approximated by a classifier to distinguish instances from these two domains.

6.3 A New Generalization Bound for Multiple Source Domain Adaptation

In this section, we first generalize the definition of the discrepancy function $d_{\mathcal{H}}(\cdot,\cdot)$ that is only appropriate when we have a single source domain. We will then use the generalized discrepancy function to derive a generalization bound for multisource domain adaptation. We conclude this section with a discussion and comparison of our bound to existing generalization bounds for multisource domain adaptation [11, 72]. We refer readers to Appendix A for proof details and we focus mainly on the interpretation and implications of the theorems.

Let $\{\mathcal{D}_{S_i}\}_{i=1}^k$ and $\mathcal{D}_T$ be k source domains and the target domain, respectively. We define the discrepancy function $d_{\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ induced by $\mathcal{H}$ to measure the distance between $\mathcal{D}_T$ and a set of domains $\{\mathcal{D}_{S_i}\}_{i=1}^k$ as follows:

Definition 6.3.1.

$$d_{\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) := \max_{i\in[k]} d_{\mathcal{H}}(\mathcal{D}_T; \mathcal{D}_{S_i}) = 2 \max_{i\in[k]} \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{D_T}(A) - \Pr_{D_{S_i}}(A)|$$

Again, let h^* be the optimal hypothesis that achieves the minimum combined risk:

$$h^* := \arg\min_{h\in\mathcal{H}} \left(\varepsilon_T(h) + \max_{i\in[k]} \varepsilon_{S_i}(h) \right)$$

and define

$$\lambda := \varepsilon_T(h^*) + \max_{i\in[k]} \varepsilon_{S_i}(h^*)$$

i.e., the minimum risk that is achieved by h^*. The following lemma holds for $\forall h \in \mathcal{H}$:

Theorem 6.3.1. $\varepsilon_T(h) \leq \max_{i\in[k]} \varepsilon_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) + \lambda$.

Remark. Let us take a closer look at the generalization bound: to make it small, the discrepancy measure between the target domain and the multiple source domains need to be small. Otherwise we cannot hope for successful adaptation by only using labeled instances from the source domains. In this case there will be no hypothesis that performs well on both the source domains and the target domain. It is worth pointing out here that in Theorem 6.3.1, the second term and the third term together introduce a tradeoff (regularization) on the complexity of our hypothesis class $\mathcal{H}$. Namely, if $\mathcal{H}$ is too restricted, then the third term λ can be large while the discrepancy term can be small. On the other hand, if $\mathcal{H}$ is very rich, then we expect the optimal error, λ, to be small, while we expect the discrepancy measure $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ to be large. The first term is a standard source risk term that usually appears in generalization bounds under the PAC-learning framework [104, 107]. Later we shall upper bound this term by its corresponding empirical risk.

The discrepancy distance $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ is usually unknown. We can bound $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ from its empirical estimation using $i.i.d.$ samples from $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$:

Theorem 6.3.2. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions*

of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then for $\epsilon > 0$, we have:

$$\Pr\left(\left|d_{\mathcal{H}}\left(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k\right) - d_{\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k\right)\right| \geq \epsilon\right) \leq 4k\left(\frac{em}{d}\right)^d \exp\left(-m\epsilon^2/8\right)$$

The main idea of the proof is to use VC theory [107] to reduce the infinite hypothesis space to a finite space when acting on finite samples. The theorem then follows from standard union bound and concentration inequalities. Details for these proofs are in Appendix A. Equivalently, the following corollary holds:

Corollary 6.3.1. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, for $0 < \delta < 1$, with probability at least $1 - \delta$ (over the choice of samples), we have:*

$$\left|d_{\mathcal{H}}\left(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k\right) - d_{\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k\right)\right| \leq 2\sqrt{\frac{2}{m}\left(\log\frac{4k}{\delta} + d\log\frac{em}{d}\right)}$$

Note that multiple source domains do not increase the sample size too drastically: it grows only through the square root of a log term where k appears in Corollary. 6.3.1.

Similarly, we do not usually have access to the true error $\max_{i\in[k]} \varepsilon_{S_i}(h)$ on the source domains, but we can often have an estimate ($\max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h)$) from training samples. We now provide a probabilistic guarantee to bound the difference between $\max_{i\in[k]} \varepsilon_{S_i}(h)$ and $\max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h)$ uniformly for all $h \in \mathcal{H}$:

Theorem 6.3.3. *Let $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, for $\epsilon > 0$, we have:*

$$\Pr\left(\sup_{h\in\mathcal{H}}\left|\max_{i\in[k]} \varepsilon_{S_i}(h) - \max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \leq 2k\left(\frac{me}{d}\right)^d \exp(-2m\epsilon^2)$$

Again, Thm. 6.3.3 can be proved by a combination of concentration inequalities and a reduction from infinite space to finite space, along with the subadditivity of the $\max$ function, see Appendix A.

Equivalently, the following corollary holds:

Corollary 6.3.2. *Let $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, for $0 < \delta < 1$, with probability at least $1 - \delta$ (over the choice of samples), we have:*

$$\sup_{h\in\mathcal{H}} \left| \max_{i\in[k]} \varepsilon_{S_i}(h) - \max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h) \right| \leq \sqrt{\frac{1}{2m}\left(\log\frac{2k}{\delta} + d\log\frac{me}{d}\right)}$$

Combining Thm. 6.3.1, Corollaries. 6.3.1, and Corollaries. 6.3.2, and realizing that $VC\text{-dim}(\mathcal{H}\Delta\mathcal{H}) \leq 2VC\text{-dim}(\mathcal{H})$ [3], we have the following theorem:

Theorem 6.3.4. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, for $0 < \delta < 1$, with probability at least $1 - \delta$ (over the choice of samples), we have:*

$$\begin{aligned}\varepsilon_T(h) &\leq \max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h) + \sqrt{\frac{1}{2m}\left(\log\frac{4k}{\delta} + d\log\frac{me}{d}\right)} + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k) + \sqrt{\frac{2}{m}\left(\log\frac{8k}{\delta} + 2d\log\frac{me}{2d}\right)} + \lambda \\ &= \max_{i\in[k]} \widehat{\varepsilon}_{S_i}(h) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k) + O\left(\sqrt{\frac{1}{m}\left(\log\frac{k}{\delta} + d\log\frac{me}{d}\right)}\right) + \lambda \qquad (6.2)\end{aligned}$$

Remark. Each term in Thm. 6.3.4 has a nice interpretation: the first term measures the worst case accuracy of hypothesis h on the k source domains and the second term measures the discrepancy between the target and the k source domains. For domain adaptation to succeed in the multiple sources setting, we have to expect these two terms to be small: we pick our hypothesis h based on its source training errors, and it will generalize only if the discrepancy between sources and target is small. The third term bounds the additional error we may incur because of the possible bias from finite samples. The last term λ is the optimal error we can hope to achieve. Hence, if λ is large, one should not expect the generalization error to be small by training on the source domains. [1] It is also worth pointing out

[1]Of course it is still possible that $\varepsilon_T(h)$ is small while λ is large, but in domain adaptation we do not have access to labeled samples from $\mathcal{D}_T$.

that these four terms appearing in the generalization bound also capture the tradeoff between using a rich hypothesis class $\mathcal{H}$ and a limited one as we discussed above: when using a richer hypothesis class, the first and the last terms in the bound will decrease, while the value of the second term will increase; on the other hand, choosing a limited hypothesis class can decrease the value of the second term, but we may incur additional source training errors and a large λ due to the simplicity of $\mathcal{H}$.

One interesting prediction implied by Thm. 6.3.4 is that the performance on the target domain depends on the worst empirical error among multiple source domains, i.e., it is not always beneficial to naively incorporate more source domains into training. As we will see with experiments, this is indeed the case in many real-world problems. One alternative approach to obtaining an upper bound for multiple source domains is to apply Thm. 6.2.1 repeatedly k times, one for each source and target pair, followed by a union bound to combine them. It is easy to show that this approach can lead to a slightly tighter upper bound with the same asymptotic order in terms of m and k as the one in Thm. 6.3.4. However, as we will see in the next section, the bound in Thm. 6.3.4 provides a nice decoupling of the four terms so that minimizing the bound leads to two practical learning algorithms. As a comparison, the alternative bound cannot be minimized as it requires knowledge of unknown quantities $\lambda_i, \forall i \in [k]$, i.e., the optimal error on each pair of source and target domains.

Comparison with Existing Bounds First, it is easy to see that, upto a multiplicative constant, our bound in (6.2) reduces to the one in Thm. 6.2.1 when there is only one source domain ($k = 1$). Hence Thm. 6.3.4 can be treated as a generalization of Thm. 6.2.1. Reference [13] give a generalization bound for semi-supervised multisource domain adaptation where, besides labeled instances from multiple source domains, the algorithm also has access to a fraction of labeled instances from the target domain. Although in general our bound and the one in [13, Thm. 3] cannot be comparable, it is instructive to see the connections and differences between them: on the one hand, the multiplicative constants of the discrepancy measure and the optimal error in our bound are half of those in the bound in reference [13], leading to a tighter bound; on the other hand, because of the access to labeled instances from the target domain, their bound is expressed relative to the optimal error rate on the target domain, while ours is in terms of the empirical error on the source domain. Finally, thanks to our generalized definition of

$d_{\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$, we do not need to manually specify the optimal combination vector α in [13, Thm. 3], which is unknown in practice. Reference [71] also gives a generalization bound for multisource domain adaptation under the assumption that the target distribution is a mixture of the k sources and the target hypothesis can be represented as a convex combination of the source hypotheses. While the distance measure we use assumes $0-1$ loss function, their generalized discrepancy measure can also be applied to other loss functions [70, 71, 72].

6.4 Multisource Domain Adaptation with Adversarial Learning

In this section we describe a neural network based implementation to minimize the generalization bound we derive in Thm. 6.3.4. The key idea is to reformulate the generalization bound by a minimax saddle point problem and optimize it via adversarial training.

Suppose we are given samples drawn from k source domains $\{\mathcal{D}_{S_i}\}$, each of which contains m instance-label pairs. Additionally, we also have access to unlabeled instances sampled from the target domain $\mathcal{D}_T$. Once we fix our hypothesis class $\mathcal{H}$, the last two terms in the generalization bound (6.2) will be fixed; hence we can only hope to minimize the bound by minimizing the first two terms, i.e., the maximum source training error and the discrepancy between source domains and target domain. The idea is to train a neural network to learn a representation with the following two properties: indistinguishable between the k source domains and the target domain; and informative enough for our desired task to succeed. Note that both requirements are necessary: without the second property, a neural network can learn trivial random noise representations for all the domains, and such representations cannot be distinguished by any discriminator; without the first property, the learned representation does not necessarily generalize to the unseen target domain. Taking these two properties into consideration, we propose the following optimization problem:

$$\min_{h} \quad \max_{i\in[k]} \left(\widehat{\varepsilon}_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}} \left(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k \right) \right) \tag{6.3}$$

One key observation that leads to a practical approximation of $d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k)$ from [10] is that

computing the discrepancy measure is closely related to learning a classifier that is able to distinguish samples from different domains:

$$d_{\mathcal{H}\Delta\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k\right) = \max_{i\in[k]}\left(1 - 2\min_{h\in\mathcal{H}\Delta\mathcal{H}}\left(\frac{1}{2m}\sum_{\mathbf{x}\sim\widehat{\mathcal{D}}_T}\mathbb{I}(h(\mathbf{x})=1) + \frac{1}{2m}\sum_{\mathbf{x}\sim\widehat{\mathcal{D}}_{S_i}}\mathbb{I}(h(\mathbf{x}=0))\right)\right)$$

Let $\widehat{\varepsilon}_{T,S_i}(h)$ be the empirical risk of hypothesis h in the domain discriminating task. Ignoring the constant terms that do not affect the optimization formulation, we can reformulate (6.3) as:

$$\min_h \quad \max_{i\in[k]}\left(\widehat{\varepsilon}_{S_i}(h) - \min_{h'\in\mathcal{H}\Delta\mathcal{H}}\widehat{\varepsilon}_{T,S_i}(h')\right) \tag{6.4}$$

The two terms in (6.4) correspond exactly to the two criteria we just proposed: the first term asks for an informative feature representation for our desired task to succeed, while the second term captures the notion of invariant feature representations between different domains.

Algorithm 3 Multiple Source Domain Adaptation via Adversarial Training

for $t = 1$ to ∞ **do**
  Sample $\{S_i^{(t)}\}_{i=1}^k$ and $T^{(t)}$ from $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ and $\widehat{\mathcal{D}}_T$, each of size m
  for $i = 1$ to k **do**
    Compute $\widehat{\varepsilon}_i^{(t)} := \widehat{\varepsilon}_{S_i^{(t)}}(h) - \min_{h'\in\mathcal{H}\Delta\mathcal{H}}\widehat{\varepsilon}_{T^{(t)},S_i^{(t)}}(h')$
    Compute $w_i^{(t)} := \exp(\widehat{\varepsilon}_i^{(t)})$
  end for
  # Hard version
  Select $i^{(t)} := \arg\max_{i\in[k]}\widehat{\varepsilon}_i^{(t)}$
  Update parameters via backpropagating gradient of $\widehat{\varepsilon}_{i^{(t)}}^{(t)}$
  # Smoothed version
  for $i = 1$ to k **do**
    Normalize $w_i^{(t)} \leftarrow w_i^{(t)} / \sum_{i'\in[k]} w_{i'}^{(t)}$
  end for
  Update parameters via backpropagating gradient of $\sum_{i\in[k]} w_i^{(t)}\widehat{\varepsilon}_i^{(t)}$
end for

Inspired by [35], we use the gradient reversal layer to effectively implement (6.4) by backpropagation. The network architecture is shown in Figure. 6.2. The pseudo-code is listed in Alg. 3 (the hard version). One notable drawback of the hard version in Alg. 3 is that in each iteration the algorithm only updates its parameter based on the gradient from one of the k domains. This is data inefficient and can waste

our computational resources in the forward process. To improve this, we approximate the $\max$ function in (6.4) by the log-sum-exp function, which is a frequently used smooth approximation of the $\max$ function. Define $\widehat{\varepsilon}_i(h) := \widehat{\varepsilon}_{S_i}(h) - \min_{h' \in \mathcal{H}\Delta\mathcal{H}} \widehat{\varepsilon}_{T,S_i}(h')$:

$$\max_{i \in [k]} \widehat{\varepsilon}_i(h) \approx \frac{1}{\gamma} \log \sum_{i \in [k]} \exp(\gamma \widehat{\varepsilon}_i(h))$$

where $\gamma > 0$ is a parameter to control the approximation accuracy. As $\gamma \to \infty$, $\frac{1}{\gamma} \log \sum_{i \in [k]} \exp(\gamma \widehat{\varepsilon}_i(h)) \to \max_{i \in [k]} \widehat{\varepsilon}_i(h)$. Correspondingly, we can formulate a smoothed version of (6.4) as:

$$\min_h \quad \frac{1}{\gamma} \log \sum_{i \in [k]} \exp\left(\gamma \left(\widehat{\varepsilon}_{S_i}(h) - \min_{h' \in \mathcal{H}\Delta\mathcal{H}} \widehat{\varepsilon}_{T,S_i}(h')\right)\right) \tag{6.5}$$

During the optimization, (6.5) naturally provides an adaptive weighting scheme for the k source domains depending on their relative error. Use θ to denote all the model parameters, then:

$$\frac{\partial}{\partial \theta} \frac{1}{\gamma} \log \sum_{i \in [k]} \exp\left(\gamma \left(\widehat{\varepsilon}_{S_i}(h) - \min_{h' \in \mathcal{H}\Delta\mathcal{H}} \widehat{\varepsilon}_{T,S_i}(h')\right)\right) = \sum_{i \in [k]} \frac{\exp \gamma \widehat{\varepsilon}_i(h)}{\sum_{i' \in [k]} \exp \gamma \widehat{\varepsilon}_{i'}(h)} \frac{\partial \widehat{\varepsilon}_i(h)}{\partial \theta} \tag{6.6}$$

The approximation trick not only smooths the objective, but also provides a principled and adaptive way to combine all the gradients from the k source domains.

In words, (6.6) says that the gradient of MDAN is a convex combination of the gradients from all the domains. The larger the error from one domain, the larger the combination weight in the ensemble. Formally, we show that (6.5) also corresponds to minimizing an upper bound of the generalization error on the target domain. But different from Thm. 6.3.4 where the worst case $\mathcal{H}$-divergence is considered, the following theorem characterizes the divergence between the target and multiple source domains using a weighted combination of divergences between the target domain and each source domain. As we will see in Sec. 6.5, the optimization problem (6.5) often leads to better generalizations in practice, which may partly be explained by the ensemble effect of multiple sources implied by the upper bound.

Theorem 6.4.1. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions*

of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, $\forall \alpha \in \mathbb{R}_+^k, \sum_{i\in[k]} \alpha_i = 1$, for $0 < \delta < 1$, with probability at least $1-\delta$ (over the choice of samples), we have:

$$\varepsilon_T(h) \leq \sum_{i\in[k]} \alpha_i \cdot \left(\widehat{\varepsilon}_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \widehat{\mathcal{D}}_{S_i})\right) + O\left(\sqrt{\frac{1}{m}\left(\log\frac{k}{\delta} + d\log\frac{me}{d}\right)}\right) + \lambda_\alpha \quad (6.7)$$

where λ_α is a constant that only depends on $\mathcal{H}$.

By a proper choice of α in the upper bound of Thm. 6.4.1, we obtain the following upper bound that is minimized by the optimization problem (6.5).

Theorem 6.4.2. *Choose $\alpha_i = \exp\left(\widehat{\varepsilon}_i(h)\right) / \sum_{j\in[k]} \exp\left(\widehat{\varepsilon}_j(h)\right)$ with $\widehat{\varepsilon}_i(h) := \widehat{\varepsilon}_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \widehat{\mathcal{D}}_{S_i}\right)$, then we have*

$$\varepsilon_T(h) \leq \log \sum_{i\in[k]} \exp\left(\widehat{\varepsilon}_i(h)\right) + O\left(\sqrt{\frac{1}{m}\left(\log\frac{k}{\delta} + d\log\frac{me}{d}\right)}\right) + \lambda^* \quad (6.8)$$

where λ^* is a constant that only depends on $\mathcal{H}$.

Remark. It is not hard to see that, up to a constant that does not depend on the training errors on multiple domains, the upper bound given by Thm. 6.4.2 is tighter than that of Thm. 6.3.4. We also note that both sample complexity bounds given in Thm. 6.3.4 and Thm. 6.4.2 are optimal in terms of the number of training instances m in each source domain, as it matches the $\Omega(\sqrt{1/m})$ lower bound in the non-realizable binary classification scenario [74, Thm. 3.7]. We summarize this algorithm in the smoothed version of Alg. 3. Note that both algorithms, including the hard version and the smoothed version, reduce to the DANN algorithm [35] when there is only one source domain.

6.5 Experiments

We evaluate both hard and soft MDANs and compare them with state-of-the-art methods on three real-world datasets: the Amazon benchmark dataset [22] for sentiment analysis; a digit classification task that includes 4 datasets: MNIST [62], MNIST-M [35], SVHN [77], and SynthDigits [35]; and our own CityCam dataset descriped in Chapter 2. Details about network architecture and training

parameters of proposed and baseline methods, and detailed dataset description will be introduced in the Appendix B.

6.5.1 Amazon Reviews Classification

Domains within the dataset consist of reviews on a specific kind of product (Books, DVDs, Electronics, and Kitchen appliances). Reviews are encoded as 5000 dimensional feature vectors of unigrams and bigrams, with binary labels indicating sentiment. We conduct 4 experiments: for each of them, we pick one product as target domain and the rest as source domains. Each source domain has 2000 labeled examples, and the target test set has 3000 to 6000 examples. During training, we randomly sample the same number of unlabeled target examples as the source examples in each mini-batch. We implement the Hard-Max and Soft-Max methods according to Alg. 3, and compare them with three baselines: MLPNet, marginalized stacked denoising autoencoders (mSDA) [22], and DANN [35]. DANN cannot be directly applied in the multiple source domains setting. In order to compare MDAN with baselines, we use two protocols. The first one combines all the source domains into a single one and train it using DANN, which we denote as Combine-DANN. The second protocol is to train multiple DANNs separately, where each one corresponds to a source-target pair. Among all the DANNs, we report the one achieving the best performance on the target domain. We denote this experiment as Best-Single-DANN. For fair comparison, all these models are built on the same basic network structure with one input layer (5000 units) and three hidden layers (1000, 500, 100 units).

Train/Test	**MLPNet**	**mSDA**	**Best-Single-DANN**	**Combine-DANN**	**MDANs**	
					Hard-Max	Soft-Max
D+E+K/B	0.7655	0.7698	0.7650	0.7789	0.7845	**0.7863**
B+E+K/D	0.7588	0.7861	0.7732	0.7886	0.7797	**0.8065**
B+D+K/E	0.8460	0.8198	0.8381	0.8491	0.8483	**0.8534**
B+D+E/K	0.8545	0.8426	0.8433	**0.8639**	0.8580	0.8626

Table 6.1: Sentiment classification accuracy.

Results and Analysis We show the accuracy of different methods in Table 6.1. Clearly, Soft-Max significantly outperforms all other methods in most settings. When Kitchen is the target domain,

	MLPNet	mSDA	Best-Single-DANN	Combine-DANN	Hard-Max
	Soft-Max	**Soft-Max**	**Soft-Max**	**Soft-Max**	**Soft-Max**
B	0.550	0.101	0.521	0.013	0.946
D	0.000	0.072	0.000	0.051	0.000
E	0.066	0.000	0.097	0.150	0.022
K	0.306	0.001	0.001	0.239	0.008

Table 6.2: p-values under Wilcoxon test.

cDANN performs slightly better than Soft-Max, and all the methods perform close to each other. Hard-Max is typically slightly worse than Soft-Max. This is mainly due to the low data-efficiency of the Hard-Max model (Section 6.4, Eq. 6.4, Eq. 6.5). We argue that, with more training iterations, the performance of Hard-Max can be further improved. These results verify the effectiveness of MDANs for multisource domain adaptation. To validate the statistical significance of the results, we run a non-parametric Wilcoxon signed-ranked test for each task to compare Soft-Max with the other competitors, as shown in Table 6.2. Each cell corresponds to the p-value of a Wilcoxon test between Soft-Max and one of the other methods, under the null hypothesis that the two paired samples have the same mean. From these p-values, we see Soft-Max is convincingly better than other methods.

6.5.2 Digits Classification

Following the setting in [35], we combine four popular digits datasets (MNIST, MNIST-M, SVHN, and SynthDigits) to build the multisource domain dataset. We take each of MNIST-M, SVHN, and MNIST as target domain in turn, and the rest as sources. Each source domain has $20,000$ labeled images and the target test set has $9,000$ examples.

Baselines We compare Hard-Max and Soft-Max of MDANs with ten baselines:

1. *Best-Single-Source*. A basic network trained on each source domain ($20,000$ images) without domain adaptation and tested on the target domain. Among the three models, we report the one that achieves the best performance on the test set.
2. *Combine-Source*. A basic network trained on a combination of three source domains ($20,000$ images for each) without domain adaptation and tested on the target domain.
3. *Best-Single-DANN*. We train DANNs [35] on each source-target domain pair ($20,000$ images for

each source) and test it on target. Again, we report the best score among the three.

4. *Combine-DANN*. We train a single DANN on a combination of three source domains ($20,000$ images for each).
5. *Best-Single-ADDA*. We train ADDA [101] on each source-target pair ($20,000$ images for each source) and test it on the target domain. We report the best accuracy among the three. ADDA is an unsupervised adversarial adaptation method, which learns a discriminative representation using the labels in the source domain and then a separate encoding that maps the target data to the same space using an asymmetric mapping learned through a domain-adversarial loss.
6. *Combine-ADDA*. We train ADDA on a combination of three source domains ($20,000$ images for each).
7. *Best-Single-MTAE*. We train MTAE [39] on each source-target domain pair ($20,000$ images for each source) and test it on the target domain. We report the best accuracy among the three. MTAE is feature learning algorithm that extends the standard denoising autoencoder framework by substituting corruption with naturally occurring inter-domain variability in the appearance of objects. It learns to transform the original image into analogs in multiple related domains, thereby learns features that are robust to variations across domains. The learned features are then used as inputs to the classifier.
8. *Combine-MTAE*. We train MTAE on a combination of three source domains ($20,000$ images for each).
9. *MDAC*. MDAC [119] is a multiple source domain adaptation algorithm that explores causal models to represent the relationship between the features X and class label Y. It models $P_X|P_Y$ (the process to generate effect X from cause Y) on the target domain as a linear mixture of those on source domains, and estimate all involved parameters by matching the target-domain feature distribution. As MDAC is designed for multiple source domain adaptation, we directly train MDAC on a combination of three source domains.
10. *Target-only*. It is the basic network trained and tested on the target data. It serves as an upper bound on the DA algorithms.

Method	Sv+Mm+Sy/Mt	Mt+Sv+Sy/Mm	Mm+Mt+Sy/Sv
Best-Single-Source	0.964	0.519	0.814
Best-Single-DANN	0.967	0.591	**0.818**
Best-Single-ADDA	0.968	0.657	0.800
Best-Single-MTAE	0.862	0.534	0.703
Combine-Source	0.938	0.561	0.771
MDAC	0.755	0.563	0.604
Combine-DANN	0.925	0.651	0.776
Combine-ADDA	0.927	0.682	0.804
Combine-MTAE	0.821	0.596	0.701
MDAN-Hard-Max	0.976	0.663	0.802
MDAN-Soft-Max	**0.979**	**0.687**	0.816
Target-only	0.987	0.901	0.898

Table 6.3: Accuracy on digit classification. Mt: MNIST; Mm: MNIST-M, Sv: SVHN, Sy: SynthDigits.

All the MDANs and baseline methods are built on the same basic network structure to put them on a equal footing.

Results and Analysis The classification accuracy is shown in Table 6.3. The results show that MDAN outperforms all the baselines in the first two experiments and is comparable with Best-Single-DANN in the third experiment. For the combined sources, MDANs always perform better than the source-only baseline (MDANs vs. Combine-Source). However, a naive combination of different training datasets can sometimes decrease the performance of the baseline methods. This conclusion comes from three observations: First, directly training DANN on a combination of multiple sources leads to worse results than the source-only baseline (Combine-DANN vs. Combine-Source); Second, The performance of Combine-DANN can be worse than the Best-Single-DANN (the first and third experiments); Third, directly training DANN on a combination of multiple sources has always lower accuracy when compared with our approach (Combine-DANN vs. MDANs). We can make similar observations for ADDA and MTAE. Such observations verify that the domain adaptation methods designed for single source lead to suboptimal solutions when applied to multiple sources. They also confirm the superiority of MDAN for multiple source adaptation. Though MDAC is designed for multiple source domain adaptation, it has obviously lower accuracy than MDANs. Furthermore, we observe that adaptation to the SVHN dataset (the third experiment) is hard. In this case, increasing the number of source domains

does not help. We conjecture this is due to the large dissimilarity between the SVHN data to the others. Surprisingly, using a single domain (best-Single DANN) in this case achieves the best result. This indicates that in domain adaptation the quality of data (how close to the target data) is much more important than the quantity (how many source domains). As a conclusion, this experiment further demonstrates the effectiveness of MDANs when there are multiple source domains available, where a naive combination of multiple sources using DANN may hurt generalization.

6.5.3 CityCam Vehicle Counting

As explained in Chapter 2, CityCam is a public dataset for vehicle counting from large-scale city camera videos, which has low resolution (352×240), low frame rate (1 frame/second), and high occlusion. It has $60,000$ frames annotated with vehicle bounding box and count, divided into training and testing sets, with $42,200$ and $17,800$ frames, respectively. Here, we demonstrate the effectiveness of MDANs to count vehicles from an unlabeled target camera by adapting from multiple labeled source cameras: we select 8 cameras that each has more than $2,000$ labeled images for our evaluations. As shown in Fig. 6.3, they are located in different intersections of the city with different scenes. Among these 8 cameras, we randomly pick two cameras and choose each camera as target camera, with the other 7 cameras as sources. We compute the proxy $\mathcal{A}$-distance (PAD) [10] between each source camera and the target camera to approximate the divergence between them. We then rank the source cameras by the PAD from low to high and choose the first k cameras to form the k source domains. Thus the proposed methods and baselines can be evaluated on different numbers of sources (from 2 to 7). We implement the Hard-Max and Soft-Max MDANs according to Alg. 3, based on the basic FCN vehicle counting network of FCN-rLSTM introduced in Chapter 5. We compare our method with two baselines: FCN [120], a basic network without domain adaptation, and DANN [35], implemented on top of the same basic network. We record mean absolute error (MAE) between true count and estimated count.

Results and Analysis The counting error of different methods is compared in Table 6.4. The Hard-Max version achieves lower error than DANN and FCN in most settings for both target cameras. The

S	T	MDANs		DANN	FCN	T	MDANs		DANN	FCN
		Hard-Max	Soft-Max				Hard-Max	Soft-Max		
2	A	1.8101	**1.7140**	1.9490	1.9094	B	2.5059	**2.3438**	2.5218	2.6528
3	A	1.3276	**1.2363**	1.3683	1.5545	B	1.9092	**1.8680**	2.0122	2.4319
4	A	1.3868	**1.1965**	1.5520	1.5499	B	**1.7375**	1.8487	2.1856	2.2351
5	A	1.4021	**1.1942**	1.4156	1.7925	B	1.7758	**1.6016**	1.7228	2.0504
6	A	1.4359	**1.2877**	2.0298	1.7505	B	1.5912	**1.4644**	1.5484	2.2832
7	A	1.4381	**1.2984**	1.5426	1.7646	B	1.5989	**1.5126**	1.5397	1.7324

Table 6.4: Counting error statistics. S is the number of source cameras; T is the target camera id.

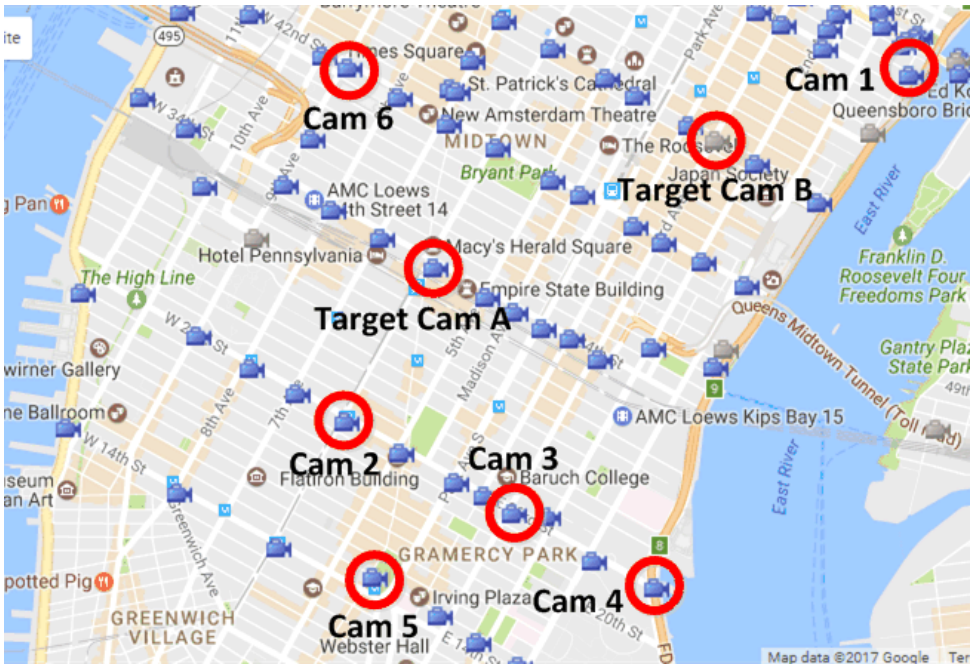


Figure 6.3: Source&target camera map.

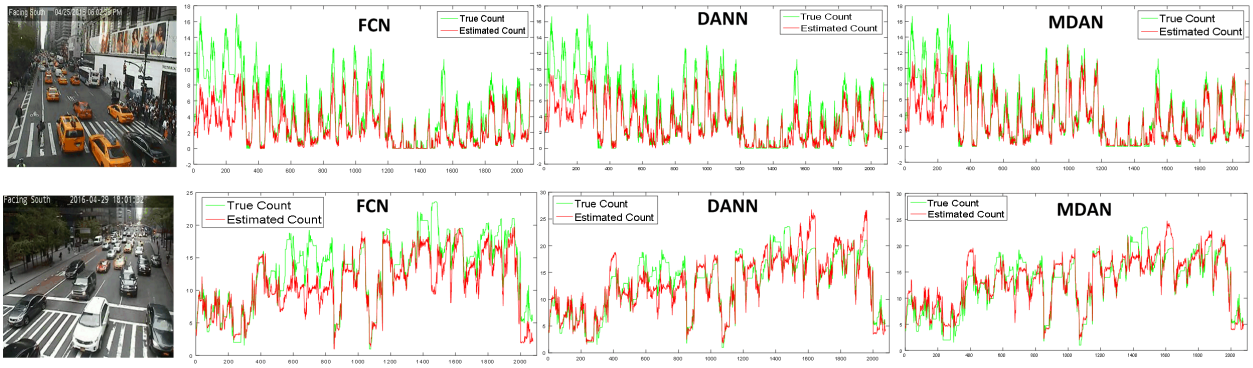
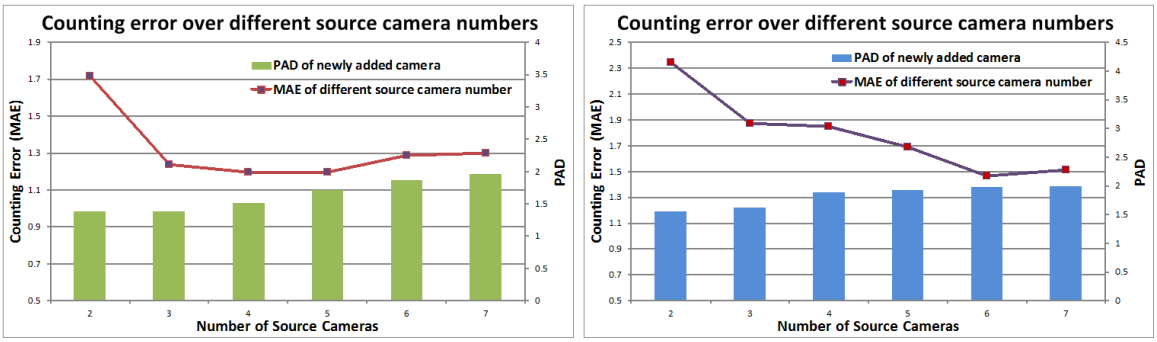


Figure 6.4: Counting results for target camera A (first row) and B (second row). X-frames; Y-Counts.



(a) Camera A (b) Camera B

Figure 6.5: Counting error over different source numbers.

Soft-Max approximation outperforms all the baselines and the Hard-Max in most settings, demonstrating the effectiveness of the smooth and adaptative approximation. The lowest MAE achieved by Soft-Max is 1.1942. Such MAE means that there is only around one vehicle miscount for each frame (the average number of vehicles in one frame is around 20). Fig. 6.4 shows the counting results of Soft-Max for the two target cameras under the 5 source cameras setting. We can see that the proposed method accurately counts the vehicles of each target camera for long time sequences. We can ask if adding more source cameras always helps improvinig the performance on the target camera. To answer this question, we analyze the counting error when we vary the number of source cameras as shown in Fig. 6.5. From the curves, we see the counting error goes down with more source cameras at the beginning, while it goes up when more sources are added at the end. This phenomenon corresponds to the prediction implied by Thm. 6.3.4 (the last remark in Section 6.3): the performance on the target domain depends on the worst empirical error among multiple source domains, i.e., it is not always beneficial to naively incorporate more source domains into training. To illustrate this prediction better, we show the PAD of the newly added camera (when the source number increases by one) in Fig. 6.5. By observing the PAD and the counting error, we see the performance on the target can degrade when the newly added source camera has large divergence from the target camera.

6.6 Conclusion

We derive a new generalization bound for DA under the setting of multiple source domains with labeled instances and one target domain with unlabeled instances. The new bound has interesting

interpretation and reduces to an existing bound when there is only one source domain. Following our theoretical results, we propose MDANs to learn feature representations that are invariant under multiple domain shifts while at the same time being discriminative for the learning task. Both hard and soft versions of MDANs are generalizations of the popular DANN to the case when multiple source domains are available. Empirically, MDANs outperform the state-of-the-art DA methods on three real-world datasets, including a sentiment analysis task, a digit classification task, and a visual vehicle counting task, demonstrating its effectiveness for multisource domain adaptation.

Chapter 7

Conclusions

7.1 Accomplishments

Understanding traffic density is of great importance for many real-world applications, such as urban traffic management and autonomous driving. In this thesis, we propose CityScapeEye, a system to extract vehicle counts from streaming real-time videos captured by hundreds of low resolution web cameras and to construct maps of traffic density and traffic flow in a city environment. This is an extremely challenging problem as videos from web cameras have low spatial and temporal resolution, high occlusion, large perspective, and variable environment conditions. To overcome these challenges, we develop several techniques:

1. a block-level regression model with a rank constraint to map the dense image feature into vehicle densities.
2. a deep multi-task learning framework based on fully convolutional neural networks (FCN-MT) to jointly learn vehicle density and vehicle counts;
3. deep spatio-temporal networks for vehicle counting to incorporate temporal information of the traffic flow;
4. a multi-source domain adaptation mechanism with adversarial learning to adapt the deep counting model to multiple cameras.

To train and validate the developed system, we collected a large-scale webcam traffic dataset CityCam

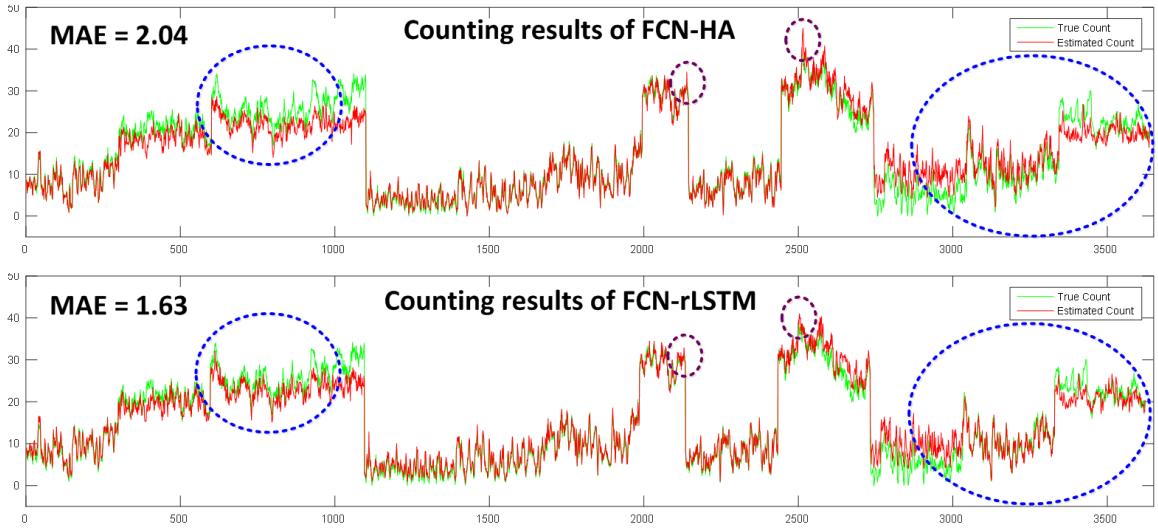


Figure 7.1: Failure case analysis. X axis-frames; Y axis-Counts.

that contains 60 million frames from 212 webcams installed in the key intersections of NYC. Of there, $60,000$ frames have been annotated with rich information, leading to about $900,000$ annotated objects. The proposed methods are extensively evaluated and compared on different counting tasks and datasets to existing techniques, with experimental results demonstrating their effectiveness and robustness.

While we have achieved the above accomplishments, we also notice some failure cases from our methods. As shown in Figure. 7.1, though the FCN-rLSTM achieves better counting accuracy than the baseline methods, there are still some errors between the estimated vehicle counts and the ground truth (blue dotted circle). One reason of these errors is the strong sunshine and shadows in the sunny weather, which confuse the network to distinguish the cars from the shadows. We also observe that there are less training images with very congested or sparse traffic, resulting in the imbalance of the training data and increasing the difficulty to estimate very congested or sparse traffic in the testing data. Another reason for the failure cases is the imperfection of the predefined ROI mask, which covers only half of the parking vehicles on its edge.

7.2 Claims

Cities are becoming increasingly instrumented with a variety of sensors, such as magnetic loop detectors, infra-red sensors, or web cameras. In particular, the increasing availability of video cameras

installed at intersections of urban streets or other roads in the city, allows one to extract real time estimation of traffic density. The citywide web cameras capture traffic video twenty four/seven, continuously, generating large-scale traffic video data with low resolution, low frame rate, high occlusion, and large perspective, precluding most existing techniques for traffic flow analysis. This thesis is the first to develop deep learning based methodologies that extract vehicle counts from streaming real-time videos captured by hundreds of low resolution and low frame rate web cameras. The increasing instrumentation and increasing automation in public sectors leads to increasing amounts of data and the desire to use that data for different applications such as improving automation and, more generally, increasing realtime awareness of information that can be useful for decision-makers. The extracted traffic densities from city wide infrastructure cameras significantly augment autonomous intelligent systems. For example, it enhances the intelligence and autonomy of driverless vehicles by making them interact intelligently with the instrumented infrastructure of the city. It will allow driverless and autonomous vehicles to better plan their routes through the city as they move around the city. The CityScapeEye system that we develop in the thesis can provide each driverless vehicle with maps of traffic flows across the entire city by determining vehicle counts at all traffic cameras available in a city.

The CityScapeEye system outperforms all the state-of-the-art systems for vehicle counting from streaming traffic videos captured by webcams. The system is robust to different weather, light, and environmental conditions, counting vehicles accurately over long time sequences. CityScapeEye can be applied to other counting problems, such as crowd counting, achieving better performance than the best state-of-the-art methods.

We are the first to study the problem of traffic counting for multiple cameras with limited annotations. We formulate this problem as unsupervised multi-source domain adaptation, and we propose a new generalization bound and algorithms to solve it. We develop multi-source domain adaptation networks with adversarial learning to adapt the deep counting model trained from multiple source cameras to the target cameras, which have no annotations, achieving better performance than the best state-of-the-art methods. CityScapeEye is the first system that explores both temporal information (correlation among

sequential frames) and spatial information (correlation among multiple cameras) for traffic counting from large scale webcams.

The collected CityCam dataset is the first and largest annotated webcam traffic dataset to date. Unlike the car data set KITTI[37] and Detrac[112] that focus on vehicle models, CityCam emphasizes real world traffic network analysis in a large metropolis. Benefits provided by CityCam include: 1. It motivates research on vision based traffic flow analysis, posing new challenges for state-of-the-art algorithms. 2. With various street scenes, it can serve as benchmark for transfer learning and domain adaptation. 3. With large amount of labeled data, it provides a training set for various learning based models, especially for deep learning techniques.

7.3 Current Explorations and Future Work

7.3.1 Density Estimation Based Vehicle Detection

Given the superior performance of CityScapeEye on vehicle density estimation, we will further explore it to detect vehicles in traffic videos from webcams. To overcome the limitations of existing methods, we propose a deep multi-task model to jointly learn vehicle density and detect vehicle types. In our preliminary work so far, to detect small and highly occluded vehicles, we take the estimated density map as a prior to the vehicle detection. This method is distinct from existing detection based counting methods that first detect individual vehicles and then count the number of vehicles. In the test stage, our system takes an image as input and outputs vehicle density, class confidence score, and predicted location offsets for each pixel.

Vehicle detection from traffic video is faced with the challenge that high occlusion and small vehicle scales induce high missing rate for vehicle detection. To address this challenge, instead of directly detecting individual vehicles, we regress the vehicle bounding box for each pixel with the prior of the estimated density map. This action encourages pixels with positive density values to have a high confidence score, while pixels with zero density values should have low confidence score. In similar fashion, pixels with large density values are encouraged to have small bounding boxes, while pixels

with small density values to have large bounding boxes.

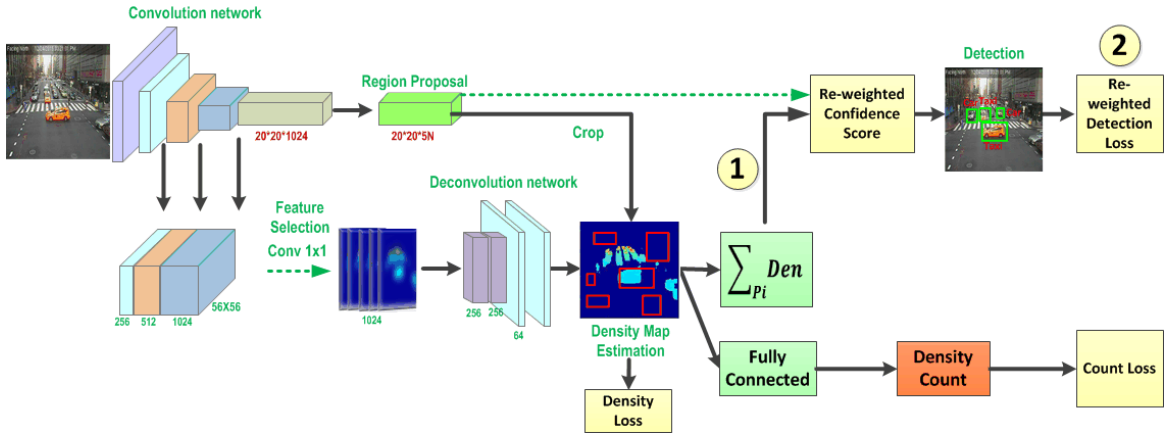


Figure 7.2: Current framework of Den-Det. Jointly learn vehicle density and detect vehicles.

To test this design, we developed the density estimation based vehicle detection network (Den-Det). The overall structure of Den-Det is illustrated in Fig 7.2. The whole network includes two branches: the density estimation branch and the detection branch. The density estimation branch is based on a fully convolutional network; the detection branch is based on Faster RCNN [83] architecture; these two branches share the first several layers for feature extraction. We crop patches from the estimated density map corresponding to each region proposal and calculate the sum of the density for each patch. To take the estimated density as prior for vehicle detection, we reweight the confidence score for each region proposal and the regression loss of the detection, based on the closeness between 1 and the sum of the density. If the sum of the density of that region proposal is close to 1, we encourage this proposal to have a high confidence score and low detection loss; if the sum of the density of that region proposal is far from 1 (either greater or smaller), we encourage this proposal to have a low confidence score and high detection loss. The re-weighted confidence score is:

$$\text{Score}' = N\left(\sum_{p\in P_i} \text{Den}(p); 1, \delta\right) * \text{Score} \tag{7.1}$$

where $Score'$ is the re-weighted score, and P_i is the pixels in the region proposal i. The re-weighted

Figure 7.3: Detection results of Den-Det.

detection loss is:

$$L_{\text{Det}}{}' = \left(1 - N\left(\sum_{p \in P_i} \text{Den}; 1, \delta\right)\right) * L_{\text{Det}} \tag{7.2}$$

where $L_{\text{Det}}{}'$ is the re-weighted detection loss. By jointly learning vehicle density and detect vehicles, both tasks achieve better performance. As a preliminary evaluation, the mean Average Precision (mAP) of Den-Det is 72.8, outperforming YOLO V2 (52.1), SSD (68.6), Tiny Face (65.3), and Faster RCNN [83] (72.3). As a qualitative evaluation, some detection results are shown in Fig. 7.3. From these results we can see that Det-Den is capable of detecting vehicles with various scales, including small cars and big buses. Det-Den is also able to detect vehicles in highly occluded scenarios.

7.3.2 Understanding Camera Perspective

In order to understand different camera perspectives and adapt the counting model to different cameras, we propose an adversarial domain adaptation method that reduces both feature distribution shift and geometric shift. The basic idea is similar to MDAN as described in Chapter 6. In addition to the domain discriminator and density estimation branches, we add a Perspective Discriminator branch to

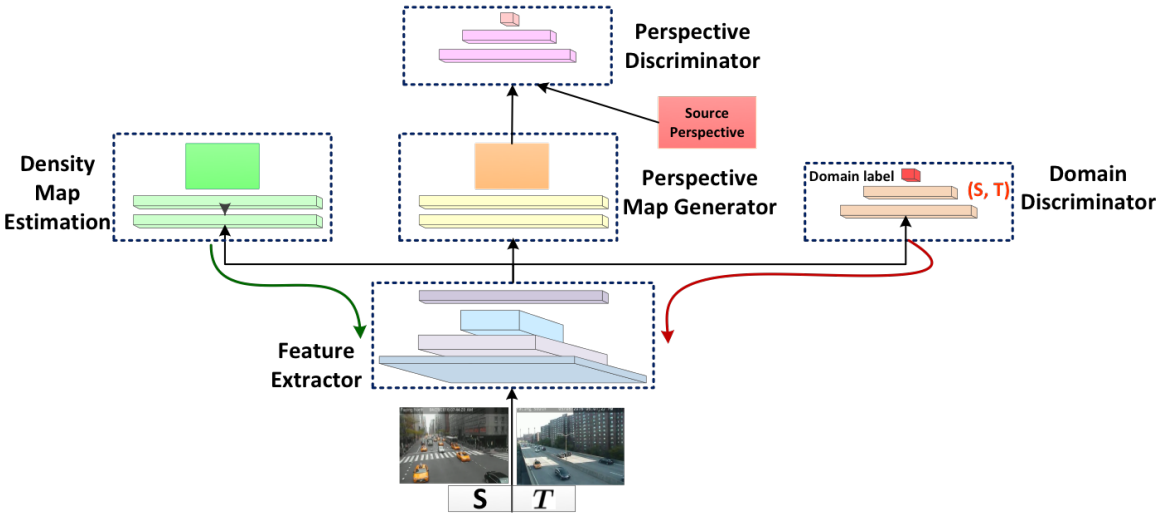


Figure 7.4: Domain adaptation architecture to reduce both feature distribution shift and geometric shift.

reduce the geometric shift between the source camera and the target camera.

The framework is shown in Fig. 7.4. The Perspective Discriminator branch shares the idea of conditional generative adversarial networks (C-GANs) [52, 82, 105]. The input of the whole network is either the image from the source camera or from the target camera. After the Feature Extractor, we add a generator network to generate a perspective map for the input image. Then we input this generated perspective map and the ground truth source camera perspective map into a discriminator network, which should determine if the generated perspective map is a true pair of the ground truth perspective map. With adversarial training of the generator and discriminator networks, we encourage the generator network to generate a source-like perspective map even when the input image is from the target camera. This way, we encourage the Feature Extractor network to reduce the shift between the source perspective and the target perspective.

We have manually generated the ground truth source perspective map (Pmap), as shown in Fig. 7.5. Each pixel of the Pmap is the weight based on the expected depth of the vehicle which generated the pixel, with larger weights given to farther away vehicles. We approximate the ground truth source Pmap by linearly interpolating between the two extremes of the scene. A ground lane is first marked, as shown in Fig. 7.5 (a), and the width of the bottom line and top line are measured by w_1 and w_2. Next, two reference cars are selected, and the length h_1 and h_2 are measured when the center of the cars are on the bottom line and the top line respectively. The pixels on the bottom line are given a

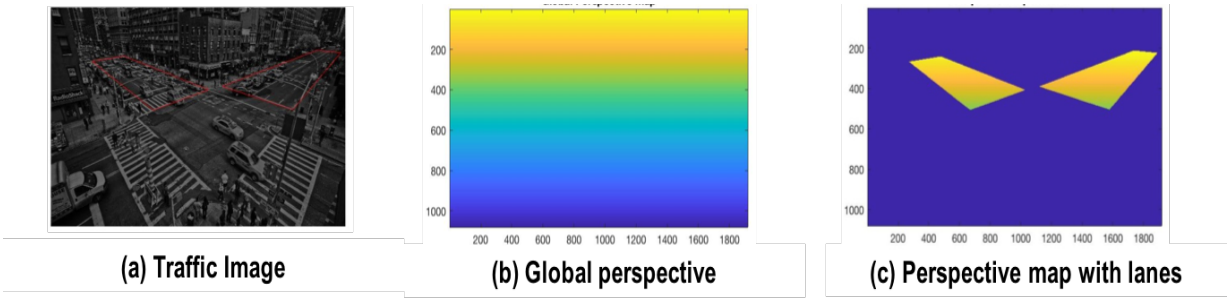


Figure 7.5: Generate perspective map for each camera.

weight of 1, and the pixels on the top line are given the weight of $h_1 w_1 / h_2 w_2$. Finally, the remaining pixel weights are computed by interpolating linearly between the two lines. Fig. 7.5 (c) shows the perspective map of the scene using the above procedure.

7.3.3 Learning Traffic Pattern from Estimated Traffic Density

Finally, an interesting application of traffic density estimation is to learn traffic pattern and detect changes of traffic density when special events occur in a city. Traffic patterns can be analyzed based on the estimated traffic density, vehicle type, and traffic flows. To verify the capability of detecting traffic pattern changes by our method, we estimate traffic densities for two cameras and multiple days. From the results we find that traffic density on July 4th at 18h is different from that on other normal days, as illustrated in Figure 7.6. For the camera in downtown (3Ave@49st), the traffic density on July 4th is on average lower than that of other days, and the traffic can be periodically very sparse. This can be explained by the fact that several streets around 3Ave@49st are closed after 3 pm due to the fireworks show on Independence Day, resulting in less traffic around it. For the camera on parkway (FDR Dr @ 79st), the average traffic is less than on Friday (4-29), but larger than on a normal Monday (5-2). The detected increase of the traffic density on July 4th corresponds to the fact that FDR under 68 St is closed, resulting in larger traffic congestion on FDR above 68 St. We can also compute the Isomap of the estimated density maps and cluster them into different traffic patterns. All these observations verify that CityScapeEye can help to detect traffic density changes when special events occur.

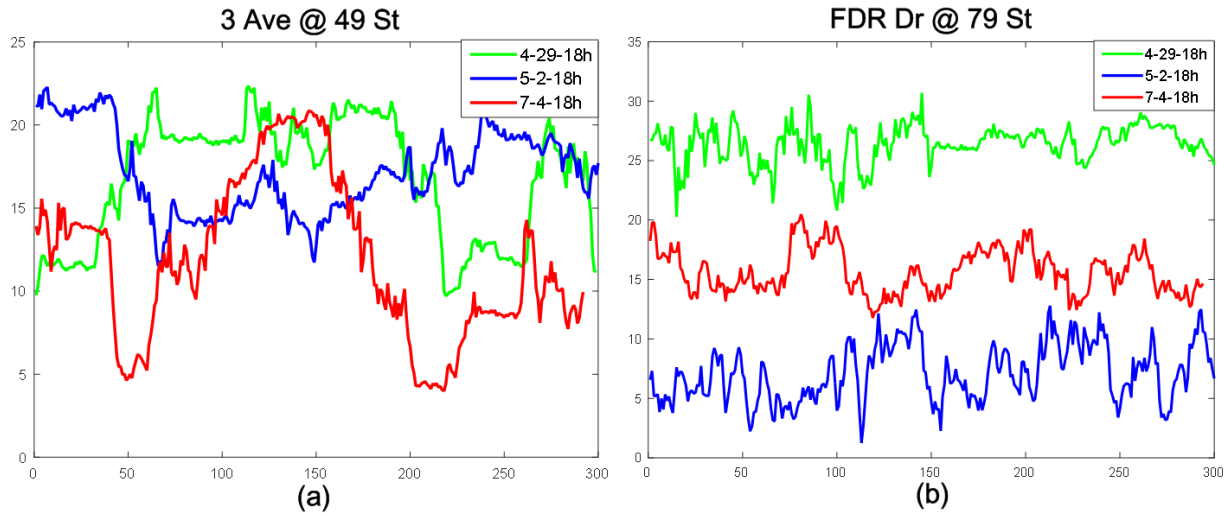


Figure 7.6: Independence Day traffic density detection

Appendix A

Theoretical Proofs

A.1 Technical Tools

Definition A.1.1 (Growth function)**.** *The* growth function $\Pi_{\mathcal{H}} : \mathbb{N} \to \mathbb{N}$ *for a hypothesis class* $\mathcal{H}$ *is defined by:*

$$\forall m \in \mathbb{N}, \quad \Pi_{\mathcal{H}}(m) = \max_{X_m \subseteq \mathcal{X}} |\{(h(x_1), \ldots, h(x_m)) \mid h \in \mathcal{H}\}|$$

where $X_m = \{x_1, \ldots, x_m\}$ *is a subset of* $\mathcal{X}$ *with size* m*.*

Roughly, the growth function $\Pi_{\mathcal{H}}(m)$ computes the maximum number of distinct ways in which m points can be classified using hypothesis in $\mathcal{H}$. A closely related concept is the *Vapnik–Chervonenkis dimension* (VC dimension) [107]:

Definition A.1.2 (VC dimension)**.** *The VC-dimension of a hypothesis class* $\mathcal{H}$ *is defined as:*

$$VC\text{-}dim(\mathcal{H}) = \max\{m : \Pi_{\mathcal{H}}(m) = 2^m\}$$

A well-known result relating VC-dim$(\mathcal{H})$ and the growth function $\Pi_{\mathcal{H}}(m)$ is the Sauer's lemma:

Lemma A.1.1 (Sauer's lemma)**.** *Let* $\mathcal{H}$ *be a hypothesis class with* $VC\text{-}dim(\mathcal{H}) = d$*. Then, for* $m \geq d$*,*

the following inequality holds:

$$\Pi_{\mathcal{H}}(m) \leq \sum_{i=0}^{d} \binom{m}{i} \leq \left(\frac{em}{d}\right)^{d}$$

The following concentration inequality will be used:

Theorem A.1.1 (Hoeffding's inequality)**.** *Let $X_1, \ldots, X_n$ be independent random variables where each X_i is bounded by the interval $[a_i, b_i]$. Define the empirical mean of these random variables by $\bar{X} := \frac{1}{n}\sum_{i=1}^{n} X_i$, then $\forall \varepsilon > 0$:*

$$\Pr\left(\left|\bar{X} - \mathbb{E}[\bar{X}]\right| \geq \varepsilon\right) \leq 2\exp\left(-\frac{2n^2\varepsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right)$$

The VC inequality allows us to give a uniform bound on the binary classification error of a hypothesis class $\mathcal{H}$ using growth function:

Theorem A.1.2 (VC inequality)**.** *Let $\Pi_{\mathcal{H}}$ be the growth function of hypothesis class $\mathcal{H}$. For $h \in \mathcal{H}$, let $\varepsilon(h)$ be the true risk of h w.r.t. the generation distribution $\mathcal{D}$ and the true labeling function h^*. Similarly, let $\hat{\varepsilon}_n(h)$ be the empirical risk on a random $i.i.d.$ sample containing n instances from $\mathcal{D}$, then, for $\forall \varepsilon > 0$, the following inequality hold:*

$$\Pr\left(\sup_{h \in \mathcal{H}} |\varepsilon(h) - \hat{\varepsilon}_n(h)| \geq \varepsilon\right) \leq 8\Pi_{\mathcal{H}}(n)\exp\left(-n\varepsilon^2/32\right)$$

Although the above theorem is stated for binary classification error, we can extend it to any bounded error. This will only change the multiplicative constant of the bound.

A.2 Proofs

For all the proofs presented here, the following lemma shown by [13] will be repeatedly used:

Lemma A.2.1 ([13])**.** $\forall h, h' \in \mathcal{H}, \quad |\varepsilon_S(h, h') - \varepsilon_T(h, h')| \leq \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_S, \mathcal{D}_T)$.

A.2.1 Proof of Thm. 6.3.1

One technical lemma we will frequently use to prove Thm. 6.3.1 is the triangular inequality w.r.t. $\varepsilon_{\mathcal{D}}(h), \forall h \in \mathcal{H}$:

Lemma A.2.2. *For any hypothesis class $\mathcal{H}$ and any distribution $\mathcal{D}$ on $\mathcal{X}$, the following triangular inequality holds:*

$$\forall h, h', f \in \mathcal{H}, \quad \varepsilon_{\mathcal{D}}(h, h') \leq \varepsilon_{\mathcal{D}}(h, f) + \varepsilon_{\mathcal{D}}(f, h')$$

Proof.

$$\varepsilon_{\mathcal{D}}(h, h') = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[|h(\mathbf{x}) - h'(\mathbf{x})|] \leq \mathbb{E}_{\mathbf{x}\sim\mathcal{D}}[|h(\mathbf{x}) - f(\mathbf{x})| + |f(\mathbf{x}) - f(\mathbf{x})|] = \varepsilon_{\mathcal{D}}(h, f) + \varepsilon_{\mathcal{D}}(f, h')$$

■

Now we are ready to prove Thm. 6.3.1:

Theorem 6.3.1. $\varepsilon_T(h) \leq \max_{i\in[k]} \varepsilon_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) + \lambda$.

Proof. $\forall h \in \mathcal{H}$, define $i_h := \arg\max_{i\in[k]} \varepsilon_{S_i}(h, h^*)$:

$$\begin{aligned}
\varepsilon_T(h) &\leq \varepsilon_T(h^*) + \varepsilon_T(h, h^*) \\
&= \varepsilon_T(h^*) + \varepsilon_T(h, h^*) - \max_{i\in[k]} \varepsilon_{S_i}(h, h^*) + \max_{i\in[k]} \varepsilon_{S_i}(h, h^*) \\
&\leq \varepsilon_T(h^*) + |\varepsilon_T(h, h^*) - \varepsilon_{S_{i_h}}(h, h^*)| + \varepsilon_{S_{i_h}}(h, h^*) \\
&\leq \varepsilon_T(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T, \mathcal{D}_{S_{i_h}}) + \varepsilon_{S_{i_h}}(h, h^*) \\
&\leq \varepsilon_T(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) + \varepsilon_{S_{i_h}}(h, h^*) \\
&\leq \varepsilon_T(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) + \varepsilon_{S_{i_h}}(h) + \varepsilon_{S_{i_h}}(h^*) \\
&\leq \varepsilon_T(h^*) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) + \max_{i\in[k]} \varepsilon_{S_i}(h) + \max_{i\in[k]} \varepsilon_{S_i}(h^*) \\
&= \max_{i\in[k]} \varepsilon_{S_i}(h) + \lambda + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)
\end{aligned}$$

The first and the fifth inequalities are due to the triangle inequality, and the third inequality is based on Lemma A.2.1. The second holds due to the property of $|\cdot|$ and the others follow by the definition of $\mathcal{H}$-divergence. ■

A.2.2 Proof of Thm. 6.3.2

Theorem 6.3.2. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then for $\epsilon > 0$, we have:*

$$\Pr\left(\left|d_{\mathcal{H}}\left(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k\right) - d_{\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k\right)\right| \geq \epsilon\right) \leq 4k\left(\frac{em}{d}\right)^d \exp\left(-m\epsilon^2/8\right)$$

Proof.

$$\begin{aligned}
&\Pr\left(\left|d_{\mathcal{H}}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) - d_{\mathcal{H}}(\hat{\mathcal{D}}_T; \{\hat{\mathcal{D}}_{S_i}\}_{i=1}^k)\right| \geq \epsilon\right) \\
&= \Pr\left(\left|\max_{i\in[k]} \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - \max_{i\in[k]} \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)|\right| \geq \frac{\epsilon}{2}\right) \\
&\leq \Pr\left(\max_{i\in[k]} \sup_{A\in\mathcal{A}_{\mathcal{H}}} \left|\,|\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)|\right| \geq \frac{\epsilon}{2}\right) \\
&= \Pr\left(\exists i \in [k], \exists A \in \mathcal{A}_{\mathcal{H}} : \left|\,|\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)|\right| \geq \frac{\epsilon}{2}\right) \\
&\leq \sum_{i=1}^k \Pr\left(\exists A \in \mathcal{A}_{\mathcal{H}} : \left|\,|\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)|\right| \geq \frac{\epsilon}{2}\right) \\
&\leq \sum_{i=1}^k \Pr\left(\exists A \in \mathcal{A}_{\mathcal{H}} : |\Pr_{\mathcal{D}_T}(A) - \Pr_{\hat{\mathcal{D}}_T}(A)| + |\Pr_{\mathcal{D}_{S_i}}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)| \geq \frac{\epsilon}{2}\right) \\
&\leq 2k \Pr\left(\exists A \in \mathcal{A}_{\mathcal{H}} : |\Pr_{\mathcal{D}_T}(A) - \Pr_{\hat{\mathcal{D}}_T}(A)| \geq \frac{\epsilon}{4}\right) \\
&\leq 2k \cdot \Pi_{\mathcal{A}_{\mathcal{H}}}(m) \Pr\left(|\Pr_{\mathcal{D}_T}(A) - \Pr_{\hat{\mathcal{D}}_T}(A)| \geq \frac{\epsilon}{4}\right) \\
&\leq 2k \cdot \Pi_{\mathcal{A}_{\mathcal{H}}}(m) \cdot 2\exp(-2m\epsilon^2/16) \\
&\leq 4k\left(\frac{em}{d}\right)^d \exp(-m\epsilon^2/8)
\end{aligned}$$

The first inequality holds due to the sub-additivity of the $\max$ function, and the second inequality is due to the union bound. The third inequality holds because of the triangle inequality, and we use the averaging argument to establish the fourth inequality. The fifth inequality is an application of the VC-inequality, and the sixth is by the Hoeffding's inequality. Finally, we use the Sauer's lemma to prove the last inequality. ■

A.2.3 Proof of Thm. 6.3.3

We now show the detailed proof of Thm. 6.3.3.

Proof.

$$\begin{aligned}
\Pr\left(\sup_{h\in\mathcal{H}}\left|\max_{i\in[k]}\varepsilon_{S_i}(h)-\max_{i\in[k]}\hat{\varepsilon}_{S_i}(h)\right|\geq\epsilon\right) &\leq \Pr\left(\sup_{h\in\mathcal{H}}\max_{i\in[k]}|\varepsilon_{S_i}(h)-\hat{\varepsilon}_{S_i}(h)|\geq\epsilon\right)\\
&= \Pr\left(\max_{i\in[k]}\sup_{h\in\mathcal{H}}|\varepsilon_{S_i}(h)-\hat{\varepsilon}_{S_i}(h)|\geq\epsilon\right)\\
&\leq \sum_{i=1}^{k}\Pr\left(\sup_{h\in\mathcal{H}}|\varepsilon_{S_i}(h)-\hat{\varepsilon}_{S_i}(h)|\geq\epsilon\right)\\
&\leq k\cdot\Pi_{\mathcal{H}}(m)\Pr\left(|\varepsilon_{S_i}(h)-\hat{\varepsilon}_{S_i}(h)|\geq\epsilon\right)\\
&\leq k\cdot\Pi_{\mathcal{H}}(m)\cdot 2\exp(-2m\epsilon^2)\\
&\leq 2k\left(\frac{me}{d}\right)^d\exp(-2m\epsilon^2)
\end{aligned}$$

Again, the first inequality is due to the subadditivity of the $\max$ function, and the second inequality holds due to the union bound. We apply the VC-inequality to bound the third inequality, and Hoeffding's inequality to bound the fourth. Again, the last one is due to Sauer's lemma. ■

A.2.4 Derivation of the Discrepancy Distance as Classification Error

We show that the $\mathcal{H}$-divergence is equivalent to a binary classification accuracy in discriminating instances from different domains. Suppose $\mathcal{A}_{\mathcal{H}}$ is symmetric, i.e., $A\in\mathcal{A}_{\mathcal{H}}\Leftrightarrow\mathcal{X}\backslash A\in\mathcal{A}_{\mathcal{H}}$, and we

have samples $\{S_i\}_{i=1}^k$ and T from $\{\mathcal{D}_{S_i}\}_{i=1}^k$ and $\mathcal{D}_T$ respectively, each of which is of size m, then:

$$\begin{aligned}
d_{\mathcal{H}\Delta\mathcal{H}}(\hat{\mathcal{D}}_T; \{\hat{\mathcal{D}}_{S_i}\}_{i=1}^k) &= \max_{i\in[k]} \sup_{A\in\mathcal{A}_{\mathcal{H}\Delta\mathcal{H}}} |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)| \\
&= \max_{i\in[k]} \sup_{h\in\mathcal{H}\Delta\mathcal{H}} |\Pr_{\mathbf{x}\sim\hat{\mathcal{D}}_T}(h(\mathbf{x})=1) - \Pr_{\mathbf{x}\sim\hat{\mathcal{D}}_{S_i}}(h(\mathbf{x}=1))| \\
&= \max_{i\in[k]} \sup_{h\in\mathcal{H}\Delta\mathcal{H}} 1 - \left(\Pr_{\mathbf{x}\sim\hat{\mathcal{D}}_T}(h(\mathbf{x})=1) + \Pr_{\mathbf{x}\sim\hat{\mathcal{D}}_{S_i}}(h(\mathbf{x}=0))\right) \\
&= \max_{i\in[k]} \left(1 - 2\min_{h\in\mathcal{H}\Delta\mathcal{H}} \left(\frac{1}{2m}\sum_{\mathbf{x}\sim\hat{\mathcal{D}}_T} \mathbb{I}(h(\mathbf{x})=1) + \frac{1}{2m}\sum_{\mathbf{x}\sim\hat{\mathcal{D}}_{S_i}} \mathbb{I}(h(\mathbf{x}=0))\right)\right)
\end{aligned}$$

A.2.5 Proof of Thm. 6.4.1

Theorem 6.4.1. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, $\forall\alpha\in\mathbb{R}_+^k, \sum_{i\in[k]}\alpha_i = 1$, for $0<\delta<1$, with probability at least $1-\delta$ (over the choice of samples), we have:*

$$\varepsilon_T(h) \le \sum_{i\in[k]} \alpha_i \cdot \left(\widehat{\varepsilon}_{S_i}(h) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \widehat{\mathcal{D}}_{S_i})\right) + O\left(\sqrt{\frac{1}{m}\left(\log\frac{k}{\delta} + d\log\frac{me}{d}\right)}\right) + \lambda_\alpha \quad (6.7)$$

where λ_α is a constant that only depends on $\mathcal{H}$.

Proof. We first extend the definition of $\mathcal{H}$-divergence in the multiple sources setting from Def. 6.3.1 to the case where a convex combination α is used to combine all the single source divergence measures. Under this new divergence measure, we prove similar concentration results like Thm. 6.3.1, Thm. 6.3.2 and Thm. 6.3.3. By choosing α to be proportional to $\exp(\widehat{\varepsilon}_i)$, we obtain the upper bound in Thm. 6.4.1 using a combination of Jensen's inequality and the arithmetic-geometric mean inequality.

Let $\alpha\in\mathbb{R}^k$ be $\alpha\ge 0$ and $\sum_{i\in[k]}\alpha_i = 1$. Define $d_{\mathcal{H},\alpha}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ as follows:

Definition A.2.1.

$$d_{\mathcal{H},\alpha}(\mathcal{D}_T;\{\mathcal{D}_{S_i}\}_{i=1}^k) := \sum_{i\in[k]} \alpha_i \cdot d_{\mathcal{H}}(\mathcal{D}_T;\mathcal{D}_{S_i}) = 2\sum_{i\in[k]} \alpha_i \cdot \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{D_T}(A) - \Pr_{D_{S_i}}(A)|$$

It is easy to check that Def. A.2.1 is a generalization of Def. 6.3.1 where α is chosen to be a one-hot vector that has value 1 in the source domain with the largest discrepancy. Similarly, define h^*_α and λ_α as follows:

$$h^*_\alpha := \arg\min_{h\in\mathcal{H}} \left(\varepsilon_T(h) + \sum_{i\in[k]} \alpha_i \cdot \varepsilon_{S_i}(h)\right), \quad \lambda_\alpha := \varepsilon_T(h^*) + \sum_{i\in[k]} \alpha_i \cdot \varepsilon_{S_i}(h^*)$$

Realizing that the proof of Thm. 6.3.1 only depends on the subadditivity of the $\max$ operator, and the fact that convex combination trivially satisfies subadditivity, we can easily show that the following generalization bound holds for all such α:

$$\varepsilon_T(h) \le \sum_{i\in[k]} \alpha_i \cdot \varepsilon_{S_i}(h) + \lambda_\alpha + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H},\alpha}(\mathcal{D}_T;\{\mathcal{D}_{S_i}\}_{i=1}^k) \tag{A.1}$$

The next step is to provide finite sample bounds for the first and the third terms of (A.1). To bound the third term, we simply replace $d_{\mathcal{H}}(\mathcal{D}_T;\{\mathcal{D}_{S_i}\}_{i=1}^k)$ in the proof of Thm. 6.3.2 to $d_{\mathcal{H},\alpha}(\mathcal{D}_T;\{\mathcal{D}_{S_i}\}_{i=1}^k)$, and use the following inequality:

$$\begin{aligned}
&\Pr\left(\sum_{i\in[k]} \alpha_i \cdot \sup_{A\in\mathcal{A}_{\mathcal{H}}} \left| |\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)| \right| \ge \frac{\epsilon}{2}\right) \\
\le\ &\Pr\left(\exists i\in[k], \exists A\in\mathcal{A}_{\mathcal{H}} : \left| |\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)| \right| \ge \frac{\epsilon}{2}\right) \\
\le\ &\sum_{i=1}^k \Pr\left(\exists A\in\mathcal{A}_{\mathcal{H}} : \left| |\Pr_{\mathcal{D}_T}(A) - \Pr_{\mathcal{D}_{S_i}}(A)| - |\Pr_{\hat{\mathcal{D}}_T}(A) - \Pr_{\hat{\mathcal{D}}_{S_i}}(A)| \right| \ge \frac{\epsilon}{2}\right)
\end{aligned}$$

where the first inequality is due to the fact that $\sum_{i\in[k]} \alpha_i t_i \ge \epsilon/2 \implies \exists i\in[k], t_i \ge \epsilon/2$, otherwise $\sum_{i\in[k]} \alpha_i t_i < \epsilon/2$, and the second inequality is a simple union bound. All the other parts of the proof of Thm. 6.3.2 still hold under $d_{\mathcal{H},\alpha}(\mathcal{D}_T;\{\mathcal{D}_{S_i}\}_{i=1}^k)$, hence we immediately have the following lemma

to estimate $d_{\mathcal{H},\alpha}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k)$ using finite samples:

Lemma A.2.3. *Let $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be the target distribution and k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\widehat{\mathcal{D}}_T$ and $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\mathcal{D}_T$ and $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then for $\epsilon > 0$, we have:*

$$\Pr\left(\left|d_{\mathcal{H},\alpha}(\mathcal{D}_T; \{\mathcal{D}_{S_i}\}_{i=1}^k) - d_{\mathcal{H},\alpha}(\widehat{\mathcal{D}}_T; \{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k)\right| \geq \epsilon\right) \leq 4k\left(\frac{em}{d}\right)^d \exp\left(-m\epsilon^2/8\right)$$

To bound the first term uniformly for all $h \in \mathcal{H}$, we have:

Lemma A.2.4. *Let $\{\mathcal{D}_{S_i}\}_{i=1}^k$ be k source distributions over $\mathcal{X}$. Let $\mathcal{H}$ be a hypothesis class where $VC\text{-}dim(\mathcal{H}) = d$. If $\{\widehat{\mathcal{D}}_{S_i}\}_{i=1}^k$ are the empirical distributions of $\{\mathcal{D}_{S_i}\}_{i=1}^k$ generated with m $i.i.d.$ samples from each domain, then, for $\epsilon > 0$, we have:*

$$\Pr\left(\sup_{h\in\mathcal{H}}\left|\sum_{i\in[k]}\alpha_i \cdot \varepsilon_{S_i}(h) - \sum_{i\in[k]}\alpha_i \cdot \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \leq 2k\left(\frac{me}{d}\right)^d \exp(-2m\epsilon^2)$$

The proof of the above lemma is as follows:

$$\begin{aligned}
\Pr\left(\sup_{h\in\mathcal{H}}\left|\sum_{i\in[k]}\alpha_i \cdot \varepsilon_{S_i}(h) - \sum_{i\in[k]}\alpha_i \cdot \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) &\leq \Pr\left(\sup_{h\in\mathcal{H}}\sum_{i\in[k]}\alpha_i \left|\varepsilon_{S_i}(h) - \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \\
&\leq \Pr\left(\sum_{i\in[k]}\alpha_i \cdot \sup_{h\in\mathcal{H}}\left|\varepsilon_{S_i}(h) - \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \\
&\leq \Pr\left(\exists i \in [k] : \sup_{h\in\mathcal{H}}\left|\varepsilon_{S_i}(h) - \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \\
&\leq \sum_{i=1}^k \Pr\left(\sup_{h\in\mathcal{H}}\left|\varepsilon_{S_i}(h) - \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \\
&\leq k \cdot \Pi_{\mathcal{H}}(m) \Pr\left(\left|\varepsilon_{S_i}(h) - \hat{\varepsilon}_{S_i}(h)\right| \geq \epsilon\right) \\
&\leq k \cdot \Pi_{\mathcal{H}}(m) \cdot 2\exp(-2m\epsilon^2) \\
&\leq 2k\left(\frac{me}{d}\right)^d \exp(-2m\epsilon^2)
\end{aligned}$$

The first inequality is due to the triangle inequality of $|\cdot|$, and the second one is because of the

subadditivity of the $\sup$ function. The third inequality holds by a contrapositive argument. The fourth one is by the union bound. We apply the VC-inequality to bound the fifth inequality, and Hoeffding's inequality to bound the sixth. Again, the last one is due to Sauer's lemma.

To complete the proof, we simply combine Lemma A.2.3 and Lemma A.2.4 into (A.1), and solve for ϵ. ■

A.2.6 Proof of Thm. 6.4.2

Theorem 6.4.2. *Choose* $\alpha_i = \exp\left(\widehat{\varepsilon}_i(h)\right) / \sum_{j\in[k]} \exp\left(\widehat{\varepsilon}_j(h)\right)$ *with* $\widehat{\varepsilon}_i(h) := \widehat{\varepsilon}_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}\left(\widehat{\mathcal{D}}_T; \widehat{\mathcal{D}}_{S_i}\right)$, *then we have*

$$\varepsilon_T(h) \leq \log \sum_{i\in[k]} \exp\left(\widehat{\varepsilon}_i(h)\right) + O\left(\sqrt{\frac{1}{m}\left(\log\frac{k}{\delta} + d\log\frac{me}{d}\right)}\right) + \lambda^* \tag{6.8}$$

where λ^* *is a constant that only depends on* $\mathcal{H}$.

Proof. Define $\alpha_i = \exp(\widehat{\varepsilon}_i(h)) / \sum_{j\in[k]} \exp(\widehat{\varepsilon}_j(h))$ with $\widehat{\varepsilon}_i(h) := \widehat{\varepsilon}_{S_i}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\widehat{\mathcal{D}}_T; \widehat{\mathcal{D}}_{S_i})$, then the first term of the R.H.S. of (6.7) can be bounded as:

$$\begin{aligned}
\sum_{i\in[k]} \frac{\exp(\widehat{\varepsilon}_i(h))}{\sum_{j\in[k]} \exp(\widehat{\varepsilon}_j(h))} \cdot \widehat{\varepsilon}_i(h) &= \mathbb{E}_\alpha[\widehat{\varepsilon}_i(h)] = \mathbb{E}_\alpha[\log\exp(\widehat{\varepsilon}_i(h))] \\
&\leq \log\left(\mathbb{E}_\alpha[\exp(\widehat{\varepsilon}_i(h))]\right) \\
&= \log\left(\frac{\sum_{i\in[k]} \exp^2(\widehat{\varepsilon}_i(h))}{\sum_{i\in[k]} \exp(\widehat{\varepsilon}_i(h))}\right) \\
&\leq \log \sum_{i\in[k]} \exp(\widehat{\varepsilon}_i(h))
\end{aligned}$$

where the first inequality is due to the Jensen's inequality, and the second one is based on the fact that $\sum_i a_i^2 \leq (\sum_i a_i)^2$ when $a_i \geq 0, \forall i$. ■

Appendix B

Experimental Details

In this section, we describe more details about the datasets and the experimental settings. We extensively evaluate the proposed methods on three datasets: 1). We first evaluate our methods on Amazon Reviews dataset [22] for sentiment analysis. 2). We evaluate the proposed methods on the digits classification datasets including MNIST [62], MNIST-M [35], SVHN [77], and SynthDigits [35]. 3). We further evaluate the proposed methods on the public dataset CityCam [120] for vehicle counting. It contains 60,000 labeled images from 12 city cameras with different distributions. Due to the substantial difference between these datasets and their corresponding learning tasks, we will introduce more detailed dataset description, network architecture, and training parameters for each dataset respectively in the following subsections.

B.1 Details on Amazon Reviews Evaluation

Amazon reviews dataset includes four domains, each one composed of reviews on a specific kind of product (Books, DVDs, Electronics, and Kitchen appliances). Reviews are encoded as 5000 dimensional feature vectors of unigrams and bigrams. The labels are binary: 0 if the product is ranked up to 3 stars, and 1 if the product is ranked 4 or 5 stars.

We take one product domain as target and the other three as source domains. Each source domain has 2000 labeled examples and the target test set has 3000 to 6000 examples. We implement the Hard-Max

and Soft-Max methods according to Alg. 3, based on a basic network with one input layer (5000 units) and three hidden layers (1000, 500, 100 units). The network is trained for 50 epochs with dropout rate 0.7. We compare Hard-Max and Soft-Max with three baselines: *Baseline 1: MLPNet*. It is the basic network of our methods (one input layer and three hidden layers), trained for 50 epochs with dropout rate 0.01. *Baseline 2: Marginalized Stacked Denoising Autoencoders (mSDA)* [22]. It takes the unlabeled parts of both source and target samples to learn a feature map from input space to a new representation space. As a denoising autoencoder algorithm, it finds a feature representation from which one can (approximately) reconstruct the original features of an example from its noisy counterpart. *Baseline 3: DANN*. We implement DANN based on the algorithm described in [35] with the same basic network as our methods. Hyper parameters of the proposed and baseline methods are selected by cross validation. Table B.1 summarizes the network architecture and some hyper parameters.

Method	**Input layer**	**Hidden layers**	**Epochs**	**Dropout**	**Domains**	**Adaptation weight**	γ
MLPNet	5000	(1000, 500, 100)	50	0.01	N/A	N/A	N/A
DANN	5000	(1000, 500, 100)	50	0.01	1	0.01	N/A
MDAN	5000	(1000, 500, 100)	50	0.7	3	0.1	10

Table B.1: Network parameters for proposed and baseline methods

B.2 Details on Digit Datasets Evaluation

We evaluate the proposed methods on the digits classification problem. Following the experiments in [35], we combine four popular digits datasets-MNIST, MNIST-M, SVHN, and SynthDigits to build the multi-source domain dataset. MNIST is a handwritten digits database with $60,000$ training examples, and $10,000$ testing examples. The digits have been size-normalized and centered in a 28×28 image. MNIST-M is generated by blending digits from the original MNIST set over patches randomly extracted from color photos from BSDS500 [4, 35]. It has $59,001$ training images and $9,001$ testing images with 32×32 resolution. An output sample is produced by taking a patch from a photo and inverting its pixels at positions corresponding to the pixels of a digit. For DA problems, this domain

is quite distinct from MNIST, for the background and the strokes are no longer constant. SVHN is a real-world house number dataset with $73,257$ training images and $26,032$ testing images. It can be seen as similar to MNIST, but comes from a significantly harder, unsolved, real world problem. SynthDigits consists of $500;000$ digit images generated by [35] from WindowsTM fonts by varying the text, positioning, orientation, background and stroke colors, and the amount of blur. The degrees of variation were chosen to simulate SVHN, but the two datasets are still rather distinct, with the biggest difference being the structured clutter in the background of SVHN images.

We take MNIST-M, SVHN, and MNIST as target domain in turn, and the remaining three as sources. We implement the Hard-Max and Soft-Max versions according to Alg. 3 based on a basic network, as shown in Fig. B.1. The baseline methods are also built on the same basic network structure to put them on a equal footing. The network structure and parameters of MDANs are illustrated in Fig. B.1. The learning rate is initialized by 0.01 and adjusted by the first and second order momentum in the training process. The domain adaptation parameter of MDANs is selected by cross validation. In each mini-batch of MDANs training process, we randomly sample the same number of unlabeled target images as the number of the source images.

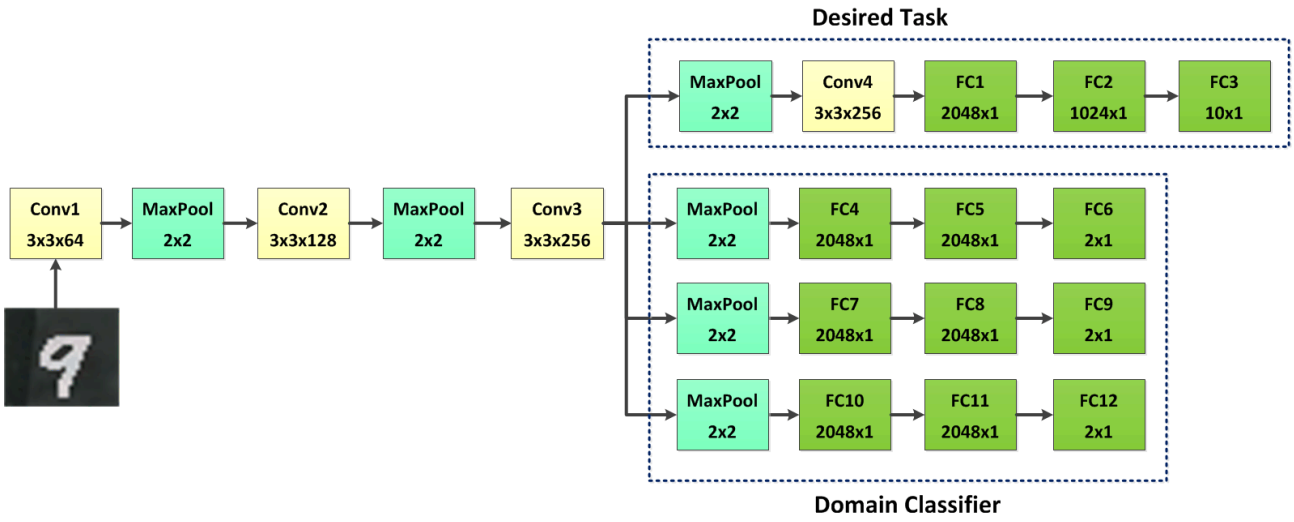


Figure B.1: MDANs network architecture for digit classification

B.3 Details on CityCam Vehicle Counting

CityCam is a public dataset for large-scale city camera videos, which have low resolution, low frame rate, and high occlusion. CityCam has $60,000$ frames annotated with rich information: bounding box, vehicle type, vehicle orientation, vehicle count, vehicle re-identification, and weather condition. The dataset is divided into training and testing sets, with 42,200 and 17,800 frames, respectively, covering multiple cameras and different weather conditions. CityCam is an appropriate dataset to evaluate domain adaptation methods, for it covers multiple city cameras and each camera is located in different intersection of the city with different perspectives and scenes. Thus, each camera data has different distribution from others. The dataset is quite challenging and in high demand of domain adaptation solutions, as it has $6,000,000$ unlabeled images from 200 cameras with only $60,000$ labeled images from 12 cameras. The experiments on CityCam provide an interesting application of our proposed MDANs: when dealing with spatially and temporally large-scale dataset with much variations, it is prohibitively expensive and time-consuming to label large amount of instances covering all the variations. As a result, only a limited portion of the dataset can be annotated, which can not cover all the data domains in the dataset. MDAN provide an effective solution for this kind of application by adapting the deep model from multiple source domains to the unlabeled target domain.

We evaluate the proposed methods on different numbers of source cameras. Each source camera provides 2000 labeled images for training and the test set has 2000 images from the target camera. In each mini-batch, we randomly sample the same number of unlabeled target images as the source images. We implement the Hard-Max and Soft-Max version of MDANs according to Alg. 3, based on the basic FCN vehicle counting network of FCN-rLSTM introduced in Chapter 5. Please refer to Chapter 5 for detailed network architecture and parameters. The input images are resized into 224×224. The learning rate is initialized by 0.01 and adjusted by the first and second order momentum in the training process. The domain adaptation parameter is selected by cross validation. We compare our method with two baselines: *Baseline 1: FCN*. It is our basic network without domain adaptation as introduced in work [120]. *Baseline 2: DANN*. We implement DANN on top of the same basic network following the algorithm introduced in work [35].

Bibliography

[1] H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, and M. Marchand. Domain-adversarial neural networks. *arXiv preprint arXiv:1412.4446*, 2014.

[2] S. An, W. Liu, and S. Venkatesh. Face recognition using kernel ridge regression. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2007.

[3] M. Anthony and P. L. Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

[4] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.

[5] C. Arteta, V. Lempitsky, and A. Zisserman. Counting in the wild. In *European Conference on Computer Vision*, pages 483–498. Springer, 2016.

[6] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. In *International Workshop on Human Behavior Understanding*, pages 29–39. Springer, 2011.

[7] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learninfag to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[8] M. Baktashmotlagh, M. T. Harandi, B. C. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 769–776, 2013.

[9] C. J. Becker, C. M. Christoudias, and P. Fua. Non-linear domain adaptation with boosting. In *Advances in Neural Information Processing Systems*, pages 485–493, 2013.

[10] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, et al. Analysis of representations for domain adaptation. *Advances in Neural Information Processing Systems*, 19:137, 2007.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 2010.

[12] J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128. Association for Computational Linguistics, 2006.

[13] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman. Learning bounds for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 129–136, 2008.

[14] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks. In *Advances in Neural Information Processing Systems*, pages 343–351, 2016.

[15] A. B. C. Rother, V. Kolmogorov. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM SIGGRAPH, 2004*. ACM, 2004.

[16] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray. Autonomous driving in urban environments: approaches, lessons and challenges. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 368(1928):4649–4672, 2010.

[17] A. B. Chan, Z.-S. J. Liang, and N. Vasconcelos. Privacy preserving crowd monitoring: Counting people without people models or tracking. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008.*, pages 1–7. IEEE, 2008.

[18] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.

[19] K. Chen, C. C. Loy, S. Gong, and T. Xiang. Feature mining for localised crowd counting. In

The British Machine Vision Conference (BMVC), volume 1, 2012.

[20] K. Chen, S. Gong, T. Xiang, and C. Change Loy. Cumulative attribute space for age and crowd density estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2467–2474, 2013.

[21] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[22] M. Chen, Z. Xu, K. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683*, 2012.

[23] Y.-L. Chen, B.-F. Wu, H.-Y. Huang, and C.-J. Fan. A real-time vision system for nighttime vehicle detection and traffic surveillance. *IEEE Transactions on Industrial Electronics*, 58(5): 2030–2044, 2011.

[24] Z. Chen, T. Ellis, and S. A. Velastin. Vehicle detection, tracking and classification in urban traffic. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 951–956. IEEE, 2012.

[25] T. E. Choe, M. W. Lee, and N. Haering. Traffic analysis with low frame rate camera networks. In *IEEE Conference on Computer Vision and Pattern Recognition-Workshops*, 2010.

[26] C. Cortes and M. Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519:103–126, 2014.

[27] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In *International Conference on Algorithmic Learning Theory*, pages 38–53. Springer, 2008.

[28] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Statistic and knowledge-based moving object detection in traffic scenes. In *Proceedings of Intelligent Transportation Systems.*, pages 27–32. IEEE, 2000.

[29] V. Dangi, A. Parab, K. Pawar, and S. Rathod. Image processing based intelligent traffic controller. *Undergraduate Academic Research Journal (UARJ)*, 1(1), 2012.

[30] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of International Conference on Machine Learning*, volume 32, pages 647–655, 2014.

[31] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[32] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1936.

[33] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117. ACM, 2004.

[34] C. Gan, T. Yang, and B. Gong. Learning attributes equals multi-source domain generalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 87–97, 2016.

[35] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

[36] K. Garg, S.-K. Lam, T. Srikanthan, and V. Agarwal. Real-time road traffic density estimation using block variance. In *IEEE Winter Conference on Applications of Computer Vision*, 2016.

[37] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[38] P. Germain, A. Habrard, F. Laviolette, and E. Morvant. A pac-bayesian approach for domain adaptation with specialization to linear classifiers. In *Proceedings of International Conference on Machine Learning*, pages 738–746, 2013.

[39] M. Ghifary, W. Bastiaan Kleijn, M. Zhang, and D. Balduzzi. Domain generalization for object

recognition with multi-task autoencoders. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2551–2559, 2015.

[40] X. Glorot, A. Bordes, and Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

[41] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *Proceedings of International Conference on Machine Learning*, pages 222–230, 2013.

[42] R. Gopalan, R. Li, and R. Chellappa. Unsupervised adaptation across domain shifts by generating intermediate data representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2288–2302, 2014.

[43] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[44] R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Onoro-Rubio. Extremely overlapping vehicle counting. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 423–431. Springer, 2015.

[45] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[46] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[47] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[48] J. Hoffman, B. Kulis, T. Darrell, and K. Saenko. Discovering latent domains for multisource domain adaptation. In *European Conference on Computer Vision*, pages 702–715. Springer,

2012.

[49] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu. Drone-based object counting by spatially regularized regional proposal network. *arXiv preprint arXiv:1707.05972*, 2017.

[50] S. Hua, J. Wua, and L. Xub. Real-time traffic congestion detection based on video analysis. *J. Inf. Comput. Sci*, 9(10):2907–2914, 2012.

[51] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems*, pages 601–608, 2006.

[52] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[53] I.-H. Jhuo, D. Liu, D. Lee, and S.-F. Chang. Robust visual domain adaptation with low-rank reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2168–2175. IEEE, 2012.

[54] A. Kanungo, A. Sharma, and C. Singla. Smart traffic lights switching and traffic density calculation using video processing. In *Engineering and computational sciences (RAECS), 2014 recent advances in*, pages 1–6. IEEE, 2014.

[55] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359–381, 2003.

[56] B. S. Kerner. *Introduction to modern traffic flow theory and control: the long road to three-phase traffic theory*. Springer Science & Business Media, 2009.

[57] G. S. Khekare and A. V. Sakhare. A smart city framework for intelligent traffic system using vanet. In *Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on*, pages 302–305. IEEE, 2013.

[58] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 180–191. VLDB Endowment, 2004.

[59] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[60] V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[62] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[63] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, et al. Towards fully autonomous driving: Systems and algorithms. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 163–168. IEEE, 2011.

[64] T. Liu, Y. Zheng, L. Liu, Y. Liu, and Y. Zhu. Methods for sensing urban noises. *Tec. Rep. MSR-TR-2014-66*, 2014.

[65] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[66] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[67] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[68] L. Maddalena and A. Petrosino. The 3dsobs+ algorithm for moving object detection. *Computer Vision and Image Understanding*, 122:65–73, 2014.

[69] Y. Mansour and M. Schain. Robust domain adaptation. In *ISAIM*, 2012.

[70] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[71] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems*, pages 1041–1048, 2009.

[72] Y. Mansour, M. Mohri, and A. Rostamizadeh. Multiple source adaptation and the rényi divergence. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 367–374. AUAI Press, 2009.

[73] G. Mo and S. Zhang. Vehicles detection in traffic flow. In *Sixth International Conference on Natural Computation*, volume 2, pages 751–754. IEEE, 2010.

[74] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[75] U. Nagaraj, J. Rathod, P. Patil, S. Thakur, and U. Sharma. Traffic jam detection using image processing. *International Journal of Engineering Research and Applications*, 3(2):1087–1091, 2013.

[76] D. Neil, M. Pfeiffer, and S.-C. Liu. Phased lstm: Accelerating recurrent network training for long or event-based sequences. In *Advances in Neural Information Processing Systems*, pages 3882–3890, 2016.

[77] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[78] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

[79] D. Onoro-Rubio and R. J. López-Sastre. Towards perspective-free object counting with deep learning. In *European Conference on Computer Vision*, pages 615–629. Springer, 2016.

[80] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[81] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks.

Proceedings of International Conference on Machine Learning, 28:1310–1318, 2013.

[82] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

[83] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[84] J. Rosenzweig and M. Bartl. A review and analysis of literature on autonomous driving.

[85] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[86] R. Shirani, F. Hendessi, and T. A. Gulliver. Store-carry-forward message dissemination in vehicular ad-hoc networks with local density estimation. In *Vehicular Technology Conference Fall (VTC 2009-Fall), 2009 IEEE 70th*, pages 1–6. IEEE, 2009.

[87] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. *arXiv preprint arXiv:1612.07828*, 2016.

[88] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[89] V. A. Sindagi and V. M. Patel. A survey of recent advances in cnn-based single image crowd counting and density estimation. *Pattern Recognition Letters*, 2017.

[90] K. Singh and B. Li. Estimation of traffic densities for multilane roadways using a markov model approach. *IEEE Transactions on industrial electronics*, 59(11):4369–4376, 2012.

[91] A. Sobral and A. Vacavant. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. *Computer Vision and Image Understanding*, 122:4–21, 2014.

[92] M. Soh. Learning cnn-lstm architectures for image caption generation.

[93] K. SuganyaDevi, N. Malmurugan, and R. Sivakumar. Efficient foreground extraction based on

optical flow and smed for road traffic analysis. *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, 1(3):177–182, 2012.

[94] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[95] G. S. Thakur, P. Hui, H. Ketabdar, and A. Helmy. Spatial and temporal analysis of planet scale vehicular imagery data. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pages 905–910. IEEE, 2011.

[96] G. S. Thakurzx, P. Huiz, and A. Helmyx. Modeling and characterization of urban vehicular mobility using web cameras. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 262–267. IEEE, 2012.

[97] E. Toropov, L. Gui, S. Zhang, S. Kottur, and J. M. F. Moura. Traffic flow from a low frame rate city camera. In *IEEE International Conference on Image Processing (ICIP)*. IEEE, 2015.

[98] C.-M. Tsai and Z.-M. Yeh. Intelligent moving objects detection via adaptive frame differencing method. In *Asian Conference on Intelligent Information and Database Systems*, pages 1–11. Springer, 2013.

[99] Y. Tsuboi, H. Kashima, S. Hido, S. Bickel, and M. Sugiyama. Direct density ratio estimation for large-scale covariate shift adaptation. *Journal of Information Processing*, 17:138–155, 2009.

[100] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4068–4076, 2015.

[101] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017.

[102] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008.

[103] A. Z. V. Lempitsky. Learning to count objects in images. In *Advances in Neural Information Processing Systems (NIPS)*. ACM, 2010.

[104] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[105] A. van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances In Neural Information Processing Systems*, pages 4790–4798, 2016.

[106] C. P. van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. Van Lint, and H. J. van Zuylen. Localized extended kalman filter for scalable real-time traffic state estimation. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):385–394, 2012.

[107] V. Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

[108] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008.

[109] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1096–1103. ACM, 2008.

[110] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

[111] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. Cnn-rnn: A unified framework for multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2285–2294, 2016.

[112] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. DETRAC: A new benchmark and protocol for multi-object detection and tracking. *arXiv CoRR*, abs/1511.04136, 2015.

[113] H. Xu and S. Mannor. Robustness and generalization. *Machine Learning*, 86(3):391–423, 2012.

[114] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural

networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014.

[115] X.-D. Yu, L.-Y. Duan, and Q. Tian. Highway traffic information extraction from skycam mpeg video. In *Intelligent Transportation Systems, 2002. Proceedings. The IEEE 5th International Conference on*, pages 37–42. IEEE, 2002.

[116] Y. Yuan, J. Van Lint, R. E. Wilson, F. van Wageningen-Kessels, and S. P. Hoogendoorn. Real-time lagrangian traffic state estimator for freeways. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):59–70, 2012.

[117] C. Zhang, H. Li, X. Wang, and X. Yang. Cross-scene crowd counting via deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 833–841, 2015.

[118] J. Zhang, Y. Zheng, and D. Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. *arXiv preprint arXiv:1610.00081*, 2016.

[119] K. Zhang, M. Gong, and B. Schölkopf. Multi-source domain adaptation: A causal view. In *AAAI Conference on Artificial Intelligence*, pages 3150–3157, 2015.

[120] S. Zhang, G. Wu, J. P. Costeira, and J. M. Moura. Understanding traffic density from large-scale web camera data. *arXiv preprint arXiv:1703.05868*, 2017.

[121] S. Zhang, G. Wu, J. P. Costeira, and J. M. F. Moura. Understanding traffic density from large-scale web camera data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5898–5907. IEEE, 2017.

[122] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 589–597, 2016.

[123] Z. Zhao, H. Li, R. Zhao, and X. Wang. Crossing-line crowd counting with two-phase deep neural networks. In *European Conference on Computer Vision*, pages 712–726. Springer, 2016.

[124] Y. Zheng and S. Peng. Model based vehicle localization for urban traffic surveillance using image gradient based matching. In *2012 15th International IEEE Conference on Intelligent*

Transportation Systems, pages 945–950. IEEE, 2012.