Deep Visual Feature Learning for Vehicle Detection, Recognition and Re-identification

Yi Zhou

School of Computing Sciences
University of East Anglia

This dissertation is submitted for the degree of
Doctor of Philosophy

April 2018

Declaration

I hereby declare that this dissertation is all my own work. Parts of the thesis have been published in academic conference proceedings and journal articles. All these papers were authored / co-authored by me, Yi Zhou, during and as a result of my Ph.D. research.

Zhou, Y. and Shao, L., 2018, Jun. Viewpoint-aware Attentive Multi-view Inference for Vehicle Re-identification. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, USA. IEEE. (Chapter 5)

Zhou, Y., Liu, L. and Shao, L., 2018. Vehicle Re-identification by Deep Hidden Multi-View Inference. *IEEE Transactions on Image Processing*. DOI: 10.1109/TIP.2018.2819820 (Chapter 4)

Zhou, Y. and Shao, L., 2018, Mar. Vehicle Re-identification by Adversarial Bi-directional LSTM Network. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, USA. IEEE. (Chapter 5)

Zhou, Y., Liu, L., Shao, L. and Mellor, M., 2017. Fast Automatic Vehicle Annotation for Urban Traffic Surveillance. *IEEE Transactions on Intelligent Transportation Systems*. DOI: 10.1109/TITS.2017.2740303 (Chapter 3)

Zhou, Y. and Shao, L., 2017, Sep. Cross-View GAN Based Vehicle Generation for Re-identification. Proceedings of the *British Machine Vision Conference (BMVC)*, London, UK. BMVA Press. [Oral. Acceptance rate 5.6%] (Chapter 4)

Zhou, Y., Liu, L., Shao, L. and Mellor, M., 2016, October. DAVE: a unified framework for fast vehicle detection and annotation. In *European Conference on Computer Vision (ECCV)* (pp. 278-293), Amsterdam. Springer International Publishing. (Chapter 3)

Liu, L., **Zhou, Y.** and Shao, L., 2018. Deep Action Parsing in Videos with Large-scale Synthesized Data. *IEEE Transactions on Image Processing*. DOI: 10.1109/TIP.2018.2813530

Liu, L., **Zhou, Y.** and Shao, L., 2017, May. DAP3D-Net: Where, what and how actions occur in videos?. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on* (pp. 138-145), Singapore. IEEE.

Yi Zhou
April 2018

Acknowledgements

I would like to convey my gratefulness to the following, for their excellent help in my Ph.D.

First of all, I would like to express my deepest thanks to my parents. They have been always providing their best love and unconditional support to me.

I would like to especially thank my primary supervisor Prof. Ling Shao, who has been kindly advising me in my Ph.D. and taught me two modules in my MSc. Under his supervision, he provides me with continuous support, patience, motivation, enthusiasm and immense knowledge which help me successfully finished my Ph.D. work. In addition to guiding me novel research ideas, he is also willing to discuss with me. He brought me into the interesting computer vision and machine learning research community.

I would like to thank the CREATEC Limited Corporation for the financial support of my living cost. Particularly, I very much appreciate Dr. Matt Mellor, who is the director of CREATEC, for his invaluable suggestions in doing industrial projects. I would also like to thank Dr. David Clark and Patrick Gordon for their technical support in collaborative projects.

I am indebted to Prof. Gerard Parr and Prof. Andy Day for their kind help in the later period of my Ph.D., and the great suggestions to my thesis.

I would like to thank my fellow colleagues in our laboratory. I am very grateful to my co-author, Dr. Li Liu, for his talented ideas and help, and other lab mates: Dr. Mengyang Yu, Dr. Feng Zheng, Dr. Yang Long, Dr. Ziyun Cai, Dr. Jungong Han, Dr. Lining Zhang, Yuming Shen, Bingzhang Hu, and Shidong Wang for their witty insights and refreshing discussions. My thanks also go to many visiting staffs and students in our lab: Dr. Haofeng Zhang, Dr. Xiaoming Liu, Yang Liu and Jin Li for their kind assistance. Thank all of them making daily research a very fun job.

The PGR director Dr. Katharina Huber and other administrative staffs in the School of Computing Sciences of the University of East Anglia, including Matthew Ladd and Binoop Pulikkottil John, have provided me with support in various matters. I appreciate their effort and kindness to me.

I would like to thank Dr. Jinchang Ren and Dr. Michal Mackiewicz as my examiners for carefully reviewing my thesis.

Finally, I would like to thank my loving wife Jiajun for her consistent encouragement to me when I was depressed and taking care of my life in the UK. To her, I dedicate this thesis.

Abstract

Along with the ever-increasing number of motor vehicles in current transportation systems, intelligent video surveillance and management becomes more necessary which is one of the important artificial intelligence fields. Vehicle-related problems are being widely explored and applied practically. Among various techniques, computer vision and machine learning algorithms have been the most popular ones since a vast of video/image surveillance data are available for research, nowadays. In this thesis, vision-based approaches for vehicle detection, recognition, and re-identification are extensively investigated. Moreover, to address different challenges, several novel methods are proposed to overcome weaknesses of previous works and achieve compelling performance.

Deep visual feature learning has been widely researched in the past five years and obtained huge progress in many applications including image classification, image retrieval, object detection, image segmentation and image generation. Compared with traditional machine learning methods which consist of hand-crafted feature extraction and shallow model learning, deep neural networks can learn hierarchical feature representations from low-level to high-level features to get more robust recognition precision. For some specific tasks, researchers prefer to embed feature learning and classification/regression methods into end-to-end models, which can benefit both the accuracy and efficiency. In this thesis, deep models are mainly investigated to study the research problems.

Vehicle detection is the most fundamental task in intelligent video surveillance but faces many challenges such as severe illumination and viewpoint variations, occlusions and multi-scale problems. Moreover, learning vehicles' diverse attributes is also an interesting and valuable problem. To address these tasks and their difficulties, a fast framework of Detection and Annotation for Vehicles (DAVE) is presented, which effectively combines vehicle detection and attributes annotation. DAVE consists of two convolutional neural networks (CNNs): a fast vehicle proposal network (FVPN) for vehicle-like objects extraction and an attributes learning network (ALN) aiming to verify each proposal and infer each vehicle's pose, color and type simultaneously. These two nets are jointly optimized so that the abundant latent knowledge learned from the ALN can be exploited to guide FVPN

training. Once the model is trained, it can achieve efficient vehicle detection and annotation for real-world traffic surveillance data.

The second research problem of the thesis focuses on vehicle re-identification (re-ID). Vehicle re-ID aims to identify a target vehicle in different cameras with non-overlapping views. It has received far less attention in the computer vision community than the prevalent person re-ID problem. Possible reasons for this slow progress are the lack of appropriate research data and the special 3D structure of a vehicle. Previous works have generally focused on some specific views (e.g. front), but these methods are less effective in realistic scenarios where vehicles usually appear in arbitrary viewpoints to cameras. In this thesis, I focus on the uncertainty of vehicle viewpoint in re-ID, proposing four different approaches to address the multi-view vehicle re-ID problem: (1) The Spatially Concatenated ConvNet (SCCN) in an encoder-decoder architecture is proposed to learn transformations across different viewpoints of a vehicle, and then spatially concatenate all the feature maps for further fusing them into a multi-view feature representation. (2) A Cross-View Generative Adversarial Network (XVGAN) is designed to take an input image's feature as conditional embedding to effectively infer cross-view images. The features of the inferred and original images are combined to learn distance metrics for re-ID. (3) The great advantages of a bi-directional Long Short-Term Memory (LSTM) loop are investigated of modeling transformations across continuous view variation of a vehicle. (4) A Viewpoint-aware Attentive Multi-view Inference (VAMI) model is proposed, adopting a viewpoint-aware attention model to select core regions at different viewpoints and then performing multi-view feature inference by an adversarial training architecture.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Vision-based Intelligent Transportation Systems

Intelligent transportation systems (ITS) are defined as the systems exploiting techniques of sensing, analysis, control, and communications to ground transportation so as to develop and improve mobility, efficiency and security. ITS contains a wide range of applications [130, 9] that process and transmit data to mitigate traffic congestion, benefit traffic management and improve rapid responses to emergent accidents.

Video surveillance is one of the main branches in public transportation systems and has a huge potential to be researched since it contributes to the planning and control of the traffic networks. With the ever-increasing number of vehicles in the world, the exploration of advanced intelligent algorithms becomes more important and imperative. Previously, observing and analyzing video data requires large manpower, which is highly inefficient. However, in the past decade, the decreasing costs of surveillance cameras make huge traffic data recorded and available to be utilized. There have been many computer vision methods developed to analyze video surveillance data.

In urban traffic systems, certain surveillance tasks such as vehicle counting [6], license plate recognition [14] and incident detection [28], can be addressed either by sensor-based or vision-based algorithms. However, vision-based methods can fully take advantages of the abundant visual patterns to recognize target objects in a human-like way. Take the application of vehicle detection as an example, the radar sensor-based techniques can only detect vehicles in a very limited area, while the vision-based approaches can find all the vehicles in a large visible area by a camera and describe additional features of each detected vehicle simultaneously. Therefore, in this thesis, several computer vision and machine learning models are presented and extensively studied to address a variety of interesting tasks in the intelligent transportation systems.

1.2 Visual Feature Learning

In this section, an overview of deep learning for computer vision is introduced. Before talking about deep visual feature learning, we first revisit the traditional machine learning approaches. In pattern recognition problems, most of the previous efforts can be grouped into extracting robust features and learning discriminative classifiers. Feature representation is always the key to recent progress in various computer vision tasks. A conventional recognition framework can be illustrated in Figure 1.1. Given input image data, carefully designed hand-crafted features based on domain knowledge are first extracted, which do not require learning in task-specific models. Such features are low-level representations directly computed from the pixel values of images, and usually difficult and expensive to design to make machine learning algorithms work. There have been many successful low-level visual features such as SIFT [101], HOG [24], SURF [7], MSER [106], LBP [2] and GLOH [107]. Once the feature extraction is completed, traditional machine learning methods are performed for classification or regression. Among numerous approaches, linear regression, logistic regression, Bayes classification [35], decision tree [116], k-Nearest Neighbor [53], support vector machine (SVM) [52] and Adaptive Boosting (AdaBoost) [141] are the most widely adopted ones in computer vision tasks.

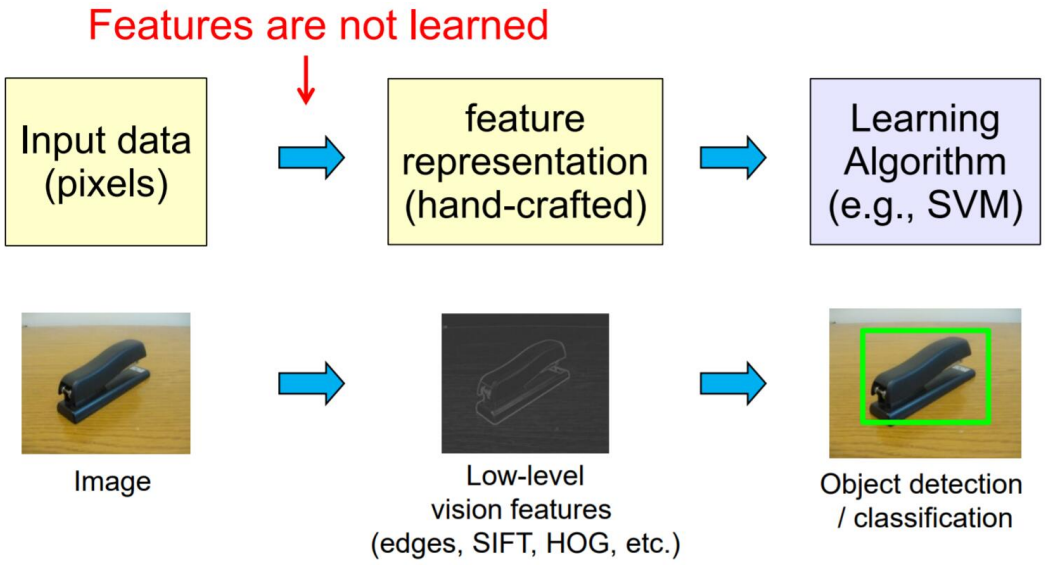


Fig. 1.1 A traditional framework for addressing pattern recognition tasks, which mainly consists of hand-crafted feature extraction and learning classifiers.

The concept of deep learning is not novel, which has been around for a couple of years now. But nowadays with the available big data and the powerful computing ability by the parallel computing platform and programming model CUDA, deep learning is getting

more attention and has achieved great success in many vision applications such as image classification [76, 128, 135], retrieval[142], segmentation[100] and object detection[38, 37]. Deep learning has two main research branches. One is the convolutional neural networks (CNNs) for addressing tasks in computer vision while the other is recurrent neural networks (RNNs) designed mainly for natural language processing. CNN is initially proposed for learning hierarchical visual features of images, where one layer extracts features from the output of its previous layer. A schematic diagram is shown in Figure 1.2. A deep CNN contains an input and an output layer as well as multiple hidden layers which usually consist of several convolutional layers, pooling layers, fully-connected layers and normalization layers. Each hidden layer is desired to learn a different level representation. For example, assume a deep network configured with three hidden layers, the first layer would learn the edge features, while the second and third layer could extract object parts' and objects' representations, respectively. Moreover, RNN is a class of neural network where connections between units form a directed cycle. This allows it to exhibit dynamic temporal behavior. Unlike forward neural network, RNNs can use their internal memory to process arbitrary sequences of inputs.

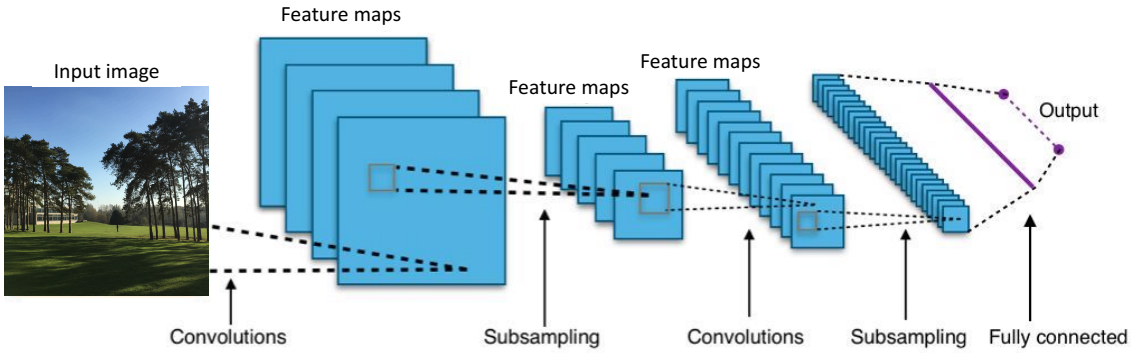


Fig. 1.2 A simple deep ConvNet for image classification.

Now we consider the approaches of learning features which can be coarsely categorized into supervised learning and unsupervised learning. The majority of practical machine learning adopts supervised learning. Supervised learning is where you have input variables x and an output variable y, and then design an algorithm to learn the mapping function from the input to the output. The goal is to approximate the mapping function so well that when you have new test data x that you can predict the output variables y for that data. Some popular examples of supervised learning algorithms are the linear regression [124] for regression problems, random forest [8] for classification and regression problems, SVM [22] for classification problems. On the other side, unsupervised learning is where you only

have input data x and no corresponding output variables. The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data. Unsupervised learning problems can be further grouped into clustering [48] and association problems [115].

Deep neural networks can be trained in an unsupervised or supervised manner for both unsupervised and supervised learning tasks. In the supervised learning, end-to-end learning of deep architectures with back-propagation is the most preferred framework. It works well when the amount of labels is large. The structure of the model is important as well. Deep neural network, convolutional neural network and recurrent neural network all have successful examples in supervised learning. Besides, deep unsupervised learning adopts layer-wise training to learn statistical structure or dependencies of the unlabeled data. Some popular examples are deep stacked denoising autoencoder [139], deep belief nets [61] and hierarchical sparse coding [66].

1.3 Object Detection

In computer vision, classification is the most well-known research topic which aims to classify an image into one of many different categories. Different from classification, localization finds the location of a single object inside the image. Iterating over the problem of localization plus classification we end up with the need for detecting and classifying multiple objects at the same time. Object detection is the problem of finding and classifying a variable number of objects on an image, which is shown in Figure 1.3.

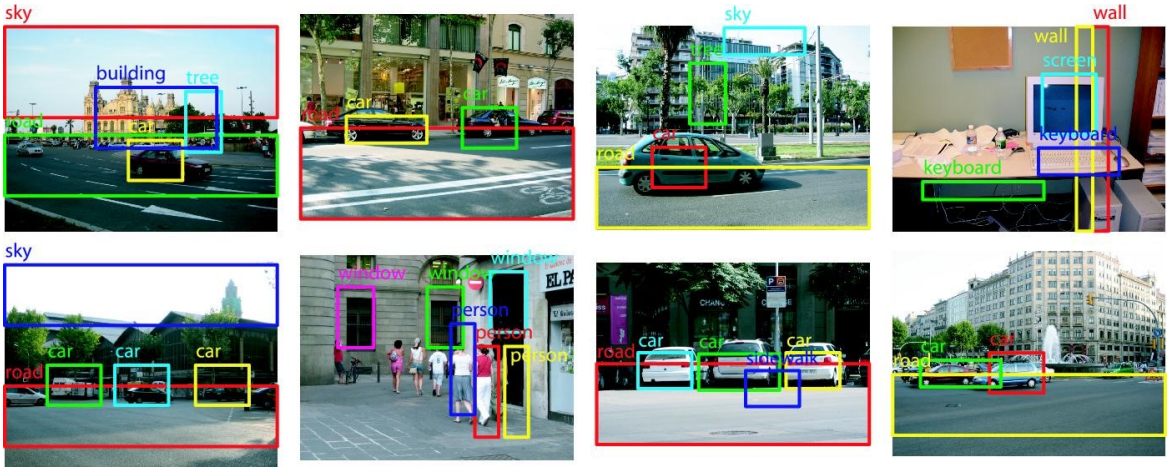


Fig. 1.3 Illustration of the object detection task: localization + classification.

Object detection is usually considered as the most fundamental task in computer vision, which encounters many challenges. The first one is the variable number of objects. When training machine learning models, we need to represent data into fixed-size vectors. Since the number of objects in the image is not known beforehand, we would not know the correct number of outputs. Historically, the variable number of outputs has been tackled using a sliding window based approach, generating the fixed-size features of that window for all the different positions of it. After getting all predictions, some are discarded and some are merged to get the final result. Another severe challenge is the different conceivable scales of objects. When doing simple classification, only objects that cover most of the image are needed to be classified. However, to detect certain objects as small as a dozen pixels (or a small percentage of the original image), traditional sliding window fashions are highly inefficient. Various other difficulties are illumination change, viewpoint variation, and deformation problems.

The most two popular classic object detection approaches are the Viola-Jones framework [141] and the deformable part models (DPM) [32]. The Viola-Jones method works by generating different simple binary classifiers using Haar features. These classifiers are assessed with a multi-scale sliding window in cascade and dropped early in case of a negative classification. The DPM method uses a histogram of oriented gradients (HOG) feature and SVM for classification. Recently, deep learning detectors combine feature learning and classifier training together to achieve satisfied performance with both the high precision and fast processing speed. In this thesis, we propose a deep CNN-based vehicle detector which has reached a superior performance over state-of-the-art methods.

1.4 Object Re-identification

Object re-identification (re-ID) is the task of recognizing an object, captured by one or more cameras, over a range of candidates targets. The re-ID problem is initially proposed to target on persons, and then extended to vehicles recently. Take vehicle re-ID as an example, the main issues are due to the fact that the same vehicle is usually acquired at different times and by different disjoint cameras. Object re-ID can be considered as a sub-area of image retrieval, dealing with matching images of the same object over multiple non-overlapping camera views.

Re-ID techniques are applicable in tracking a particular object across different cameras, tracking the long-term trajectory of a target object in surveillance, and for forensic and security applications. First, multi-camera tracking is to track an individual using multiple cameras, the identity of the object has to be retrieved from the second camera based on

the information obtained from the first camera. Second, if the locations of the cameras are known, based on the re-ID system, it is possible to track the moving path of an object from one point to another. Moreover, in surveillance and security applications, re-ID models can be employed to track the suspect in a crime scene.

At present, there are still some of the major challenges for the re-ID system being addressed in the computer vision community. Figure 1.4 gives some examples of images in a person re-ID dataset. First, illumination change causes the variation in the appeared color of the same subject across multiple cameras. As many of the older surveillance cameras give low-resolution images, it is a difficult task for the algorithm to differentiate between individuals due to the lack of appearance details. In crowded scenes, the partial or full occlusion will also severely decrease the performance. Moreover, obtaining the data for re-ID is easy, but annotated data is always scarce. For good generalization capabilities, complex algorithms need to be taught with a large number of labeled data.

Fig. 1.4 Some examples from the VIPeR [46] dataset for exploring person re-ID. Each pair of images are two shots of one person identity captured from different cameras.

Most existing re-ID approaches are carried out focusing on two aspects of the problem. To develop a mathematical representation for the images, known as a feature representation, and to develop a distance function that can reduce the distance between samples of the same identity and increase the distance between samples of different identities in an n-dimensional vector space. More details will be reviewed in Chapter 2.

1.5 Thesis Outline

The rest of the thesis is structured as follows.

Chapter 2 contains a full literature review of some representative previous works relevant to deep visual feature learning. The basics of convolutional neural networks and recurrent neural networks are mainly reviewed as well as their corresponding applications. Then, object detection approaches including traditional detection frameworks and deep learning based detectors are studied. Vehicle detection methods are specifically reviewed in a subsection. Moreover, works related to object re-identification targeting at person and vehicle are separately reviewed. Various other topics that are relevant to this thesis are also considered. Generative Adversarial Networks (GANs) achieve great success on image generation tasks because the adversarial learning forces the generated samples to be indistinguishable from real data. Visual attention mechanisms aim to automatically focus on the core regions of image inputs and ignore the useless parts. Finally, some main datasets used in our experiments and evaluation protocols are introduced.

Chapter 3 studies deep neural networks for vehicle detection and multi-task learning. It first presents a simple model using multi-level complex wavelet feature representations and deep belief neural network for training a vehicle detector. Then, a vehicle detection and annotation method DAVE, which consists of two convolutional neural networks, are proposed to jointly address vehicle detection and attributes prediction via multi-task learning.

In Chapter 4, two models for addressing vehicle re-identification via generating cross-view images are investigated. First, we present a spatially concatenated ConvNet (SCCN) to learn transformations across different viewpoint pairs of a vehicle separately in a convolutional encoder-decoder architecture, and then spatially concatenate all the feature maps and map them into a global feature for learning the distance metrics. Moreover, a novel deep cross-view generative adversarial network (XVGAN) is proposed for generating cross-view vehicle images from an input view. Through experiments, the model is extended to benefit the multi-view vehicle re-ID task.

Chapter 5 extends the works of Chapter 4 from image-level generation to feature-level transformation learning. The uncertainty of vehicle viewpoint in re-ID is still our research point. An adversarial bi-directional LSTM network (ABLN) is proposed to exploit the great advantages of the Long Short-Term Memory (LSTM) to model transformations across continuous view variation of a vehicle and adopts the adversarial architecture to enhance training. Moreover, we research the visual attention mechanism and design a viewpoint-aware attentive multi-view inference (VAMI) model. A viewpoint-aware attention model is proposed to obtain attention maps from the input image. The high-scored region of each map shows the overlapped appearance between the input vehicle's view and a target viewpoint.

Given the attentive features of a single-view input, a conditional multi-view generative network is designed to infer a global feature containing different viewpoints' information of the input vehicle. The adversarial training mechanism and auxiliary vehicle attribute classifiers are combined to achieve effective feature generation.

Finally, conclusions are drawn and future work is considered in Chapter 6.

Chapter 2

Literature Review

2.1 Deep Visual Feature Learning

2.1.1 Convolutional Neural Network

Convolutional Neural Network (CNN) [82] is an important tool for most machine learning practitioners today. A CNN model usually consists of one or more convolutional layers (often with a sub-sampling step) followed by one or more fully-connected layers as in a standard multi-layer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image, which is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have much fewer parameters than fully-connected networks with the same number of hidden units. In the following sub-sections, the main operations in a general CNN model are introduced individually.

Convolution

The initial layers that receive an input signal are called convolution filters. Convolution is a process where the network tries to label the input signal by referring to what it has learned in the past. If the input signal looks like previous cat images it has seen before, the "cat" reference signal will be mixed into, or convolved with, the input signal. The resulting output signal is then passed on to the next layer. A more intuitive schematic diagram of the convolution operation is demonstrated in Figure 2.1.

Convolution has the nice property of being translational invariant. Intuitively, this means that each convolution filter represents a feature of interest (e.g whiskers, fur), and the CNN algorithm learns which features comprise the resulting reference (i.e. cat). The output signal

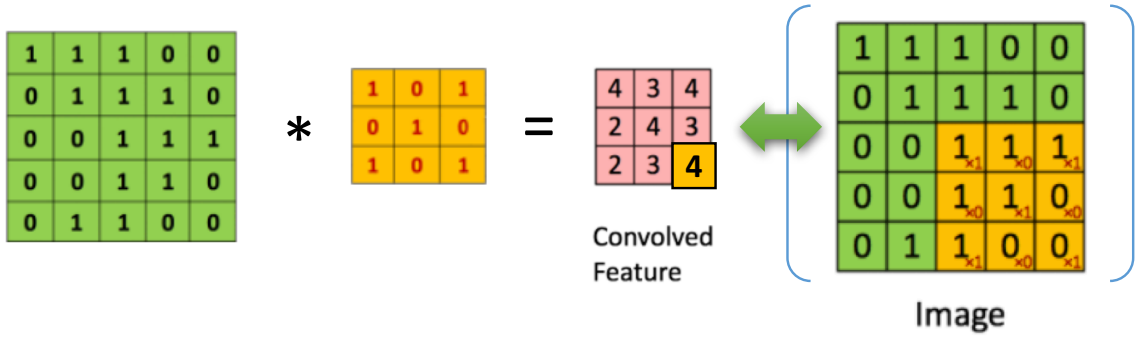


Fig. 2.1 The convolution operation. The filters are convolved over a feature map in a sliding window fashion.

strength is not dependent on where the features are located, but simply whether the features are present. Hence, a cat could be sitting in different positions, and the CNN algorithm would still be able to recognize it. Moreover, we need to specify other important parameters such as the channel depth, stride, and zero-padding. The channel depth corresponds to the number of filters we use for the convolution operation. The more filters we have, the more image features get extracted and the better our network becomes at recognizing patterns in unseen images. Stride is the number of pixels by which we slide our filter matrix over the input matrix. When the stride is 1 then we move the filters one pixel at a time. When the stride is 2, then the filters jump 2 pixels at a time as we slide them around. Having a larger stride will produce smaller feature maps. Sometimes, it is convenient to pad the input matrix with zeros around the border, so that we can apply the filter to bordering elements of our input image matrix. A useful feature of zero padding is that it allows us to control the size of the feature maps.

Non Linearity Activation

The activation layer controls how the signal flows from one layer to the next, emulating how neurons are fired in the network. Output signals which are strongly associated with past references would activate more neurons, enabling signals to be propagated more efficiently for identification. CNN is compatible with a wide variety of complex activation functions to model signal propagation, the most common function being the Rectified Linear Unit (ReLU), which is favored for its faster training speed. Its output is given by:

$$f(x) = max(0,x). \tag{2.1}$$

ReLU is an element-wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero. The purpose of ReLU is to introduce non-linearity in the CNN model since most of the real-world data we would want the network to learn would be non-linear (Convolution is a linear operation - element-wise matrix multiplication and addition, so we account for non-linearity by introducing a non-linear function like ReLU).

Pooling or Sub-sampling

Inputs from the convolutional layer can be “smoothed" to reduce the sensitivity of the filters to noise and translation variations. This smoothing process is called pooling or sub-sampling and can be achieved by taking averages or taking the maximum over a sample of the signal. Such spatial pooling reduces the dimensionality of each feature map but retains the most important information. In case of max pooling shown in Figure 2.2, we define a spatial neighborhood with a 2×2 window and take the largest element from the rectified feature map within that window. Instead of taking the largest element, the average (average pooling) or the sum of all elements can be computed in that window. In practice, max pooling has been shown to obtain better performance.

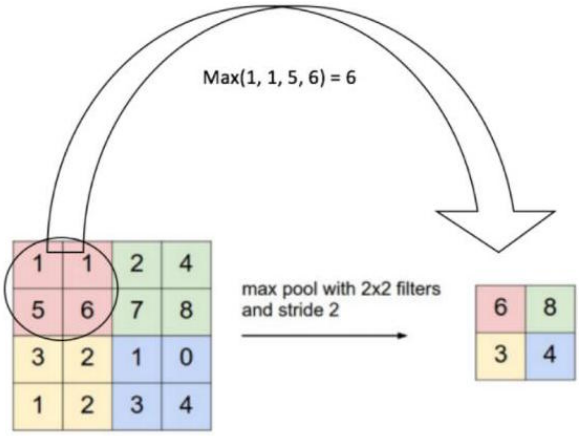


Fig. 2.2 The max pooling operation.

Fully-Connected Layer

The last layers in the network are usually fully-connected, meaning that neurons of preceding layers are connected to every neuron in subsequent layers. The output from the convolutional and pooling layers represent high-level features of the input image. The purpose of the Fully-Connected layer is to use these features for classifying the input image into various

classes based on the training dataset. Apart from classification, adding a fully-connected layer is also a cheap way of learning non-linear combinations of these features. Most of the features from convolutional and pooling layers may be good for the classification task, but combinations of those features might be even better.

Back Propagation

The training process of a CNN model is generally based on the back propagation algorithm [121]. Use back propagation to calculate the gradients of the error with respect to all weights in the network and use gradient descent to update all filter weights and parameter values to minimize the output error. First, the forward propagation can be generalized as that given the activated l-th layer $\mathbf{a}^l$, the $(l+1)$-th layer's activation is computed as:

$$\begin{aligned} \mathbf{z}^{l+1} &= \mathbf{W}^l \mathbf{a}^l + \mathbf{b}^l, \\ \mathbf{a}^{l+1} &= f(\mathbf{z}^{l+1}). \end{aligned} \tag{2.2}$$

The error term for the $(l+1)$-th layer is defined as δ^{l+1}. The loss function is set as $\mathscr{L}(\mathbf{W},\mathbf{b};\mathbf{x},\mathbf{y})$ where $(\mathbf{W},\mathbf{b})$ are the parameters and $(\mathbf{x},\mathbf{y})$ are the training data and label pairs. If the l-th layer is densely connected to the $(l+1)$-th, then the error for the l-th layer is computed as:

$$\delta^l = ((\mathbf{W}^l)^T \delta^{l+1}) \cdot f'(\mathbf{z}^l), \tag{2.3}$$

Where "$\cdot$" denotes the element-wise product operator. Then, the gradients are:

$$\begin{aligned} \nabla_{\mathbf{W}^l} \mathscr{L}(\mathbf{W},\mathbf{b};\mathbf{x},\mathbf{y}) &= \delta^{l+1}(\mathbf{a}^l)^T, \\ \nabla_{\mathbf{b}^l} \mathscr{L}(\mathbf{W},\mathbf{b};\mathbf{x},\mathbf{y}) &= \delta^{l+1}. \end{aligned} \tag{2.4}$$

If the l-th layer is a convolutional and pooling layer then the error is propagated through as:

$$\delta_k^l = \text{upsample}((\mathbf{W}_k^l)^T \delta_k^{l+1}) \cdot f'(\mathbf{z}_k^l), \tag{2.5}$$

Where k indexes the filter number and $f'(\mathbf{z}_k^l)$ is the derivative of the activation function. The upsample operation has to propagate the error through the pooling layer by calculating the error with respect to each unit incoming to the pooling layer. Finally, to calculate the gradient

with respect to the filter maps, we rely on the border handling convolution operation again and flip the error matrix δ_k^l the same way we flip the filters in the convolutional layer.

$$\nabla_{\mathbf{W}_k^l}\mathscr{L}(\mathbf{W},\mathbf{b};\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m}(\mathbf{a}_i^l) * \text{rot90}(\delta_k^{l+1},2), \tag{2.6}$$
$$\nabla_{\mathbf{b}_k^l}\mathscr{L}(\mathbf{W},\mathbf{b};\mathbf{x},\mathbf{y}) = \sum_{c,d}(\delta_k^{l+1})_{c,d}.$$

Where $\mathbf{a}^l$ denotes the l-th layer's input. $(\mathbf{a}_i^l) * \delta_k^{l+1}$ is the convolution between the i-th input in the l-th layer and the error with respect to the k-th kernel filter. m is the number of input channels. $\text{rot90}(A,2)$ is the function to rotate input A by 90×2 degrees in anti-clockwise direction. Moreover, c and d are the width and height of the kernel filter, respectively.

Common Loss Functions

In machine learning, optimization is usually driven by a loss function which specifies the goal of learning by mapping parameter settings to a scalar value specifying the"badness" of these parameter settings. The goal of learning is to find a setting of weights that minimizes the loss functions. Some common loss functions adopted to train deep neural networks are briefly explained as follows, where N and K denotes the number of training samples and classes, respectively.

Euclidean Loss is also known as ℓ_2 loss defined as:

$$\mathscr{E} = \frac{1}{2N}\sum_{n=1}^{N}||\hat{y}^n - y^n||_2^2. \tag{2.7}$$

This can be usually adopted to learn least-square regression tasks.

Information Gain Loss takes an "information gain" matrix specifying the "value" of all label pairs and is defined as:

$$\forall n \sum_{k=1}^{K}\hat{p}_{nk} = 1, \tag{2.8}$$
$$\mathscr{E}(\hat{p}_n) = \frac{-1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K}H_{l_n,k}\log\hat{p}_{n,k}.$$

Where H_{l_n} denotes row l_n of H and $l_n \in [0,1,2,...,K-1]$ indicates the correct class label among the K classes. If H is the identity matrix, this is equivalent to the multinomial logistic loss.

Softmax Loss computes the multinomial logistic loss for a one-of-many classification task, passing real-valued predictions through a softmax to get a probability distribution over classes. Its definition will be further explored in Eq. 3.4.

Dropout Operation

Dropout operation is another important concept used in training a deep model. Dropout refers to ignoring neurons during the training phase of certain set of neurons which is chosen at random. "Ignoring" means these units are not considered during a particular forward or backward pass. At each training stage, individual nodes are either dropped out of the net with probability $1 - p$ or kept with probability p, so that a reduced network is left. Dropout aims to prevent over-fitting, since a fully-connected layer occupies most of the parameters, and hence, neurons develop co-dependency amongst each other during training which curbs the individual power of each neuron leading to over-fitting of training data.

2.1.2 Recurrent Neural Network

Recurrent neural networks (RNNs) [93] have a general architecture illustrated in Figure 2.3. This chain-like nature reveals that RNNs are intimately related to sequences and lists. The natural architecture of neural network is appropriate to use for such data. In the past five years, RNNs have been successfully applied to a variety of problems: action recognition [95, 96], image captioning [150], language modeling [45], etc. In the following sub-sections, the vanilla RNN and its variants are reviewed.

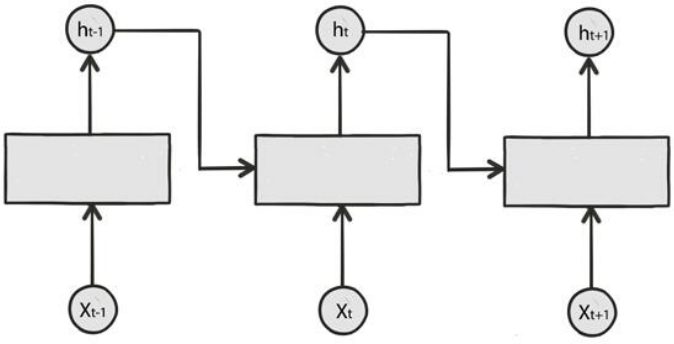


Fig. 2.3 An unrolled recurrent neural network architecture.

Vanilla RNN

For time sequence data, in addition to the current input, a hidden state representing the features in the previous time sequence also needs to be considered. For example, to make a word prediction at time step t in image captioning, both the input $\mathbf{X}_t$ and the hidden state from the previous time step $\mathbf{h}_{t-1}$ are used to compute $\mathbf{h}_t$:

$$\mathbf{h}_t = f(\mathbf{X}_t, \mathbf{h}_{t-1}). \tag{2.9}$$

In RNN, $\mathbf{h}$ serves 2 purposes: the hidden state for the previous sequence data as well as making a prediction. However, for modeling long sequence data, although RNN is theoretically able to learn the long-term dependencies, the performance is unsatisfactory in practice due to the vanishing gradient.

Long Short-Term Memory

Long Short-Term Memory networks (LSTMs) [62] aim to address learning long-term dependencies. In vanilla RNNs, each repeating module has a simple structure such as a single tanh layer. LSTM splits the $\mathbf{h}_t$ in RNN into 2 separate variables $\mathbf{h}_t$ and $\mathbf{C}$. It contains three gates: forget, input, and output, to control what information will pass through:

$$\begin{aligned}
\mathbf{gate}_{forget} &= \sigma(\mathbf{W}_{fx}\mathbf{X}_t + \mathbf{W}_{fh}\mathbf{h}_{t-1} + \mathbf{b}_f), \\
\mathbf{gate}_{input} &= \sigma(\mathbf{W}_{ix}\mathbf{X}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{b}_i), \\
\mathbf{gate}_{output} &= \sigma(\mathbf{W}_{ox}\mathbf{X}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{b}_o).
\end{aligned} \tag{2.10}$$

Where $\mathbf{gate}_{forget}$ controls what part of the previous cell state will be remained, $\mathbf{gate}_{input}$ controls what part of the new computed information will be added to the cell state $\mathbf{C}$, and $\mathbf{gate}_{output}$ controls what part of the cell state will be exposed as the hidden state. In addition, $\sigma(x) = (1+e^{-x})^{-1}$. Then, the cell state and the hidden state will be updated as:

$$\begin{aligned}
\widetilde{\mathbf{C}} &= \tanh(\mathbf{W}_{cx}\mathbf{X}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c), \\
\mathbf{C}_t &= \mathbf{gate}_{forget} \cdot \mathbf{C}_{t-1} + \mathbf{gate}_{input} \cdot \widetilde{\mathbf{C}}, \\
\mathbf{h}_t &= \mathbf{gate}_{output} \cdot \tanh(\mathbf{C}_t).
\end{aligned} \tag{2.11}$$

Where the new state $\mathbf{C}_t$ is formed by forgetting part of the previous cell state while adding part of the new proposal $\widetilde{\mathbf{C}}$ to the cell state. To update $\mathbf{h}_t$, the output gate is used to control

what cell state to export as $\mathbf{h}_t$. $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. An overall diagram of LSTM is shown in Figure 2.4.

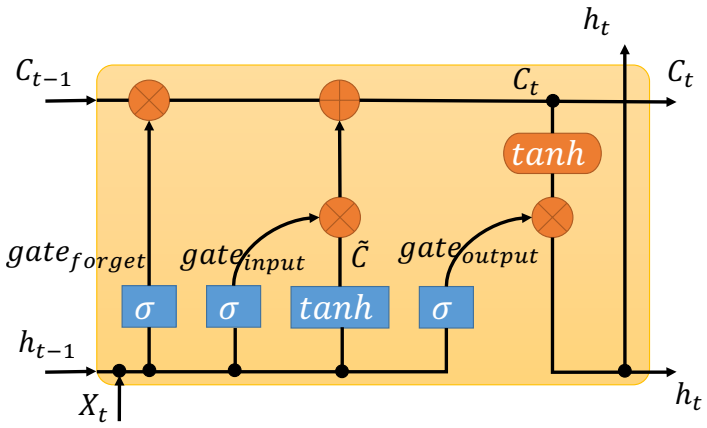


Fig. 2.4 The architecture of Long Short-Term Memory.

Gated Recurrent Unit

Gated Recurrent Unit (GRU) [21] is a variant of the LSTM with a simpler structure, which combines the forget and input gates into a single "update gate", and does not maintain a cell state **C**. Figure 2.5 illustrates its design.

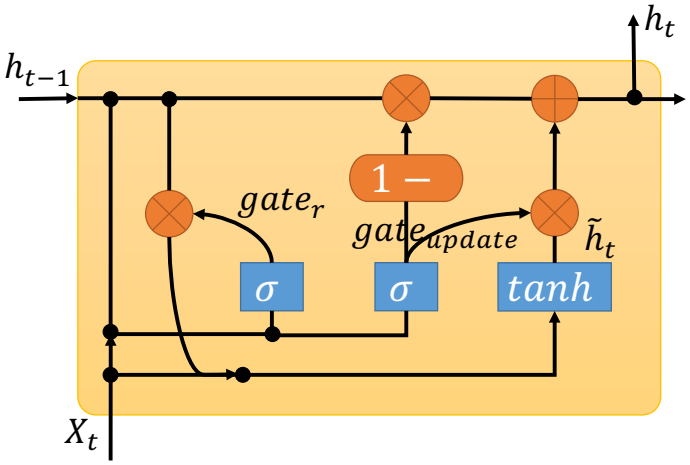


Fig. 2.5 The architecture of Gated Recurrent Unit.

$$\begin{aligned} \mathbf{gate}_r &= \sigma(\mathbf{W}_{rx}\mathbf{X}_t + \mathbf{W}_{rh}\mathbf{h}_{t-1} + \mathbf{b}_r), \\ \mathbf{gate}_{update} &= \sigma(\mathbf{W}_{ux}\mathbf{X}_t + \mathbf{W}_{uh}\mathbf{h}_{t-1} + \mathbf{b}_u). \end{aligned} \tag{2.12}$$

The new hidden state is computed as:

$$\begin{aligned} \widetilde{\mathbf{h}_t} &= \tanh(\mathbf{W}_{hx}\mathbf{X}_t + \mathbf{W}_{hh} \cdot (\mathbf{gate}_r \cdot \mathbf{h}_{t-1}) + \mathbf{b}_h), \\ \mathbf{h}_t &= (1 - \mathbf{gate}_{update}) \cdot \mathbf{h}_{t-1} + \mathbf{gate}_{update} \cdot \widetilde{\mathbf{h}_t}, \end{aligned} \tag{2.13}$$

Where $\mathbf{gate}_r$ is used to control what part of $\mathbf{h}_{t-1}$ to compute a new proposal $\widetilde{\mathbf{h}_t}$. We use $\mathbf{gate}_{update}$ instead of creating a new gate to control what we want to keep from the $\mathbf{h}_{t-1}$.

2.2 Object Detection

2.2.1 Evaluation Metrics

In the computer vision research community, the task of object detection usually refers to predict the bounding-box of the target object. (Image segmentation and contour detection requires more fine-grained contour detection.) Intersection over Union (IoU) is an evaluation metric used to measure the accuracy of an object detector on a particular dataset. We often see this evaluation metric used in object detection challenges such as the popular PASCAL VOC challenge [29] and MSCOCO [92]. More formally, in order to apply IoU to evaluate an object detector, the ground truth bounding-boxes and the predicted bounding-boxes by a model are required. Computing IoU can, therefore, be determined as:

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}. \tag{2.14}$$

Then, an IoU threshold can be specified to determine whether the localization is correct. Hence, each predicted box is either True Positive or False Positive. Each ground truth box is either True Positive or False Negative.

Precision (P) is defined as the number of True Positives (T_p) over the number of True Positives plus the number of False Positives (F_p).

$$P = \frac{T_p}{T_p + F_p}. \tag{2.15}$$

Recall (R) is defined as the number of True Positives (T_p) over the number of True Positives plus the number of False Negatives (F_n).

$$R = \frac{T_p}{T_p + F_n}. \tag{2.16}$$

The precision-recall curve shows the tradeoff between precision and recall for different IoU threshold. A high area under the curve represents both high recall and high precision, where high precision relates to a low false positive rate, and high recall relates to a low false negative rate. Finally, the average precision is computed by averaging the precision values on the precision-recall curve where the recall is in the range [0, 0.1, ..., 1].

2.2.2 General Object Detectors

In this sub-section, we would like to review some classic general object detection approaches. A general comparison of different frameworks is listed in Table 2.1 and more details of pros and cons of each detector are explained in the following paragraphs.

Table 2.1 Comparisons of different general object detection methods.

Methods	Deep / Non-Deep	One / Two-Stage	Region Proposal Mechanism	Main Contribution
DPM [32]	Non-Deep	Two	Sliding Window	Mixtures of multi-scale deformable part models.
R-CNN [39]	Deep	Two	Selective Search	CNN + SVM
SPP-Net [57]	Deep	Two	Selective Search	Spatial pyramid pooling
Fast R-CNN [37]	Deep	Two	Selective Search	RoI pooling + multi-loss
Faster R-CNN [120]	Deep	Two	Region Proposal Network	RPN + Fast R-CNN
R-FCN [23]	Deep	One	Sub-Region	Position-sensitive score maps
SSD [97]	Deep	One	Non-separate design	Single-Shot

DPM [32]

Before the success of deep learning, the deformable part model (DPM) was the most popular object detector, which is proposed to represent highly variable objects using mixtures of multi-scale deformable part models. It aims at detecting objects that vary greatly in appearance and are hard to detect using rigid templates. The main idea of DPM is modeling different parts separately and introducing deformation cost to allow some variations of objects. The practical issues are considered to implement the model, such as the part filters are placed at

twice the spatial resolution of the placement of the root filters. To avoid elaborate labeling multiple parts, DPM treats the part locations as latent variables and uses latent SVM as the classifier. However, DPM is inefficient due to the sliding window fashion.

R-CNN [39]

Regions with CNN features are proposed by Girshick *et al.* and made a considerable improvement of mean average precision (mAP) on the PASCAL VOC dataset. At the region proposals module, it uses selective search [138] to extract around 2000 category-independent region proposals. Then, R-CNN uses AlexNet to extract higher layer feature vectors such as FC-7 with 4096-dimensional feature vector. Finally, a multiple class SVM classifier is applied to evaluate the region proposal. A threshold of the score is learned to reject region proposals with low IoU overlap. Several region proposals recognized as positive might overlap and bound the same object, non-maximum suppression is used to merge them.

SPP-Net [57]

R-CNN does improve the detection precision, but it has several notable drawbacks including training with multi-stage pipeline and feature caching which is expensive in time and space. Spatial pyramid pooling network (SPP-Net) proposed to speed up R-CNN by sharing computation. SPP-Net is implemented via spatial pyramid pooling layer, but the back propagation through the SPP layer is highly inefficient when each training sample comes from a different image.

Fast R-CNN [37]

Fast R-CNN uses the idea of feature maps from SPP-Net, and integrates the training into a single-stage using a multi-task loss. Since it is single-stage training, no extra disk space for feature caching is required. A Fast R-CNN network has two sibling outputs, a discrete probability distribution (per Region of Interest) and bounding-box regression offsets for each object class. A joint loss is used to improve these two tasks performance together.

Faster R-CNN [120]

Faster R-CNN attempts to improve the region proposals generation by introducing a Region Proposal Network (RPN) which simultaneously predicts object bounds and objectness scores at each position. Once the region proposals are obtained, we feed them straight into what is essentially a Fast R-CNN. They add a pooling layer, some fully-connected layers, and finally

a softmax classification layer and bounding box regressor. In a sense, Faster R-CNN = RPN + Fast R-CNN.

R-FCN [23]

Region-based Fully Convolutional Net (R-FCN), shares 100% of the computations across every single output. The essential part of R-FCN is position-sensitive score maps. Each position-sensitive score map represents one relative position of one object class. In simple words, R-FCN considers each region proposal, divides it up into sub-regions, and iterates over the sub-regions asking: “does this look like the top-left of a baby?", “does this look like the top-center of a baby?", “does this look like the top-right of a baby?", etc. It repeats this for all possible classes. If enough of the sub-regions say“yes, I match up with that part of a baby!", the RoI gets classified as a baby after a softmax over all the classes.

SSD [97]

SSD stands for Single-Shot Detector, which provides enormous speed gains over Faster R-CNN. SSD performs the region proposal network and either fully-connected layers or position-sensitive convolutional layers in a “single shot", simultaneously predicting the bounding box and the class as it processes the image. It classifies and draws bounding-boxes from every single position in the image, using multiple different shapes, at several different scales.

2.2.3 Vehicle Detectors

Vehicle is one of most important targets in the object detection tasks. In addition to general object detectors, there are also certain vehicle-specific detection methods. Sun *et al.* [133] proposed to deploy HOG and Gabor features with SVM and neural network classification. The experiments are evaluated on static images. Zhu *et al.* [175] designed dynamic background modeling of overtake area, and performed validation on real-world video with ego-motion compensation. Moreover, Sivaraman and Trivedi [129] used Haar-like features with Adaboost classification and active learning algorithms to address vehicle detection. Active learning has been shown to improve the detection and false alarm rates. Jazayeri *et al.* [65] introduced a motion-based method which combines optical flow and hidden Markov model classification. The approach models the position and motion of preceding vehicles in the image plane.

2.3 Object Re-identification

2.3.1 Evaluation Metrics

In the task of re-identification (re-ID), the target objects' images are usually cropped and aligned. The re-ID problem is similar to the instance retrieval task. Given one query image, the candidates with the same identity in the gallery set are desired to be placed in the top positions within a ranking list. To evaluate re-ID approaches, the cumulative matching characteristics (CMC) curve [144] is usually adopted by most of the researchers. In a CMC curve, the cumulative number of correctly matched queries is shown based on the ranking list in which they have re-identified. When the number of true re-identified queries in rank i is $q(i)$, the CMC value for rank i can be defined as:

$$CMC(i) = \sum_{r=1}^{i} q(r). \tag{2.17}$$

Where r is the rank index. The advantage of CMC evaluation metric is that not only the rank-1 is computed, but also the correctly matched queries in other top ranks can be indicated as well. Thus, CMC can better describe the re-ID performance of various methods.

Apart from CMC curves, if multiple ground truths for each query in the gallery set are available, mean average precision (mAP) [99] can be deployed to trade off the precision and recall to evaluate the overall performance for re-ID. Given an query image, the average precision (AP) can be defined as:

$$AP = \frac{\sum_{k=1}^{n} P(k) \times \mathrm{G}(k)}{N_{gt}}, \tag{2.18}$$

Where N_{gt} denotes the number of ground truths, and n is the number of test tracks. $P(k)$ indicates the precision at cut-off k in the ranking lists. $\mathrm{G}(k)$ equals 1 when the k-th match is true, otherwise 0. Thus, the mAP can be calculated over all queries as:

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q}, \tag{2.19}$$

Where Q denotes the number of queries.

2.3.2 Person Re-identification

Since most of the works have only explored the single image-based person re-ID problem, we mainly review such algorithms. To date, image-based methods can be coarsely categorized into three groups: image description, distance metric learning, and deeply-learned models.

Feature Representation

Symmetry-Driven Accumulation of Local Features (SDALF) is proposed in [31] to extract three kinds of hand-crafted features on the symmetry-based silhouette. It includes the weighted color histograms, maximally stable color regions, and recurrent high-structured patches. The final feature matching is a weighted combination of distances computed by three feature representations. Gray and Tao [47] first partitioned a human image into horizontal stripes and adopted 8 color channels and 21 texture filters as local feature representations. A number of later works [168, 163, 15] introduced many enhanced hand-crafted features but did not obtain large improvement. More recently, Liao *et al.* [88] designed the local maximal occurrence (LOMO) representation which consists of the color and Scale Invariant Local Ternary Pattern (SILTP [91]) histograms. The LOMO feature analyzes the horizontal occurrence of local features and performs maximization among sub-windows. LOMO feature has been successfully adopted in [159, 160] as the feature extraction part.

In addition to above low-level texture and color features, attribute-based mid-level representations have also been widely introduced in many state-of-the-art approaches. Layne *et al.* [81] specified 15 semantic attributes such as shorts, sandals, and backpack, and adopted low-level features to train attribute classifiers. Moreover, Zhao *et al.* [164] proposed a method of learning mid-level filters to discover patch clusters. Such mid-level filters are discriminatively trained for identifying different visual patterns and distinguishing persons. To some extent, the method is invariant to cross-view variations.

Distance Metric Learning

Another significant research branch of person re-ID focus on distance metric learning whose basic idea is to put samples of the same identity together while pushing samples of different identities away. A general formulation is based on Mahalanobis distance function, which extends linear scalings and rotations of the space compared with normal Euclidean distance. The equation is defined as:

$$d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j), \tag{2.20}$$

Where $\mathbf{M}$ is a positive semi-definite matrix. $(\mathbf{x}_i, \mathbf{x}_j)$ are pairwise data inputs. Large Margin Nearest Neighbor (LMNN) [148] adopts an idea that a combination of allowance of slack for different classes (like SVMs) with the local region of a k-nearest neighbor sphere. The formulation tries to increase similarity to target neighbors while reducing it to impostors (other classes) lying within this k-nearest neighbor region. Information Theoretic Metric Learning (ITML) [25] formulates the problem as the minimization of differential relative entropy between two multi-variate Gaussian under constraints of the distance function. The trade-off lies in satisfying constraints while being close to a prior (usually identity - leading to Euclidean distance). The constraints are to keep similar pairs below some distance d_{sim} and dissimilar pairs above d_{diff}. Moreover, Logistic Discriminant Metric Learning (LDML) is introduced in [50]. The probability of a pair being similar is modeled by a sigmoid function which includes a bias term, and the Mahalanobis distance, defined as $p_{ij} = \sigma(b - d^2_{\mathbf{M}}(\mathbf{x}_i, \mathbf{x}_j))$. The metric is then estimated by gradient descent. KISSME [74] is a non-iterative and extremely fast method to learn a metric. It assumes that the feature space of both hypothesis (pair is similar or dissimilar) is Gaussian, and then formulates the metric as a difference of (inverted) covariance matrices, defined as $\hat{\mathbf{M}} = \Sigma^{-1}_{y_{ij}=1} - \Sigma^{-1}_{y_{ij}=0}$, where $y_{ij} = 1$ if a pair(i, j) is similar and 0 otherwise. Specifically, $\Sigma_{y_{ij}=1} = \sum_{y_{ij}=1}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$ and $\Sigma_{y_{ij}=0} = \sum_{y_{ij}=0}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$. They clip the spectrum of $\hat{\mathbf{M}}$ by eigenanalysis to obtain $\mathbf{M}$. In their experiments, the assumption is satisfied by reducing the dimension of the features via PCA.

Aside from learning distance metrics, some researchers investigated subspace learning to address person re-ID. Liao *et al.* [88] proposed the cross-view quadratic discriminant model to learn the projection w to a low dimensional subspace as:

$$\mathscr{J}((\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}, \tag{2.21}$$

Where $\mathbf{S}_b$ and $\mathbf{S}_w$ indicates the inter-class and intra-class scatter matrices, respectively. Furthermore, Zhang *et al.* [159] employed the null Foley-Sammon transform to optimize a discriminative null space minimising the intra-class scatter and maximizing the inter-class scatter.

Deep Learning Models

Recently, deep learning models have largely benefited person re-ID, which hugely outperform previous methods. The first work using a deep network for re-ID is [156]. It partitions the human body into three components and adopts a Siamese deep neural network to learn the

similarity metric. Li *et al.* [85] presented a filter pairing neural network to jointly process feature learning, patch matching, photometric transforms, part displacement, and viewpoint transforms. To better exploit intermediate layers of a deep model, Ahmed *et al.* [1] embedded a module of learning cross-input neighborhood differences into the Siamese network, and compares the feature patterns from one input image with those in neighboring locations of the other image. Moreover, compared with pairwise image inputs, Cheng *et al.* [20] proposed a multi-channel parts-based CNN within a triplet framework. After the first convolutional layer, four parallel sub-networks are set for different body parts and then fused into a global fully-connected layer.

A Siamese or triplet neural network for person re-ID has a shortcoming that the identity labels have not been considered. Methods in this pipeline can only discriminate whether an input image pair is similar or dissimilar. Thus, another branch of works exploits classification using identity labels to improve the performance. Xiao *et al.* [149] used multiple datasets to jointly train their model by a softmax classification and presented a domain guided dropout approach to help feature learning. When training a CNN model with samples from different domains, neurons can learn representations either shared across different domains or a specific one. Furthermore, when the scale of the dataset becomes larger, methods [166, 167] adopting the basic classification unit can achieve better performance even without data selection.

2.3.3 Vehicle Re-identification

Inspired by person re-ID, vehicle re-ID has attracted more attention in the past two years. Deep relative distance learning [94] is designed for learning the difference between similar vehicles based on a new VehicleID dataset which contains two viewpoints: front and rear. The triplet loss is extended to the coupled cluster loss that the loss function is defined over multiple samples instead of three. The mixed difference network structure is proposed to construct an end-to-end model. In [158], a vehicle is first extracted by a 3D bounding-box and then warped into a plane as a normalized image. HOG and color histogram features are computed and linear SVM is learned to match the vehicles. Liu *et al.* [98] released the VeRi-776 dataset where vehicles have more available viewpoints and proposed the FACT feature which consists of the color name, SIFT, and GoogLeNet features. FACT can well describe a vehicle's color, texture, shape, and semantic attributes. Moreover, Wang *et al.* [145] presented orientation invariant feature embedding to address vehicle re-ID and adopted spatial-temporal regularization to refine the results.

Some researchers prefer to adopt license plate and spatial-temporal information to achieve highly accurate performance. Liu *et al.* [99] presented the first work to employ visual feature, license plate and spatial-temporal relation to explore the re-ID task. The authors assume that

two vehicle images are highly possible to have the same identity if they have small space or time interval. Shen *et al.* [126] designed a chain MRF model for visual-spatio-temporal path proposal and employed deep neural networks as pairwise potential function. Moreover, along the path proposal, a Path-LSTM takes visual and spatio-temporal differences of neighboring points as inputs to predict whether the path is valid or not.

2.4 Generative Adversarial Networks

Generative Adversarial Net (GAN) is an unsupervised machine learning method, which achieves great success in image generation tasks. It consists of a generative model G and a discriminative model D competing against each other in a two-player min-max game. The generative network takes a latent random vector z from a uniform distribution as input to generate samples. The $p_z(z)$ is expected to converge to a target true data distribution $p_{data}(x)$, where x is a real image. Meanwhile, the discriminative network aims to distinguish the real data from synthesized samples. These two networks are simultaneously optimized by the following equation:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x\sim p_z(z)}[\log(1-D(G(z)))]. \tag{2.22}$$

It has been proved [44] that a global optimum can be obtained when p_G well converges to p_{data}, if G and D have enough capacity. Compared to other generative models, GAN has few restrictions (e.g. no Markov chain and variational bound is needed relative to Boltzmann machines and VAEs). Moreover, since G is very poor in the early training stage and D can easily reject synthesized samples with high confidence, $\log D(G(z))$ is maximized for better training G rather than minimizing $\log(1-D(G(z)))$.

The basic diagram of GAN is illustrated in Figure 2.6. Given enough model capacity and training time, the Algorithm 2.1 is desired to converge to a good generator of p_{data}.

2.5 Visual Attention Learning

A recent trend in deep learning is attention mechanisms. Attention models have been widely explored in long short-term memory recurrent neural networks. The encoder-decoder architecture is the most popular one and has demonstrated state-of-the-art results across a range of domains such as text translation, image captioning and visual question answering.

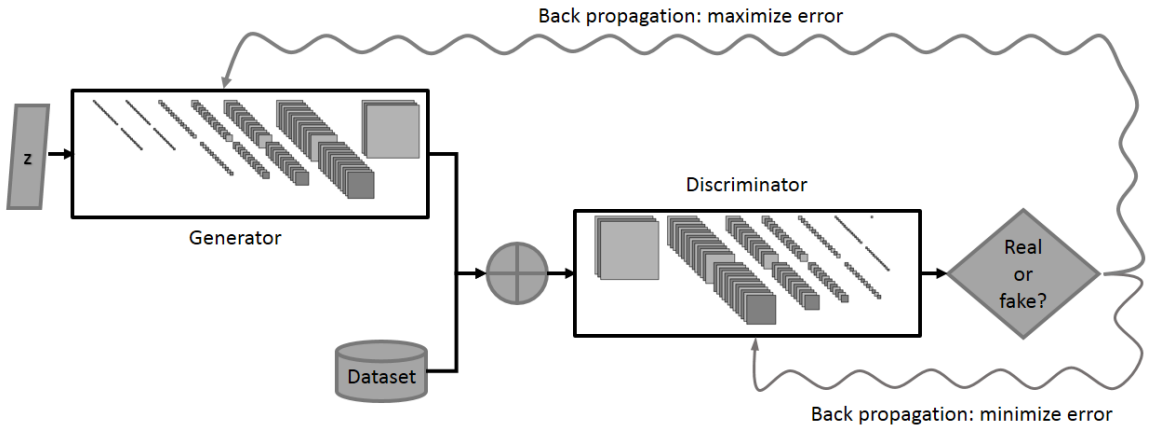


Fig. 2.6 GAN mainly consists of a Generator and a Discriminator. The Generator network takes a random input and tries to generate a sample of data. The task of Discriminator network is to take input either from the real data or from the generator and try to predict whether the input is real or generated.

Algorithm 2.1 Batch SGD training of GAN. The hyper-parameter k is usually to adjust the number of steps of discriminator in the training phase.

Inputs: Real image data X.
Outputs: learned parameters for Generator G and Discriminator D.
for number of training iterations **do**
  for k steps **do**
    Sample a batch of n random noise samples $\{z^1, z^2, ..., z^n\}$ from $p_g(z)$.
    Sample a batch of n samples $\{x^1, x^2, ..., x^n\}$ from real data distribution $p_{data}(x)$
    Update the D by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} [\log D(x^{(i)}) + \log(1 - D(G(z^{(i)})))].$$

  end for
  Sample a batch of n random noise samples $\{z^1, z^2, ..., z^n\}$ from $p_g(z)$.
  Update the G by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)}))).$$

end for

In terms of the attention within sequences [5], it is the idea of freeing the encoder-decoder architecture from the fixed-length internal representation. This is achieved by keeping the intermediate outputs from the encoder LSTM from each step of the input sequence and training the model to learn to pay selective attention to these inputs and relate them to items in the output sequence. Put another way, each item in the output sequence is conditional on selective items in the input sequence. This increases the computational burden of the model but results in a more targeted and better-performing model. Moreover, the model is also able to show how attention is paid to the input sequence when predicting the output sequence. This can help in understanding and diagnosing exactly what the model is considering and to what degree for specific input-output pairs.

Convolutional neural networks applied to computer vision problems also suffer from similar limitations, where it can be difficult to learn models on very large images. Visual attention mechanisms [49, 108, 4] aim to automatically focus on the core regions of image inputs and ignore the useless parts. For instance, in terms of image recognition, only the regions of certain objects or patterns are useful for prediction while most of the backgrounds are less effective. In this section, we mainly review the attention models proposed for vision tasks.

Different from the glimpse approach, the sequence-based attentional mechanism can be applied to computer vision problems to help get an idea of how to best use the convolutional neural network to pay attention to images when outputting a sequence, such as a caption. Xu *et al.* [150] introduced two attention models to generate image descriptions. One is a soft attention mechanism trainable by back propagation algorithms, while the other is a hard stochastic attention model learned by reinforcement learning. The caption is generated by an RNN with LSTM units which takes previous words, previous state, and a context vector in order to generate the next word in the caption. The way the context vector is created depends on the type of attention mechanism. Hard attention is based on stochastic decisions which are discrete so derivatives are 0. The loss is variational bound on the marginal log-likelihood. Soft attention is a deterministic approach taking into account all the L annotation vectors (2d vectors from the CNN) and weights them depending on current time i and the vector j of L total vectors.

Another representative attention-based work is presented in [108]. A novel RNN model is capable of extracting information from an image or video by adaptively selecting a sequence of regions or locations and only processing the selected regions at high resolution. Since the region selection stage is non-differentiable, reinforcement learning is used. The model is an RNN structure that processes inputs sequentially, attending to different locations within the image (or video frames) one at a time, and incrementally combines information from these

fixations to build up a dynamic internal representation of the scene or environment. One advantage of this approach is that the computation is independent of the size of the image. Only several local regions are used to model each image. In each time-step of the RNN, a glimpse sensor is used to extract a representation for a given coordinate. This representation is combined with the coordinates information to generate the glimpse representation with a Glimpse Network. The information to be passed along time is through the internal representation in RNN. And the output of RNN is used to predict the next location, and the action. The action can be in various forms. In a recognition task, for example, it could be a softmax layer.

In the past three years, there have been many other attention mechanisms successfully proposed in different tasks. A stacked attention network [155] is designed to learn to answer natural language questions from images, usually known as visual question answering (VQA). Haque *et al.* [56] proposed an attention mechanism that reasons on human body shape and motion dynamics to identify person ID without RGB data. Moreover, multi-scale features at each pixel location are softly weighted by an attention model presented in [16] for better addressing semantic image segmentation. To recognize fine-grained categories, a novel recurrent CNN is adopted in [36] to recursively learn multi-scale discriminative region attention and region-based feature representation.

2.6 Datasets and Evaluation Protocols

The Comprehensive Cars (CompCars) Dataset

The CompCars dataset [153] is proposed for learning vehicles' fine-grained level tasks such as attribute prediction and verification, since it has abundant attributes' annotation. The CompCars dataset contains data from two scenarios, including images from web-nature and surveillance-nature. The web-nature data contains 163 car makes with 1,716 car models. There are a total of 136,726 images capturing the entire cars and 27,618 images capturing the car parts. The full car images are labeled with bounding boxes and viewpoints. Each car model is labeled with five attributes, including maximum speed, displacement, number of doors, number of seats, and type of car. The surveillance-nature data contains 50,000 car images captured in the front view. In our experiments, we mainly adopt the web-nature data where some examples are shown in Figure 2.7 for learning attributes.

Fig. 2.7 Example images of the web-nature data from the CompCars dataset.

PASCAL VOC 2007 Car Dataset

The PASCAL Visual Object Classes (VOC) Challenge 2007 dataset [29] is proposed to for recognizing objects from a number of visual object classes in realistic scenes (i.e. not pre-segmented objects). There are twenty object classes such as person and car. For the main tasks - classification and detection, three sets of images are provided, including the training, validation and test sets. In this thesis, all the images containing cars in the trainval and test sets (totally 1434 images) are selected to be evaluated.

LISA Vehicle Detection Dataset

LISA Vehicle Detection dataset [129] contains three color video sequences captured at different times of the day and illumination settings: morning, evening, sunny, cloudy, etc. The main driving environments are highway and urban. Moreover, the traffic conditions vary from light to dense.

PKU VehicleID Dataset

VehicleID dataset [94] is a large-scale surveillance dataset which can be used for studying the re-ID problem. It consists of the training set with 110,178 images of 13,134 vehicles and the test set with 111,585 images of 13,133 vehicles. However, the dataset only includes

Fig. 2.8 The upper images are examples from the PASCAl VOC 2007 Car dataset, while the bottom ones are from the LISA Vehicle Detection dataset.

two viewpoints: front and rear. Moreover, 250 most commonly appeared vehicle models are labeled. Some example images of different vehicle models are demonstrated in Figure 2.9.

VeRi-776 Dataset

The VeRi dataset [99] contains 776 different vehicles captured in 20 cameras along a circular road within a city area (see Figure 2.10). The whole dataset is split into 576 vehicles with 37,778 images for training and 200 vehicles with 11,579 images for testing. The images are captured in a real-world unconstrained surveillance scene and labeled with varied attributes such as types, colors, and brands. So complicated models can be learned and evaluated for vehicle re-ID.

Fig. 2.9 Example images of different vehicle models from the VehicleID dataset.

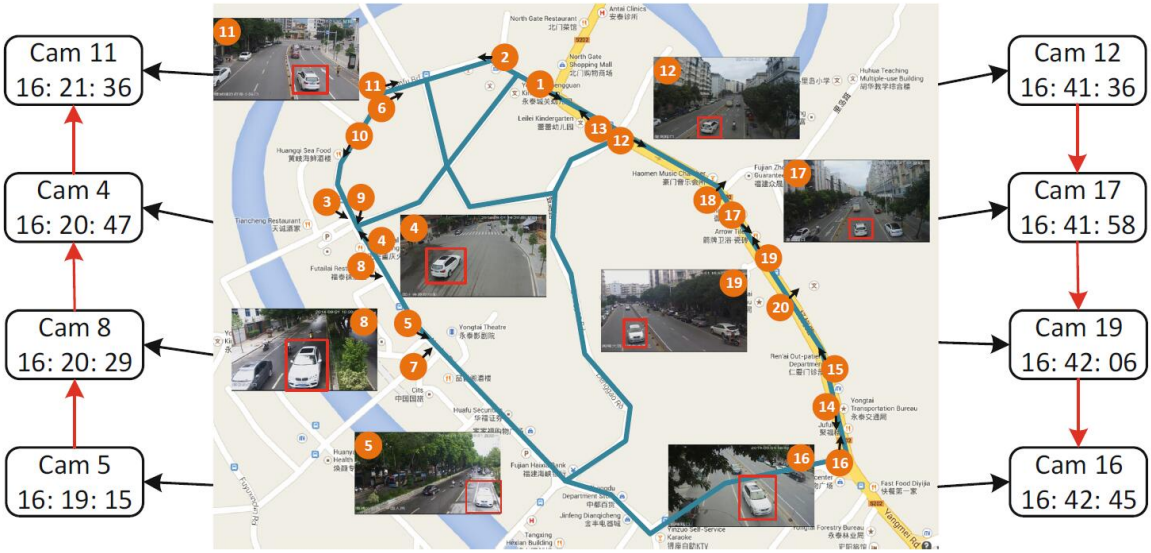


Fig. 2.10 A glance of the VeRi-776 dataset. Each vehicle is captured by more than 2 cameras in a city area.

Chapter 3

Deep Neural Networks for Fast Vehicle Detection and Multi-task Learning

Object detection is one of the most fundamental problems in computer vision. It aims to localize bounding-boxes of target objects in an image or video and recognize the objects' semantic classes. Previous detectors can be coarsely categorized into image-based and video-based approaches. Video-based methods such as frame difference and Gaussian mixture model can only detect objects with motion in videos but have less potential in applications. In this chapter, we mainly discuss the image-based object detectors particularly targeting at vehicles. Object detection is usually regarded as the most difficult task in computer vision due to many challenges including objects' viewpoint variation, illumination change, occlusion, multi-scale, objects' deformation and background clutter. Traditional object detection frameworks employ sliding window fashions which are inefficient. Recently, deep learning based methods have achieved great success on object detection by extending image classification models. They can be grouped into two-stage and one-stage frameworks. The two-stage framework usually provides object proposals first and then classifies each proposal to refine the results. Methods under this framework can achieve state-of-the-art performance but are not efficient. On the other side, the one-stage detectors are performed over dense sampling of different object scales, locations and object aspect ratios. The one-stage framework aims to propose end-to-end models which can detect objects in real time but sacrifice accuracies to some extent. Therefore, we propose a high-performance deep neural network for vehicle detection which aims to improve both the accuracy and speed.

Multi-task learning is a subarea of machine learning in which multiple tasks can be optimized simultaneously. Compared with learning the individual task-specific model, multi-task learning can explore the commonalities and discrepancies across different tasks to improve the performance and the efficiency. Thus, classification, regression, and cross-

entropy loss are combined to optimize our network. In addition to object detection, people are usually interested in the more fine-grained attributes recognition as well. For example, given detected vehicles, the corresponding attributes learning problems such as viewpoint estimation, type classification, and color recognition, can be jointly embedded into a multi-task model.

3.1 Introduction and Motivation

Intelligent traffic surveillance is being widely explored since the number of vehicles is ever-increasing and large-scale streaming video data become available. Compared with sensor-based traffic surveillance techniques, computer vision has attracted a great deal of attention and made significant contributions to practical applications such as vehicle counting, retrieval, and behavior analysis due to its low cost. Among many research areas, vehicle detection is the most fundamental problem on which other tasks such as vehicle classification and identification need to rely. Since most vision-based detection works are usually applied to general objects, we present object detection frameworks first and then focus on vehicles.

With the increase in the number of image data, fast and accurate object detection methods are rising in demand. These methods involve not only recognizing and classifying every object in an image but also localizing each one by drawing the appropriate bounding box around it. This makes object detection a significantly harder task than its traditional computer vision predecessor, image classification. In industries, object detection has the huge potential in practical applications such as video surveillance, autonomous driving and robot vision. It is usually regarded as the core competencies of vision-based artificial intelligence. However, detecting target objects in natural images faces many challenges leading to the low accuracy and efficiency. First, visual patterns of most objects of interest such as vehicles and pedestrians usually change hugely with viewpoint variations. It makes detectors become unstable for objects appearing in arbitrary viewpoints. Large illumination will change the RGB values of the same object. Severe occlusion will make some parts of an object invisible, which fools the trained kernels. Moreover, most object detection methods can only deal with objects within the limited range of scale and the multi-scale processing usually make the methods inefficient. Therefore, object detection has attracted large attention in the past decades.

Object detection has been explored with a long history but still not achieved the satisfied performance of speed and accuracy trade-offs. Traditional object detectors used to deploy the sliding window framework as shown in Figure 3.1. Take one-class object detection as an example, positive and negative samples are first selected with moderate data pre-processing. Then, conventional hand-crafted features of training data are extracted for learning a binary

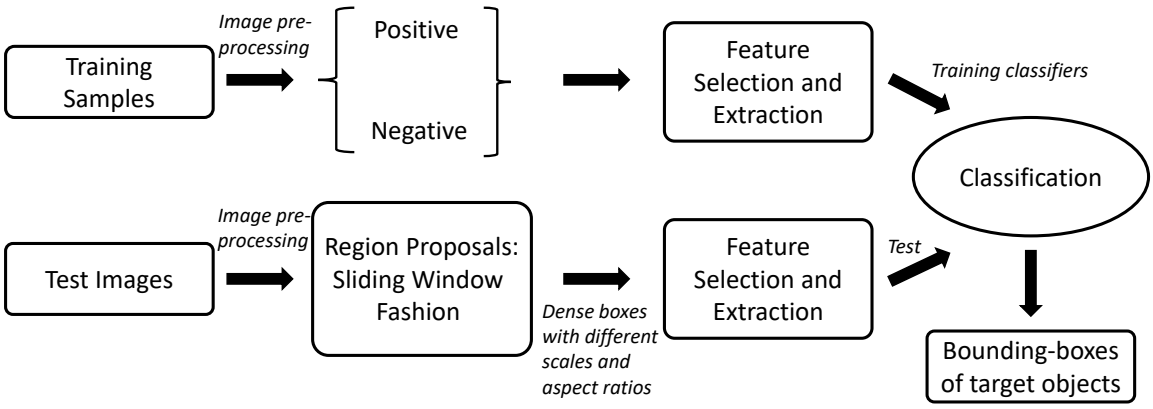


Fig. 3.1 A general framework for conventional object detection methods.

classifier. Once the model training is completed, given an upcoming test image, a sliding window is performed on the test frame with different scales and aspect ratios to get region proposals, which is extremely inefficient. Each proposed region is classified by the trained model to output the final bounding-boxes. Two representative frameworks are the Viola-Jones detector [141] and the DPM detector [32]. However, such detection frameworks usually lead to many overlapped bounding-boxes and an unsatisfactory precision and recall rate.

It's not news that deep learning has been a real game-changer in machine learning, especially in computer vision. In a similar way that deep learning models have crushed other classic models on the task of image classification, deep learning models are now state-of-the-art in object detection as well. Deep object detectors can be mainly categorized into two groups: the two-stage detector and the one-stage detector. A brief comparison of these two architectures is illustrated in Figure 3.2. The two-stage framework usually consists of an object proposal network to propose potential locations and a classification network to finalize the detection results. The well-known OverFeat [125], R-CNN [39], Fast R-CNN [37] and Faster R-CNN [120] are all the two-stage detectors which can achieve good accuracies on the detection datasets. On the other side, the one-stage framework prefers to adopt one end-to-end deep model to finish detection by the dense sampling of possible object regions. The recent successful one-stage detectors are YOLO [118], SSD [97] and R-FCN [23]. In general, the two-stage detectors aim to obtain top accuracies but the one-stage detectors prefer promising results with faster processing speed.

Vehicle is the most interested object in the video surveillance for urban traffic. Vehicle detection is a fundamental objective of vehicle-related researches. Traditional vehicle detection methods can be categorized into frame-based and motion-based approaches [130, 9]. For motion-based approaches, frames subtraction [114], adaptive background modeling [131] and optical flow [105] are often utilized. However, some non-vehicle moving objects will

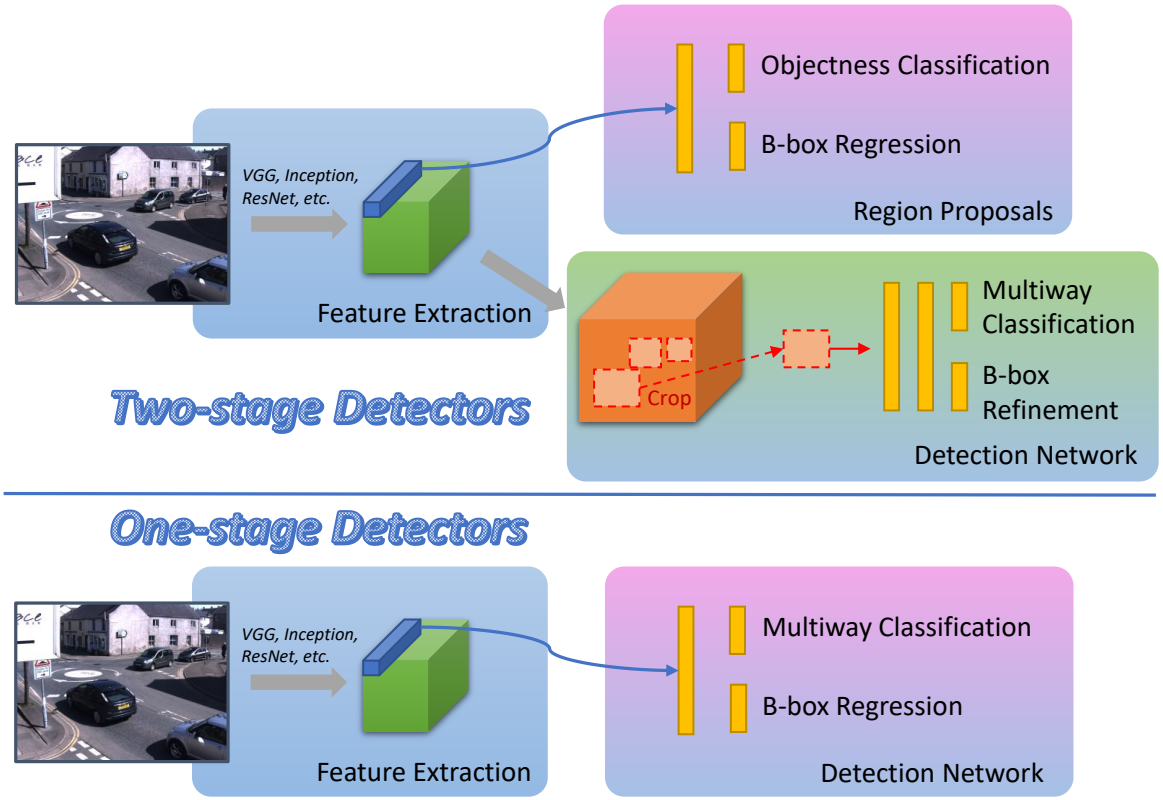


Fig. 3.2 Deep learning models for object detection, which can be categorized into two-stage and one-stage frameworks.

be falsely detected with motion-based approaches since less visual information is exploited. To achieve higher detection performance, recently, the deformable part-based model (DPM) [33] employs a star-structured architecture consisting of root and parts filters with associated deformation models for object detection. DPM can successfully handle deformable object detection even when the target is partially occluded. However, it leads to heavy computational costs due to the use of the sliding window procedure for appearance features extraction and classification.

In urban traffic surveillance, another interesting and valuable task is to extract more diverse information from detected vehicles - we call it vehicle attributes annotation. Each individual vehicle on the road has its special attributes: travel direction (i.e., pose), inherent color, type and other more fine-grained information with respect to the headlight, grille, and wheel. It is extremely beneficial to annotate a target vehicle's attributes accurately. Lin et al. [152] presents an auto-masking neural network for vehicle detection and pose estimation. In [86], an approach by vector matching of templates is introduced for vehicle color recognition. In [27], an unsupervised convolutional neural network is adopted for

vehicle type classification from frontal view images. However, independent analysis of different attributes makes the visual information not well explored and the process inefficient, and little work has been done for annotating these vehicle attributes simultaneously. Actually, there exist strong correlations between these vehicle attributes learning tasks. For example, vehicle type classification based on visual structures is very dependent on the viewpoint. Therefore, we believe multi-task learning can be helpful since such joint learning schemes can implicitly learn the common features shared by these correlated tasks. Moreover, a unified multi-attributes inference model can also significantly improve the efficiency.

In this chapter, we mainly present a fast framework of Detection and Annotation for Vehicles (DAVE), which effectively combines vehicle detection and multi-attributes annotation. DAVE consists of two CNNs: fast vehicle proposal network (FVPN) and attributes learning network (ALN). The FVPN is a shallow *fully convolutional network* which is able to predict all the bounding-boxes of vehicle-like objects in real-time. The latter ALN configured with a very deep structure can precisely verify each proposal and infer pose, color and type information for positive vehicles, simultaneously. It is noteworthy that informative features learned from the deep ALN can be regarded as latent data-driven knowledge to guide the training of the shallow FVPN, thus we bridge the ALN and FVPN with such knowledge guidance and jointly optimize these two CNNs at the same time in our architecture. In this way, more exhaustive vehicle descriptions learned from the ALN as helpful supervision benefits the FVPN with better performance. Once the joint training is completed, a two-stage inference scheme will be adopted for final vehicle annotation. The main works for vehicle detection and multi-task learning are highlighted as follows:

1. To have a preliminary understanding on deep neural network, a baseline is designed to investigate the Deep Belief Network (DBN) for vehicle detection. We adopt multi-level complex wavelet features (CWF) as the vehicle descriptor and train a DBN which consists of a number of layers of Restricted Boltzmann Machines (RBM) to implement vehicle detection.

2. Multiple vehicle-related tasks are unified into one deep vehicle annotation framework DAVE which can effectively and efficiently annotate each vehicle's bounding-box, pose, color and type simultaneously. Two CNNs proposed in our method, i.e., the Fast Vehicle Proposal Network (FVPN) and the vehicle Attributes Learning Network (ALN), are optimized in a joint manner by bridging two CNNs with latent data-driven knowledge guidance. In this way, the deeper ALN can benefit the performance of the shallow FVPN. We also introduce a new Urban Traffic Surveillance (UTS) vehicle dataset consisting of six 1920×1080 (FHD) resolution videos with varying illumination conditions and viewpoints to evaluate our proposed model.

3.2 Related Work

Vehicle-related systems usually adopt sensor-based approaches which are robust against illumination and viewpoint variations, to efficiently and stably detect and annotate vehicles. Sonar sensors [71] are configured in the front and rear of vehicles to help detection. Wireless magneto-resistive sensors [146] are adopted to test whether there are vehicles passing by. Strain gauge sensors are introduced in [127] to automatically classify vehicle types, and dead-reckoning/GPS sensors are exploited to estimate the pose of a driverless vehicle in [104].

Compared to the high costs of industrial grade sensors, computer vision methods only require low-cost cameras and attract increasingly more interests in intelligent surveillance applications in past decades. Most traditional vision-based vehicle detection methods follow the sliding window fashion that is composed of appearance features extraction and classification. For instance, Histogram of Oriented Gradients (HOG) [24] and Haar-like features [55] are usually extracted, and SVMs [10], and AdaBoost [34] are adopted to discriminate whether each window with different scales and aspect ratios is a positive vehicle. The deformable part-based model (DPM) [33] successfully handles the detection of deformable objects but is not efficient due to the sliding window framework.

In recent years, with the great success of deep learning methods on image classification [76], Girshick et al. [38] proposed Region-based CNN which combines object proposal [138, 176], CNN learned features and SVM classifiers to perform detection. For increasing the detection speed and accuracy, Fast RCNN [37] adopts a region of interest (ROI) pooling layer and multi-task loss to estimate object classes while predicting bounding-box positions. Furthermore, Faster RCNN [120] employs initial layers with shared convolutional features to enable cost-free effective proposals. However, deep models deployed by these methods are designed for general object detection. Our work advances the idea of detection by focusing on one specific object: private motor vehicles.

Previous automatic vehicle annotation methods only focused on some single-purpose tasks such as color recognition [154] and coarse vehicle type classification [19]. Little work has been conducted for annotating different vehicle attributes simultaneously including pose, color and type information. We mainly review separate related work here. Lin et al. [152] presented an auto-masking neural network for vehicle detection and viewpoint estimation. In [86], an approach by vector matching of the template was introduced for vehicle color recognition. In [27], an unsupervised convolutional neural network was designed for vehicle type classification from frontal view images. However, all these models were implemented on their own small datasets without any robust comparisons.

3.3 Deep Belief Network for Vehicle Detection

In this section, we propose a method using deep belief network (DBN) to address vehicle detection. DBN is a class of deep neural network which consists of multiple layers of hidden units. It has demonstrated its success on the classification task in the MNIST [61] and NORB [109] datasets.

3.3.1 Deep Belief Networks

Deep Belief Network (DBN) is a generative graphical model. Through training the weights between different layers' neural nodes, the entire network can be used for probabilistically reconstructing inputs. The intermediate hidden layers can be regarded as feature detectors. Once each layer is pre-trained, supervision can be adopted to fine-tune the network to perform classification or other discriminative tasks.

DBN is composed of multiple layers of visible and hidden units. Each component of DBN contains an unsupervised Restricted Boltzmann Machine (RBM) which is an undirected generative energy-based model. It is significant that connection is not available between neural nodes within a layer but only exists between layers. Therefore, given the visible node activations $\mathbf{v}$, the hidden node activations $\mathbf{h}$ are mutually independent, which can be denoted as:

$$P(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{N} P(\mathbf{h}_j|\mathbf{v}). \tag{3.1}$$

Where N is the number of hidden nodes. Conversely, given the hidden node activations, M visible node activations are mutually independent as:

$$P(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{M} P(\mathbf{v}_i|\mathbf{h}). \tag{3.2}$$

The structures of the RBM and the DBN stacked by RBMs are illustrated in Figure 3.3. Training an RBM aims to maximize the product of probabilities assigned to a training set. Hinton et al. proposed the contrastive divergence algorithm [11] to train the RBM.

Training a DBN is to train each RBM sequentially. Specifically, after training the first RBM h_1, the second RBM h_2 is trained using the first RBM's hidden layer as the second RBM's visible layer. To train the second RBM, a training sample is clamped to x and

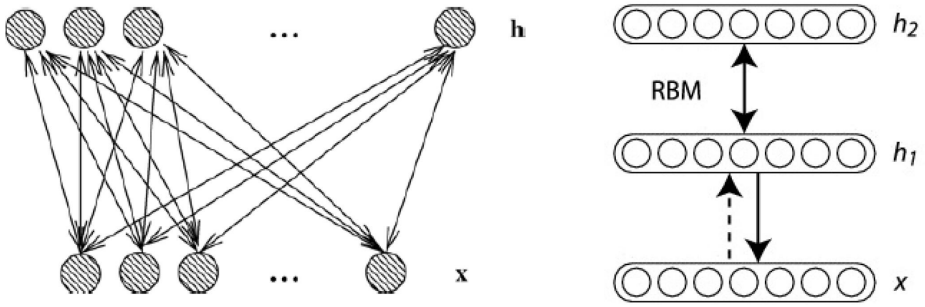


Fig. 3.3 The left sub figure is Restricted Boltzmann Machine (RBM) and the right one is the Deep Belief Network (DBN) which is stacked by RBMs.

transformed to h_1 by the first RBM and then contrastive divergence is used to train the second RBM. Training the second RBM is exactly equal to training the first RBM, except that the training data is mapped through the first RBM before being used as training samples. The intuition is that if the RBM is a general method for extracting a meaningful representation of data, then it should be robust for any embedding in which it is applied. Generally speaking, the RBM doesn't know whether the visible layer is image pixels or the output of another RBM. With this intuition, it becomes interesting to add a second RBM to see if a more representative higher level feature of the input can be learned. Hinton et al. have shown that adding a second RBM decreases a variational band on the log-likelihood of the training data. Once the pre-training of N layers in such unsupervised greedy manner is completed, a softmax layer can be configured at the end of the last RBM layer to fine-tune the entire network for certain discriminative tasks.

To use the DBN for classification, a sample is clamped to the lowest level visible layer and transformed upwards through the DBN until it reaches the top RBM hidden layer. At the top RBM, a few iterations of Gibbs sampling is performed after which the label is read out. Alternatively, the exact 'free energy' of each label can be computed and the one with the lowest free energy is chosen. To fine-tune the entire model for classification, Hinton et al. use an 'up-down' algorithm by normal back propagation algorithm.

3.3.2 Multi-level Complex Wavelet Features (CWF)

Given an input image containing vehicles with different poses, the descriptor is desired to be able to capture the orientation, intensity and even contour information which are the main cues of a vehicle. Meanwhile, a low time-cost for feature computation is also required. In our experiments, we adopt the multi-level complex wavelet features as the vehicle descriptors.

In texture feature extraction, conventional discrete wavelet transform (DWT) is an available option but suffers from shift variance which means huge oscillation in energy distribution of wavelet coefficients would be caused by small shift of the input signal. The dual-tree complex wavelet transform (DTCWT) [73] has been proposed to address this problem. It consists of two separate DWT decompositions performing the complex transform of an input image. The coefficients of the first DWT are real, while the coefficients of the second DWT are imaginary. An image signal $f(x,y)$ can be decomposed with a complex scaling and six complex wavelet operations, defined as the following equation:

$$f(x,y) = \sum_{l\in\mathbb{Z}^2} A_{j_0,l}\phi_{j_0,l}(x,y) + \sum_{k\in\alpha}\sum_{j=1}^{j_0}\sum_{l\in\mathbb{Z}^2} D^k_{j,l}\varphi^k_{j,l}(x,y). \tag{3.3}$$

Where j denotes different decomposition levels and l is the coordinates of each pixel. $A_{j_0,l}$ and $D^k_{j,l}$ are the scaling and wavelet coefficients, respectively. $\phi_{j_0,l}(x,y)$ is for scaling and $\varphi^k_{j,l}(x,y)$ are six wavelet functions which are oriented at $k \in \alpha = \{ -5\pi/12, -\pi/4, -\pi/12, \pi/12, \pi/4, 5\pi/12 \}$. It can produce six directionally selective sub-bands for each scale. The six kernels at different orientations are demonstrated in Figure 3.4.

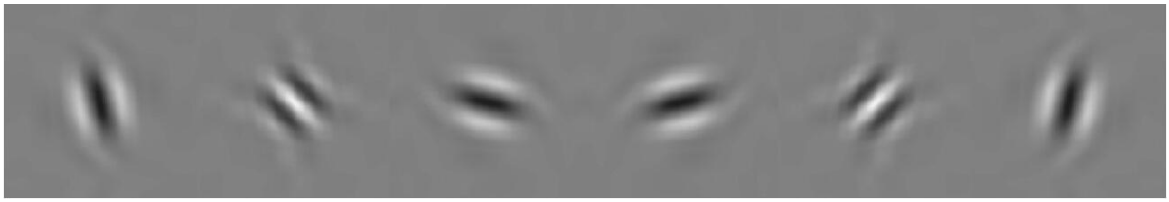


Fig. 3.4 Kernels of DTCWT at different orientations: $-5\pi/12$, $-\pi/4$, $-\pi/12$, $\pi/12$, $\pi/4$, and $5\pi/12$, from left to right.

The extraction of the multi-level complex wavelet features (CWF) is visualized in Figure 3.5. The CWF is composed of two parts: energy map (EM) and orientation map (OM). In OM, the orientation information is computed on each level of complex wavelet decomposition. Then, EM is calculated as the magnitudes of the OM for the corresponding scale level.

3.3.3 Implementation

The proposed detection framework is mainly based on a scheme of feature extraction and neural network (NN) for classification. NN is one of the classic methods for object detection, consisting of many simple elements called neurons which take input, change their activation

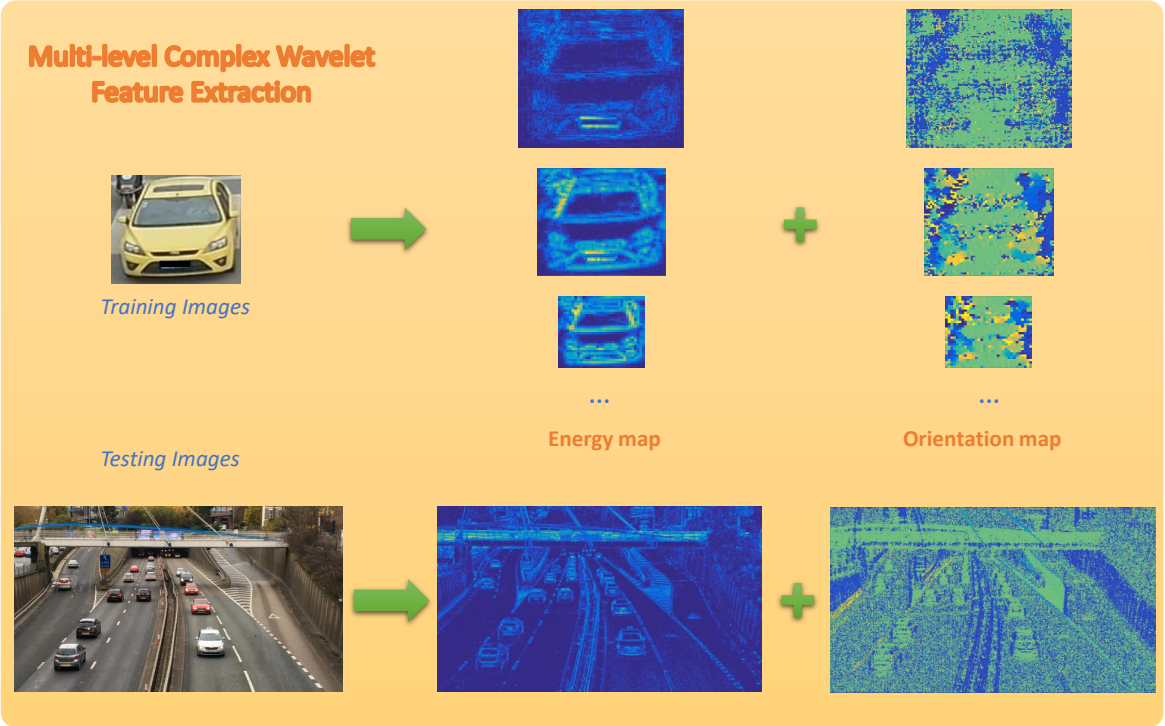


Fig. 3.5 Multi-level complex wavelet feature extraction.

according to that input, and produce output based on the input and activation. After setting the network structure, the DBN is adopted for pre-training layer by layer.

In the experiments, we variate the hyper-parameter settings of the number of layer and the number of units for each layer. A 3-layer network with 500 hidden units for each layer is finally constructed for our model, which is evaluated to achieve the best detection results. Moreover, we use 100,000 vehicle images in the CompCars dataset [153] as the positive training samples. 150,000 negative samples without any vehicles are cropped from Google Street View Images. All the training images are resized to four different level scales as 256×256, 128×128, 64×64 and 32×32 to extract their corresponding complex wavelet features. We adopt the training data to train the neural network as a binary classifier using the softmax cross-entropy loss defined in the following equation:

$$\hat{p}_{nk} = \exp(\mathbf{x}_{nk}) / [\sum_{k'} \exp(\mathbf{x}_{nk'})] \tag{3.4}$$

$$\mathscr{E}(\hat{p}_n) = \frac{-1}{N} \sum_{n=1}^{N} l_n \log \hat{p}_n + (1 - l_n) \log(1 - \hat{p}_n)$$

Where N is the number of training data. $l_n \in [0, 1, 2, ..., K-1]$ indicates the correct class label among the K classes and $\hat{p}_n$ is the softmax output class probabilities. The base learning rate is set as 0.01, and the model is trained by 10 epochs. Once the model training is completed, the trained NN weights are used as the kernels to perform filtering over the features of test frames. Thus, we can efficiently find the highest detection response which indicate the vehicles' locations on the detection map.

3.3.4 Experiments and Results

We compare the DBN-based method with three baselines using the AdaBoost, Tree AdaBoost and NN as classifiers, respectively. The experiments are evaluated on the PASCAL VOC2007 Car dataset [29]. Moreover, all the models are implemented based on the DeepLearnToolbox [112] and run on a 4.0 GHz CPU.

Baselines

CWF + AdaBoost: AdaBoost is one of the most popular machine learning algorithms, which aims to construct a strong classifier from several weak ones. Given a training set $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_N, y_N)\}$, where x_i is a pattern and $y_i \in \{+1, -1\}$ is the class label of the corresponding pattern. At the beginning, all the training patterns are assigned with the equal weights. In the learning phase, one weak classifier is trained and all patterns are updated. Patterns that are incorrectly classified have their weights increased, while the weights of those patterns that are correctly classified are decreased. After training, patterns that are consistently difficult to classify acquire large weights, while easily classified patterns acquire small weights. Here we adopt the Classification and Regression Tree as the basic weak classifier. The outline of AdaBoost selection algorithm can be found in [141] which also introduces the Viola-Jones detector. In our experiments, we train an AdaBoost classifier using the CWF features.

CWF + Tree AdaBoost: For an object class with subtle intra-class variation, e.g. frontal faces, a single cascade classifier proposed by Viola and Jones has been considered to be a successful and efficient method. However, for multi-view vehicle detection, a single AdaBoost classifier seems to be not effective for our tasks due to the huge orientation and shape variance. Thus, we design a new baseline for multi-view boost detection, which contains two main elements: unsupervised hierarchical sub-categorization and a boosting-based tree-structured detector. For the former part, locality preserving projection (LPP) is adopted to reduce the feature dimension, mapping representations into a compact space. Since CWF focuses on orientation responses, LPP will pre-group features beyond the orientations. Within this compact space, vehicle samples are clustered into several sub-categories by using

Gaussian mixture model (GMM) [41] method. These sub-categories, as leaf nodes, are progressively merged together to construct a tree from bottom up top illustrated in Figure 3.6.

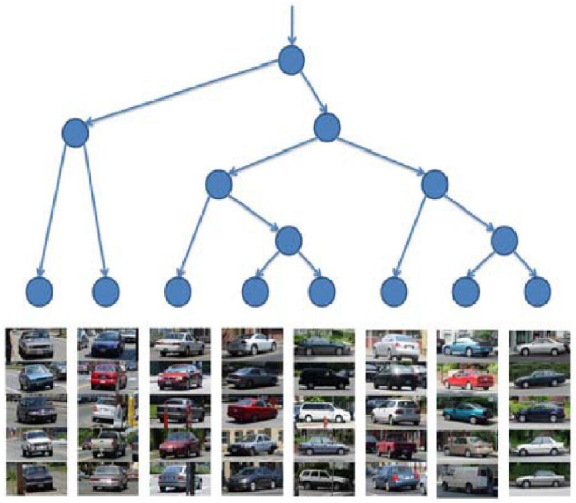


Fig. 3.6 Results of hierarchical GMM clustering. Two clusters merge first if they have similar viewpoints.

CWF + NN: This baseline is a similar version for the proposed model but each layer in the network is not pre-trained by DBN. The whole network is directly optimized using the back-propagation method.

Evaluation

All the methods are evaluated on the PASCAL VOC2007 Car dataset and compared both on the average precision and computational time. Results listed in Table 3.1 illustrate that the DBN+NN classifiers achieve higher average precision (AP) scores and detection speed based on the CWF compared with other traditional classifiers. Specifically, built on the same network architecture (3-layer, 500 units per layer), the DBN+NN increases 1.74% of AP compared with NN and largely reduces the training time, because the unsupervised DBN pre-training is effective to better encode the input images. Moreover, the NN-based methods can obtain higher frames per second (fps) compared with the AdaBoost-based methods, since the trained NN kernels can be directly filtered over the features of the input test frames to obtain the detection score maps. Besides, we compare different network architectures of NN and observe that the accuracy generally depends on the depth of the network and the capacity of each layer. We only studied these baselines for a preliminary understanding of deep learning so did not explore more architectures.

Table 3.1 Comparisons of different methods on average precision and computational time.

Methods	Average Precision (AP)	Training time (min)	Test Speed (fps)
CWF + AdaBoost	44.29%	41	0.8
CWF + Tree AdaBoost	47.95%	69	0.6
CWF + NN (3-layer, 500 units per layer)	47.38%	504	2.5
CWF + DBN + NN (2-layer, 300 units per layer)	47.20%	**29**	**3.2**
CWF + DBN + NN (2-layer, 500 units per layer)	47.86%	31	3.0
CWF + DBN + NN (3-layer, 300 units per layer)	48.03%	35	2.6
CWF + DBN + NN (3-layer, 500 units per layer)	**49.12%**	36	2.5

3.3.5 Conclusion

In this section, we have a preliminary investigation in deep neural networks for vehicle detection. However, the DBN-based methods cannot preserve the 2D spatial information of images, thus largely limits the detectors' performance. In the next section, a convolutional neural network-based framework is proposed to address vehicle detection and attributes learning simultaneously.

3.4 Convolutional Neural Networks for Vehicle Detection and Multi-tasking Learning

In this section, as illustrated in Figure 3.7, detection and annotation of vehicles (DAVE) on pose, color, and type are unified into one framework. DAVE consists of two convolutional neural networks called fast vehicle proposal network (FVPN) and attributes learning network (ALN), respectively. For training the models, FVPN and ALN are optimized together, while two-stage inference is performed in the test phase. FVPN aims to predict all the positions of vehicles in real-time. Afterwards, these vehicle candidates are passed to the ALN to simultaneously infer their corresponding pose, color, and type, and verification is also operated to discard those false alarms classified by FVPN. Training our DAVE is inspired by Hinton [60] that knowledge learned from solid deep networks can be distilled to teach shallower networks. We design to apply latent data-driven knowledge from the deep ALN to guide training the shallow FVPN. This method is proved to be able to enhance the performance of the FVPN through experiments. The architecture of FVPN and ALN are described in the following subsections. More detailed training and inference methods are presented as well.

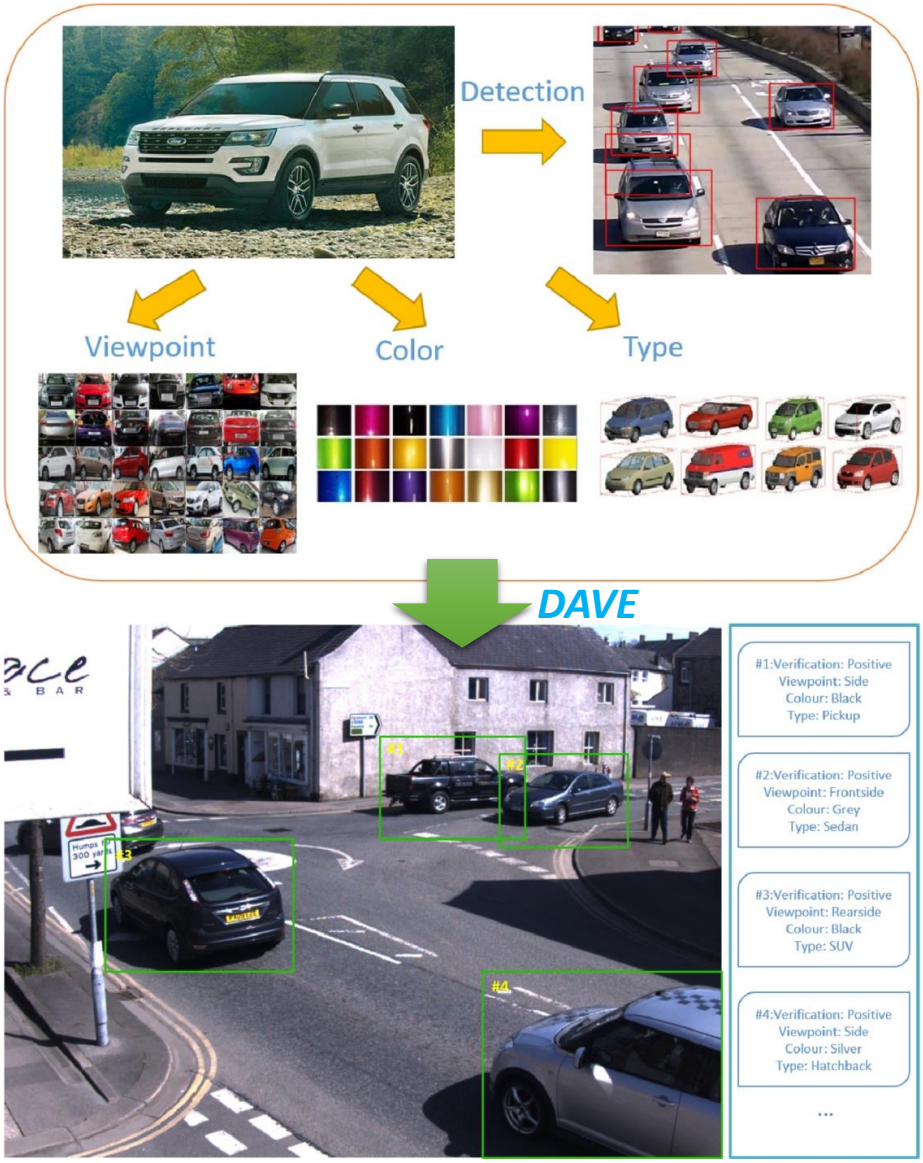


Fig. 3.7 Illustration of DAVE. A vehicle has many semantic attributes that can be applied to intelligent transportation systems, as shown in the upper sub-figure. Given numerous surveillance videos, human labeling is expensive and time-consuming. The motivation of our proposed DAVE is to annotate the location, pose, type and color of all the vehicles on the raw videos automatically.

3.4.1 Fast Vehicle Proposal Network (FVPN)

Searching the whole image to classify whether each region is a vehicle in a sliding window fashion is prohibitive for real-time applications. Traditional object proposal methods are put forward to alleviate this problem, but thousands of proposals usually contain numerous false alarms and duplicate predictions which heavily lower the efficiency. Particularly for one specific object, we expect very fast and accurate detection performance can be achieved.

Our proposed fast vehicle proposal network (FVPN) is a shallow fully convolutional network, which aims to precisely localize all the vehicles in real-time. We are interested in exploring whether or not a small scale CNN is enough to handle the single object detection task. A schematic diagram of the FVPN is depicted in the bottom part of Figure 3.8. The first convolutional layer (*conv_1*) filters the 60×60 resolution training images with 32 kernels of size 5×5. All the convolutional layers in FVPN are configured with stride parameter as 1 and padding as 0. The second convolutional layer (*conv_2*) takes as input the feature maps obtained from the previous layer and filters them with 64 kernels of size 5×5. Max pooling and Rectified Linear Units (ReLU) layers are configured after the first two convolutional layers. The third convolutional layer (*conv_3*) with 64 kernels of size 3×3 is branched into three siblings of 1×1 convolutional layer transformed by traditional fully-connected layers. In detail, *Conv_fc_class* outputs softmax probabilities of positive samples and the background; *Conv_fc_bbr* encodes bounding-box coordinates for each positive sample; *Conv_fc_knowledge* is configured for learning latent data-driven knowledge distilled from the ALN, which makes the FVPN be trained with more meticulous vehicle features. Inspired by [100], these 1×1 convolutional layers can successfully lead to differently purposed heatmaps in the inference phase. This property can achieve real-time vehicle localization from whole images/frames by our FVPN.

We employ different loss supervision layers for three corresponding tasks in the FVPN. First, discrimination between a vehicle and the background is a simple binary classification problem. A softmax loss layer is applied to predict vehicle confidence, $p^c = \{p^c_{ve}, p^c_{bg}\}$. Besides, each bounding-box is encoded by 4 predictions: (x, y, w, h) denoted as $\mathbf{loc}_{gt}$. x and y denote the left-top coordinates of the vehicle position, while w and h represent the width and height of the vehicle size. We normalize all the 4 values relative to the image width and height so that they can be bounded between 0 and 1 ($x \leftarrow x/width_{image}$, $y \leftarrow y/height_{image}$, $w \leftarrow w/width_{image}$, $h \leftarrow h/height_{image}$). Note that all bounding boxes' coordinates are set as *zero* for background samples. Following [37], a smooth L1 loss layer is used for bounding-box regression to output the refined coordinates vector, $\mathbf{loc} = (\hat{x}, \hat{y}, \hat{w}, \hat{h})$. Finally, for guiding with latent data-driven knowledge of an N-dimensional vector distilled from a deeper net, the cross-entropy loss is employed for $\mathbf{p}^{know} = \{p^{know}_0 \dots p^{know}_{N-1}\}$.

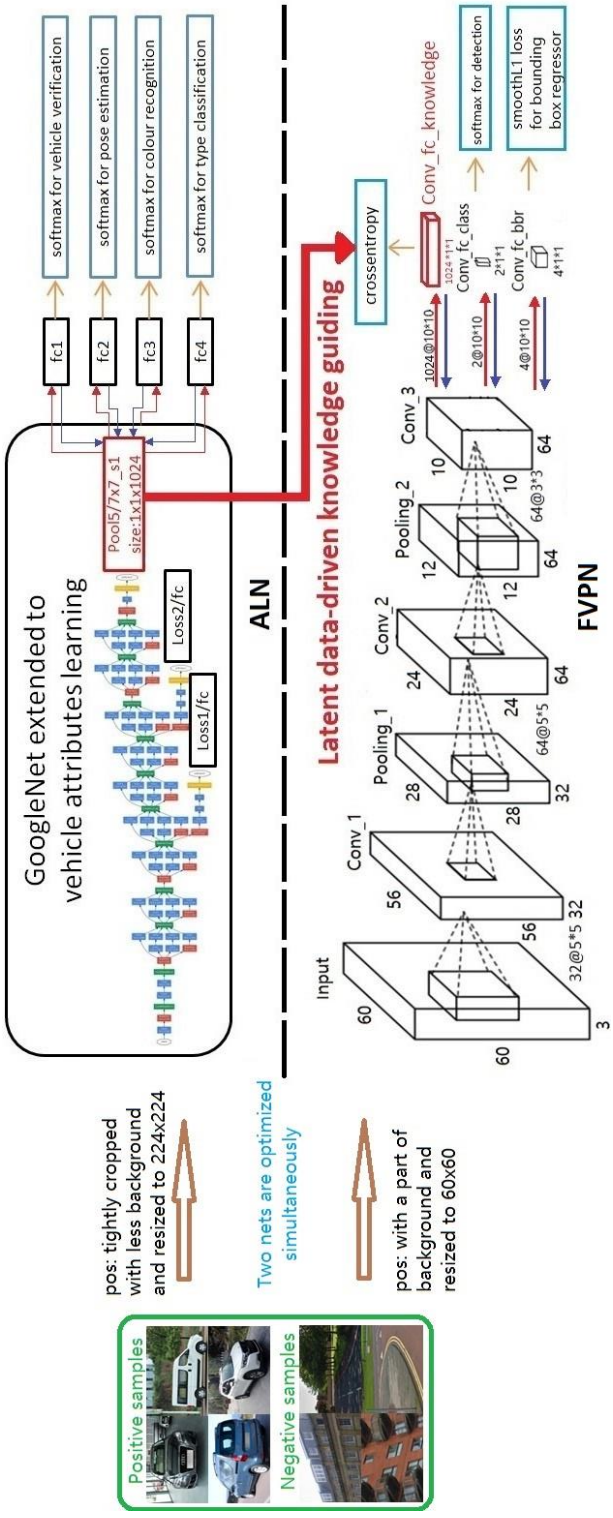


Fig. 3.8 Training Architecture of DAVE. The FVPN is a shallow fully convolutional network, which aims to precisely localize all the vehicles in real-time. The ALN is built by adding 4 fully-connected layers to extend the deep GoogLeNet into a multi-attribute learning model. These two networks are simultaneously optimized in a joint manner by bridging them with latent data-driven knowledge guidance.

We adopt a multi-task loss L_{FVPN} on each training batch to jointly optimize binary classification of the vehicle against background, bounding-box regression and learning latent knowledge from a deeper net as the following function:

$$L_{FVPN}(\mathbf{loc}, p^{bic}, \mathbf{p}^{know}) = L_{bic}(p^{bic}) + \alpha L_{bbox}(\mathbf{loc}) + \beta L_{know}(\mathbf{p}^{know}), \tag{3.5}$$

Where L_{bic} denotes the softmax loss for binary classification of vehicle and background. L_{bbox} indicates a smooth ℓ_1 loss defined in [37] as:

$$L_{bbox}(\mathbf{loc}) = f_{\ell_1}(\mathbf{loc} - \mathbf{loc}_{gt}), \tag{3.6}$$
$$\text{s.t. } f_{\ell_1}(\mathbf{x}) = \begin{cases} 0.5\mathbf{x}^2, \text{ if } ||\mathbf{x}|| < 1 \\ ||\mathbf{x}|| - 0.5, \text{ otherwise} \end{cases}$$

Furthermore, the cross-entropy loss L_{know} is to guide the training of the FVPN by a latent N-dimensional feature vector $\mathbf{t}^{know}$ learned from a more solid net, which is defined as:

$$L_{know}(\mathbf{p}^{know}) = -\frac{1}{N}\sum_{i=1}^{N} \mathbf{t}_i^{know} \log \mathbf{p}_i^{know} + (1 - \mathbf{t}_i^{know}) \log(1 - \mathbf{p}_i^{know}). \tag{3.7}$$

It is noteworthy that a bounding-box for the background is meaningless in the FVPN back propagation phase and will cause training to diverge early [118], thus we set $\alpha = 0$ for background samples, otherwise $\alpha = 0.5$, whiles β remains at a fixed weighting value of 0.5.

3.4.2 Attributes Learning Network (ALN)

Attributes learning is also an interesting task [95]. Modeling vehicles' pose, color and type information separately is less accurate and inefficient. Actually, relationships between these tasks can be explored, so that designing a multi-task network is beneficial for learning shared features which can lead to extra performance gains. The attribute learning network (ALN) is a unified network to verify vehicle candidates and annotate their poses, colors, and types. The network architecture of the ALN is mainly inspired by the GoogLeNet [135] model. Specifically, we design the ALN by adding 4 fully-connected layers to extend the GoogLeNet into a multi-attribute learning model. The reason to adopt such a very deep structure here is

that vehicle annotation belongs to fine-grained categorization problems and a deeper net has the more powerful capability to learn representative and discriminative features. Another advantage of the ALN is its high-efficiency inherited from the GoogLeNet which has lower computation and memory costs compared with other deep nets such as the VGGNet [128].

The ALN is a multi-task network optimized with four softmax loss layers for vehicle annotation tasks. Each training image has four labels in V, P, C and Y. V determines whether a sample is a vehicle. If V is a true vehicle, the remaining three attributes P, C and Y represent its pose, color and type respectively. However, if V is the background or a vehicle with a catch-all[1] type or color, P, C and Y are set as *zero* denoting attributes are unavailable in the training phase. The first softmax loss layer $L_{verify}(p^V)$ for binary classification (vehicle vs. background) is the same as $L_{bic}(p^c)$ in the FVPN. The softmax loss $L_{pose}(p^P)$, $L_{color}(p^C)$ and $L_{type}(p^Y)$ are optimized for pose estimation, color recognition and vehicle type classification respectively, where $p^P = \{p_1^P, \ldots, p_{np}^P\}$, $p^C = \{p_1^C, \ldots, p_{nc}^C\}$ and $p^Y = \{p_1^Y \ldots, p_{nt}^Y\}$. $\{np, nc, nt\}$ indicate the number of vehicle poses, colors and types respectively. The whole loss function is defined as follows:

$$\begin{aligned} L_{ALN}(p^V, p^P, p^C, p^Y) = L_{verify}(p^V) + \lambda_1 L_{pose}(p^P) \\ + \lambda_2 L_{color}(p^C) + \lambda_3 L_{type}(p^Y), \end{aligned} \tag{3.8}$$

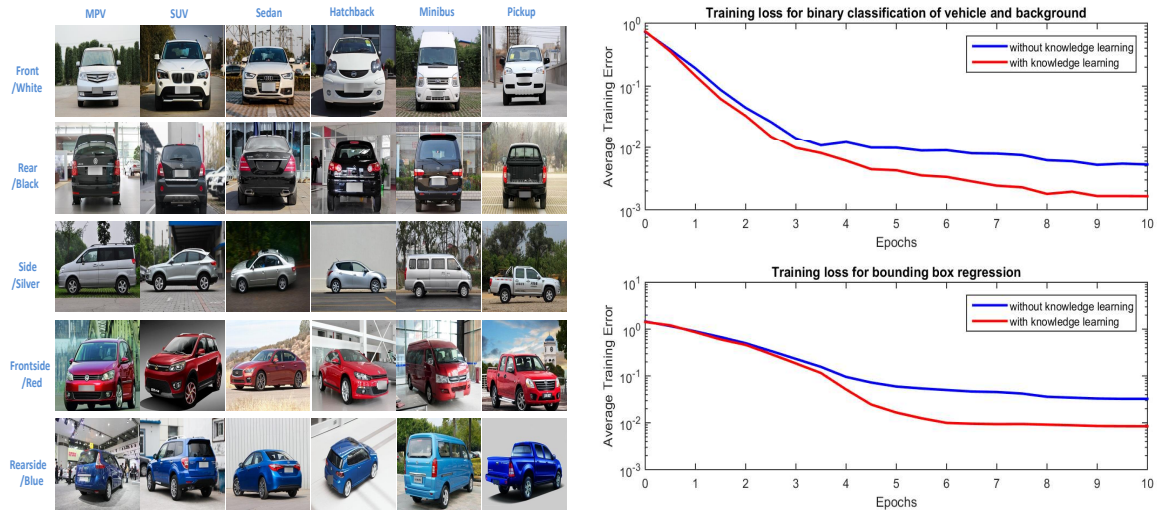
Where all the four sub loss functions are softmax loss for vehicle verification (***"verification" in this section means confirming whether a detection is a vehicle***), pose estimation, color recognition, and type classification. Following the similar case of α in Eq. 3.5, parameters $\{\lambda_1, \lambda_2, \lambda_3\}$ all remain at a fixed weighting value of 1 for the positive samples, otherwise 0 for the background.

3.4.3 Deep Nets Training

Training Dataset and Data Augmentation

To train the DAVE, we also adopt the large-scale CompCars dataset [153] with more than 100,000 web-nature data as the positive training samples which are annotated with tight bounding-boxes and rich vehicle attributes such as pose, type, make and model. In detail, the web-nature part of the CompCars dataset provides five poses as *front*, *rear*, *side*, *frontside* and *rearside*, twelve vehicle types as *MPV, SUV, sedan, hatchback, minibus, pickup, fastback, estate, hardtop-convertible, sports, crossover* and *convertible*. To achieve a uniform training

[1] "Catch-all" indicates other undefined types and colors which are not included in our training model.



(a) (b)

Fig. 3.9 (a) Training data (columns indicate vehicle types, while rows indicate poses and colors), (b) Training loss with/without knowledge learning.

distribution, we discard less common vehicle types with few training images and finally select six types with all the five poses illustrated in Figure 3.9(a) to train our model. Besides, since the color is another important vehicle attribute, we additionally annotated colors on more than 10,000 images with five common vehicle colors as *black, white, silver, red* and *blue* to train our final model. Apart from positive samples, about 150,000 negative samples (by hard negative mining) without any vehicles are cropped from Google Street View Images to compose our training data.

For data augmentation, we first triple the training data with increased and decreased image intensities for making our DAVE more robust under different lighting conditions. In addition, image downsampling up to 20% of the original size and image blurring are introduced to improve annotation precision and recall of detected vehicles that were small in scale within the image.

Jointly Training with Latent Knowledge Guidance

The entire training structure of DAVE is illustrated in Figure 3.8. We optimize the FVPN and the ALN jointly but with different sized input training data at the same time. The input resolution of the ALN is 224×224 for fine-grained vehicle attributes learning, while it is decreased to 60×60 for the FVPN to fit smaller scales of the test image pyramid for efficiency in the inference phase. In fact, the resolution of 60×60 can well guarantee the coarse shape and texture of a vehicle is discriminative enough against the background.

Besides, another significant difference between the ALN and the FVPN is that input vehicle samples for the ALN are tightly cropped, however, for the FVPN, uncropped vehicles are used for bounding-box (labeled as loc_t in Eq. 3.6) regressor training.

The pre-trained GoogLeNet model for 1000-class ImageNet classification is used to initialize all the convolutional layers in the ALN, while the FVPN is trained from scratch. A 1024-dimensional feature vector of the *pool5/7×7_s1* layer in the ALN, which can exhaustively describe a vehicle, is extracted as the latent data-driven knowledge guidance to supervise the same dimensional *Conv_fc_knowledge* layer in the FVPN by cross-entropy loss. We set the dimension of layer *Conv_fc_knowledge* in FVPN with the same value of 1024 correspondingly.

We first jointly train ALN and FVPN for about 10 epochs on the selected web-nature data that only contains pose and type attributes from the CompCars dataset. In the next 10 epochs, we fine-tune the models by a subset with our complementary color annotations. Throughout the training process, we set the mini-batch size as 64, and the momentum and weight decay as 0.9 and 0.0005 using empirical settings, respectively. Learning rate is scheduled as 10^{-3} for the first 10 epochs and 5×10^{-4} for the second 10 epochs. To make our method more convincing, we train two models with and without knowledge guidance, respectively. During training, we definitely discover that knowledge guidance can indeed benefit training the shallow FVPN to obtain lower training losses. Training loss curves for the first 10 epochs are depicted in Figure 3.9(b).

3.4.4 Two-stage Deep Nets Inference

Once the joint training is finished, a two-stage scheme is implemented for inference of DAVE. First, the FVPN takes as input the 10-level test image Gaussian pyramid. For each level, the FVPN is operated over the input frame to infer *Conv_fc_class* and *Conv_fc_bbr* layers as corresponding heatmaps. All the 10 *Conv_fc_class* heatmaps are unified into one map by rescaling all the channels to the largest size among them and keeping the maximum along channels, while the index of each maximum within 10 channels is used to obtain four unified *Conv_fc_bbr* heatmaps (10 levels by similar rescaling). After unifying different levels *Conv_fc_class* heatmaps into the final vehicle detection score map, we first filter the score map with threshold *thres* to discard low hot spots, and then local peaks on the map are detected by a circle scanner with tuneable radius r. In all our experiments, $r = 8$ and $thres = 0.5$ are evaluated to achieve the highest average precision. Thus, these local maximal positions are considered as the central coordinates of proposals, $(\hat{x}_i, \hat{y}_i)$. Coarse width and height of each detection can be simply predicted based on the bounding-box of its corresponding hot spot centered on each local peak. If one hot spot contains multiple peaks, the width

and height will be shared by these peaks (i.e. proposals). For preserving the complete vehicle body, coarse width and height are multiplied by fixed parameter $m = 1.5$ to generate ($\hat{w}_i^{nobbr}$,$\hat{h}_i^{nobbr}$). Thus, a preliminary bounding-box can be represented as ($\hat{x}_i$,$\hat{y}_i$,$\hat{w}_i^{nobbr}$,$\hat{h}_i^{nobbr}$). Finally, bounding-box regression offset values (within [0,1]) are extracted from four unified heatmaps of *Conv_fc_bbr* at those coordinates ($\hat{x}_i$,$\hat{y}_i$) to obtain the refined bounding-box.

Vehicle candidates inferred from the FVPN are taken as inputs into the ALN. Although verifying each detection and annotation of attributes are at the same stage, we assume that verification has a higher priority. For instance, in the inference phase, if a detection is predicted as a positive vehicle, it will then be annotated with a bounding-box and inferred pose, color, and type. However, a detection predicted as the background will be neglected in spite of its inferred attributes. Finally, we perform non-maximum suppression (NMS) adopted in RCNN [38] to eliminate duplicate detections. NMS clusters all the bounding-boxes that have overlap with each other greater than 0.5. For each cluster, NMS calculates the mean bounding-box and output it (to calculate the mean point between all top right corners and all bottom-right corners). The full inference scheme is demonstrated in Figure 3.10. At present, it is difficult to train a model that has the capability to annotate all the vehicles with enormously rich vehicle colors and types. During inference, a vehicle with untrained colors and types is always categorized into similar classes or a catch-all “others" class, which is a limitation of DAVE. In future work, we may expand our training data to include more abundant vehicle classes.

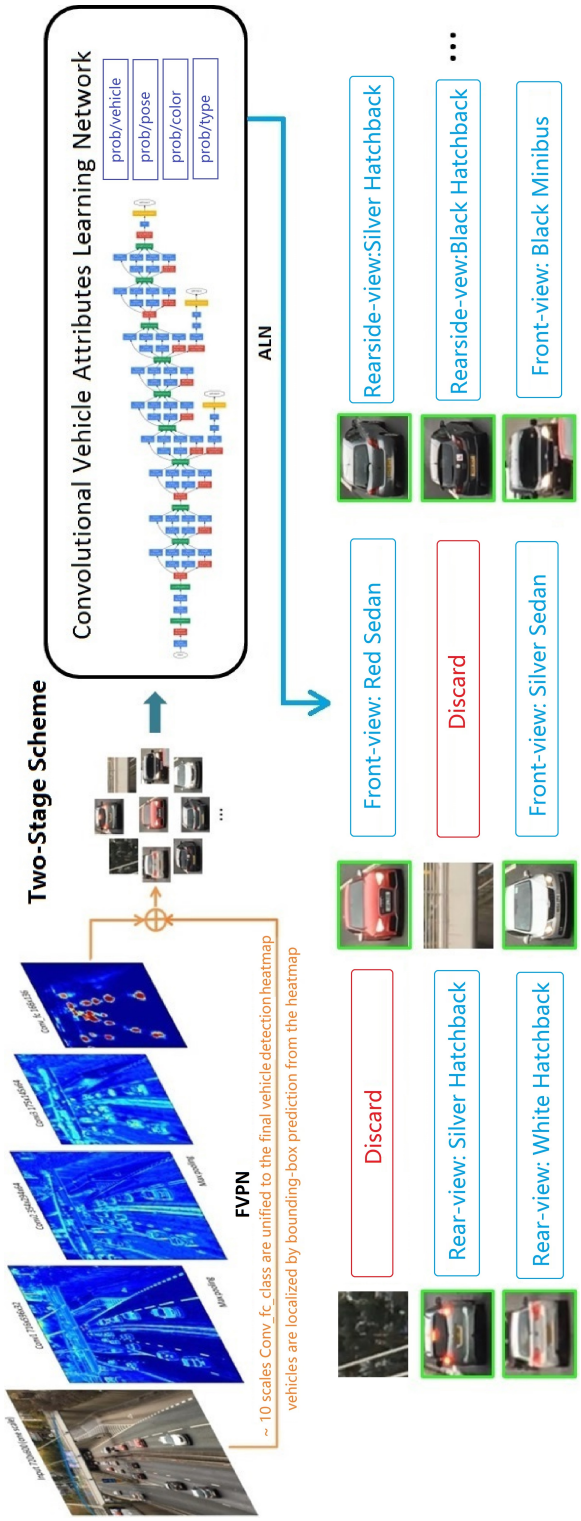


Fig. 3.10 A two-stage inference phase of DAVE. Vehicle candidates are first obtained from FVPN in real-time. Afterwards, we use ALN to verify each detection and annotate each positive one with the vehicle pose, color and type.

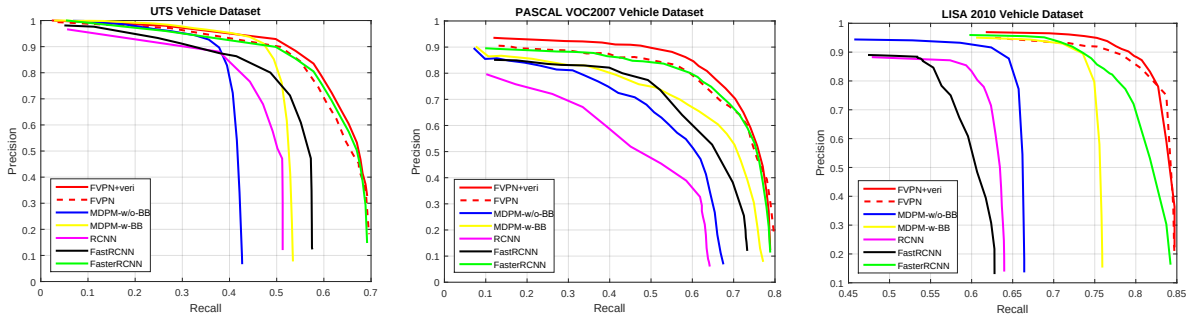
3.4.5 Experiments and Results

In this sub-section, we evaluate our DAVE for detection and annotation of pose, color, and type for each detected vehicle. Experiments are mainly divided into two parts: vehicle detection and attributes annotation. DAVE is implemented based on the deep learning framework Caffe [67] (introduced in Appendix B.1) and run on a workstation configured with an NVIDIA TITAN X GPU.

Evaluation of Vehicle Detection

To evaluate vehicle detection, we train our models using the large-scale CompCars dataset as mentioned before, and test on three other vehicle datasets. We collect a full high definition (1920×1080) Urban Traffic Surveillance (UTS) vehicle dataset with six videos which were captured from different viewpoints and illumination conditions. Each video sequence contains 600 annotated frames. To be more convincing, we also compare our method on two other public datasets: the PASCAL VOC2007 car dataset [29] and the LISA 2010 dataset [129] with four competitive models: DPM [33], RCNN [38], Fast RCNN [37] and Faster RCNN [120]. These four methods obtain state-of-the-art performances on general object detection and the codes are publicly available. We adopted the trained car model from voc-release5 [Girshick et al.] for DPM, while the competitive NN models (VGG-16 based) were trained for this study using the CompCars dataset to implement vehicle detection. The vehicle detection evaluation criterion is the same as PASCAL object detection [29]. Intersection over Union (IoU) is set as 0.7 to assess correct localization.

Testing on the UTS dataset We not only test our real-time and highly accurate FVPN independently but also verify that, by the deeper ALN (i.e., FVPN+verify in Figure 3.11), the detection performance can be further improved, because some false alarms by the shallow FVPN will be discarded after the more rigorous ALN. The detection accuracy as average precision (AP) and speed as frames-per-second (FPS) are compared in the left column of Table 3.2. Our model outperforms all the other methods with obvious improvements. Specifically, the shallow FVPN obtains an increased AP of 2.11% compared to the best model Faster RCNN, while the detection speed is significantly improved from 4 fps to 30 fps which can be termed as real-time. After verification by the deep ALN, 1.1% AP increase can be further achieved compared to FVPN, but the efficiency superiority is lost due to the deep architecture of ALN. However, the more complicated ALN is designed for fine-grained vehicle attributes annotation. Therefore, if we only consider implementing vehicle detection tasks, our proposed FVPN is preferred to be independently adopted as a high-performance vehicle detector.



(a) (b) (c)

Fig. 3.11 Precision-recall curves on three vehicle datasets. FVPN+veri illustrates the detection results after verification by the ALN. MDPM-w/o-BB and MDPM-w-BB denote Mixture-DPM without / with bounding-box prediction, respectively.

Table 3.2 Vehicle detection AP (%) and speed (fps) comparison on the UTS, PASCAL VOC2007 and LISA 2010 datasets

Methods	UTS		PASCAL VOC2007		LISA 2010	
	Average Precision (AP)	Processing Speed (fps)	Average Precision (AP)	Processing Speed (fps)	Average Precision (AP)	Processing Speed (fps)
MDPM-w/o-BB	41.96%	0.25	48.44%	1.25	63.61%	0.7
MDPM-w-BB	52.73%	0.2	57.14%	1.25	72.89%	0.7
RCNN	44.87%	0.03	38.52%	0.08	55.37%	0.06
FastRCNN	51.58%	0.4	52.95%	0.5	53.37%	0.5
FasterRCNN	59.82%	**4**	63.47%	**6**	77.09%	**6**
FVPN-w/o-knowledge guide	55.73%	30	60.27%	46	73.88%	42
FVPN-w/o-bbr	57.04%	30	60.81%	46	73.46%	42
FVPN	61.93%	**30**	65.12%	**46**	80.46%	**42**
FVPN+Verify	**63.03%**	2	**66.44%**	4	**81.10%**	4

"bbr" indicates the bounding-box regressor used in our model, while "BB" denotes bounding-box prediction used in DPM model. "w" and "w/o" are the abbreviations of "with" and "without", respectively. "Verify" denotes the vehicle verification in the ALN.

The other two deep models, RCNN and Fast RCNN, do not produce satisfactory results mainly due to the low-precision proposals extracted by Selective Search [138]. Mixture-DPM with bounding-box prediction (MDPM-w-BB [33]) significantly improve the performance compared to MDPM-w/o-BB [33] by 10.77%. In addition, the speed of all these baselines is slower than 1 fps which is far from real-time vehicle detection.

We also test the FVPN trained without knowledge guidance, with the AP decreased by 6.20%, which proves the significant advantage of knowledge guidance. Moreover, if FVDN-w/o-bbr is adopted for simplifying the algorithm, most predicted bounding boxes will get some offsets or include more backgrounds, which makes detection unsatisfactory. Corresponding experiments are carried out to demonstrate that bounding-box regression can be helpful with the AP increased by 4.89%.

Testing on the PASCAL VOC2007 car dataset and the LISA 2010 dataset To make our methods more convincing, we also evaluate on two other public datasets. All the images containing vehicles in the trainval and test sets (totally 1434 images) in the PASCAL VOC 2007 dataset are extracted to be evaluated. In addition, the LISA 2010 dataset contains three video sequences with low image quality captured by an on-board camera. All the results are shown in the middle and right columns of Table 3.2. For the PASCAL VOC2007 dataset, the FVPN achieves 65.12% in AP with high-speed of 46 fps, which outperforms MDPM-w-BB, RCNN, FastRCNN and Faster RCNN by 7.98%, 26.6%, 12.17% and 1.65%, respectively. Likewise, FVPN+veri can even obtain a higher AP of 66.44%. Similarly, for the LISA 2010 dataset, the highest accuracy of 81.10% by FVPN+veri and 80.46% by only FVPN beats all other methods as well. Therefore, it demonstrates that our method is able to stably detect vehicles with different viewpoints, occlusions, and varied image qualities.

Figure 3.11 presents the precision-recall curves of all the compared methods on UTS, PASCAL VOC2007 car and the LISA 2010 datasets, respectively. From all these figures, we can further discover that, for all three datasets, both FVPN+Verify and only FVPN-based system achieve better performance than other vehicle detection methods by comparing Area Under the Curve (AUC). Besides, some qualitative detection results including successful and failure cases are shown in Figure 3.12. It can be observed that the FVPN cannot handle highly occluded cases at very small sizes, since local peaks within the corresponding FVPN heat map overlap. The similar situation also exists in most of the deep networks based detection approaches [38, 37, 118, 120].

PASCAL VOC2007 Car dataset error analysis To further examine the detection results, we look into an in-depth error analysis on the VOC2007 car dataset. We categorize all the positive predictions into four different types with their corresponding IoU settings:

- Correct localization, $IoU >= 0.7$
- Incorrect localization due to occlusions, $0.3 <= IoU < 0.7$
- Incorrect localization due to others, $0.3 <= IoU < 0.7$
- False alarm due to backgrounds, $IoU < 0.3$

Figure 3.13 demonstrates the error type analysis of false positives by a pie chart. We find that the severe occlusion between cars is the main factor for incorrect bounding-box prediction. Apart from false positives, the case of missing detection (i.e. false negatives) is always observed when the scale size is extremely small or a car body is occluded over 50%. Moreover, we can observe that vehicles in some dark colors cannot be detected under

Fig. 3.12 Examples of successful and failure cases for detection. A green box denotes correct localization, a red box denotes false alarm and a blue box denotes missing detection.

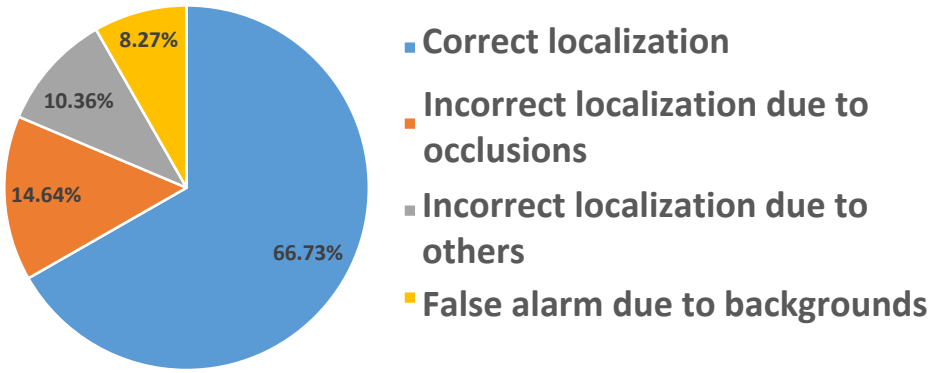


Fig. 3.13 Error analysis for detection results on VOC2007 car dataset. It shows the false positive detections are mainly due to the incorrect localization.

very low illumination. If we drop the data augmentation by adjusting image intensities for training, this situation will become more severe, decreasing the AP to 62.35%.

Evaluation of Vehicle Attributes Annotation

The experiments and analysis of the ALN are mainly based on the CompCars dataset and the UTS dataset. The web-nature data in the CompCars dataset are labeled with five viewpoints and twelve types about 136000 and 97500 images, respectively. We neglect those images without type annotation and randomly split the remaining data into the training and validation subsets as 7:3. In addition to pose estimation and type classification, we complement the annotation of five common vehicle colors on about 10000 images for evaluation of color recognition. Besides, for type classification, we compare the results of both the selected 6 common vehicle types and the total 12 types as mentioned in Section 3.4.3. Vehicle verification (i.e., binary classification of vehicle and background) is evaluated in all the experiments as well.

In the following subsections, we first explain the superiority of our method compared to the state-of-the-art one-net pipeline. Then, we implement four different experiments to investigate the gain of the multi-task architecture, the accuracy by inputs with different image qualities, the effect of layer depths and the difficulty of fine-grained classification under different viewpoints.

Comparison to state-of-the-art one-net pipeline For investigating the necessity of our two-stage inference architecture for vehicle detection and attributes annotation, we compare it with the one-net pipeline FasterRCNN VGG-16 [120] trained with multi-attributes learning. We modify the last fully-connected layers of FasterRCNN to implement both bounding-box regression for detection and softmax for different attributes classification. Table 3.3 illustrates that the annotation accuracy of one-net FasterRCNN is much lower than that of our DAVE. The reasons are as follows. Vehicle attributes annotation is a fine-grained task, which requires relatively high-resolution vehicle images for better results, especially for vehicle type. Thus, the input frame of FVPN for a test should be large (e.g. 1920×1080) to ensure all vehicle proposals inside are clear and informative to be fed into ALN for better annotation. However, one-net FasterRCNN has to take fixed size input (600×600) for initial layers (i.e. RPN), which leads to relatively small-sized vehicle proposals (usually less than 100×100) that subsequently reduced the performance on later annotation due to lack of visual details. Although we can reconstruct FasterRCNN by uniformly amplifying the spatial size of each layer to meet the requirement of resolution for good annotation, the computational and memory costs will dramatically increase and the detection speed will drop

to 0.6 fps for FasterRCNN. Therefore, our two-stage inference architecture is necessary and achieves significant advancement in real-world vehicle annotation tasks.

Table 3.3 Evaluation (%) of attributes annotation compared to one-net pipeline on the UTS dataset

Methods	Detection	Pose Estimation	12-type Classification	6-type Classification	Color Recognition
Faster RCNN VGG-16	59.82	91.30	58.19	88.43	69.44
DAVE	63.03	98.03	69.64	94.91	79.25

Single-task learning or multi-task learning? To explore this problem, we compare the multi-task ALN with the case of training networks for each attribute separately (i.e., single task). In addition, results by the combination of deeply learned features and an SVM classifier are compared as well. All the model architectures are based on the GoogLeNet (introduced in Appendix A), and 1024-dimensional features are extracted from layer *pool5/7*$\times$7*_s1* to train the corresponding SVM classifier [13]. As shown in the top part of Table 3.4, the multi-task model consistently achieves higher accuracies on four different tasks, which reveals the benefit of joint training. Although the combination of extracted features and SVM classifiers sometimes can lead to a small increase, we still prefer the proposed end-to-end model because of its elegance and efficiency.

How small a vehicle size can DAVE annotate? Since vehicles within surveillance video frames are usually in different sizes. Visual details of those vehicles far from the camera are significantly unclear. Although they can be selected by the FVPN with coarse requirements, after rescaling to 224×224, these vehicle proposals with low image clarity are hard to be annotated with correct attributes by the ALN. To explore this problem, we test vehicle images with original sizes of 224, 112, 56 and 28 using the trained ALN. The middle part of Table 3.4 illustrates that the higher resolution the original input size is, the better accuracy it can achieve.

Deep or shallow? How deep of the network is necessary for vehicle attributes learning is also worth to be explored. Since our ALN can be established on different deep models, we compare popular deep networks: AlexNet [76] and GoogLeNet with 8 layers and 22 layers, respectively. As VGGNet (16 layers version) [128] configured with numerous parameters requires heavy computation and large memory, we do not expect to employ it for our ALN. Besides, our proposed shallow FVPN with 4 layers is also used for attributes learning. From the bottom part of Table 3.4, we can see that a deeper network does not obtain much better performance on vehicle verification compared to a shallow one. However, for pose estimation, type classification and color recognition, the deepest GoogLeNet consistently outperforms other nets with obvious gaps. Particularly for type classification which belongs to fine-grained categorization, the shallow FVPN gives extremely poor results. It illustrates that a deeper

Table 3.4 Evaluation (%) of attributes annotation for vehicles on the UTS dataset

Tasks	**Vehicle Verification**	**Pose Estimation**	**Vehicle Type Classification**		**Color Recognition**
			12 types	**6 types**	
Comparison of single-task learning (STL) and multi-task learning (MTL) for attributes prediction					
STL	98.73	96.94	60.37	88.32	78.33
MTL	**99.45**	98.03	69.64	94.91	**79.25**
STL feature+SVM	99.11	97.12	60.86	90.75	78.06
MTL feature+SVM	99.36	**98.10**	**69.86**	**95.12**	79.19
Comparison of Attributes prediction with different sizes of vehicle images					
28×28	90.45	83.49	37.52	53.66	49.73
56×56	98.12	91.33	52.02	77.02	66.14
112×112	99.37	96.56	63.41	90.67	**80.31**
224×224	**99.45**	**98.03**	**69.64**	**94.91**	79.25
Comparison of Attributes prediction with different deep models					
ALN based on FVPN ($depth = 4$)	95.96	81.21	27.26	43.12	65.12
ALN based on AlexNet ($depth = 8$)	**99.51**	95.76	66.01	89.25	77.90
ALN based on GoogLeNet ($depth = 22$)	99.45	**98.03**	**69.04**	**94.91**	**79.25**

network with powerful discriminative capability is more suitable for fine-grained vehicle classification tasks.

Fine-grained categorization in different views. Finally, since vehicle type classification belongs to fine-grained categorization, we are interested in investigating its difficulty in different views due to its importance for our future work such as vehicle re-identification. As demonstrated in Table 3.5, for both 12-type and 6-type classification, higher precision is easier to be achieved from the side and rear-side views, while it is difficult to discriminate vehicle types from the front view. In other words, if we aim to re-identify a target vehicle from two different viewpoints, the type annotation predicted from a side view is more credible than that from a front view.

Table 3.5 Evaluation (%) of fine-grained vehicle type classification on the UTS dataset

Number of vehicle type	Front	Rear	Side	FrontSide	RearSide
12	58.02	60.37	66.73	61.28	64.90
6	79.42	84.60	92.93	86.77	92.53

Figure 3.14 shows some qualitative evaluation results of our DAVE on vehicle attributes annotation. It demonstrates that our model is robust to detect vehicles and annotate their poses, colors, and types simultaneously for urban traffic surveillance. The failure cases mainly take place on incorrect colors and vehicle types.

3.4.6 Conclusion

In this section, we developed a multi-task learning framework for fast vehicle detection and annotation: DAVE, which consists of two convolutional neural networks FVPN and

Fig. 3.14 Qualitative results of attributes annotation. Red marks denote incorrect annotation, and N/A(C) means a catch-all color.

ALN. The detection and attributes learning networks predict bounding-boxes for vehicles and infer their attributes: pose, color, and type, simultaneously. Extensive experimental results have shown that our method outperforms state-of-the-art frameworks and achieves a highly accurate vehicle attributes annotation system.

3.5 Discussion

In the current computer vision and deep learning community, researchers always reluctantly build end-to-end systems which aim to make all the results outputted from one model. The true novelty of DAVE emphasizes building a balanced and least-cost framework not only for vehicle detection but more importantly for accurate attributes annotation in realistic scenarios. We originally indeed tried one end-to-end pipeline with attributes annotation. Though such one-net framework seems elegant theoretically, the low annotation accuracy and high complexity are unacceptable based on experiments.

Since each vehicle looks highly different in varying viewpoints and many vehicles have almost similar appearances, learning a viewpoint-invariant feature of a vehicle is challenging but highly useful in tasks such as vehicle re-identification. Compared with vehicle detection and coarsely semantic attributes annotation, vehicle re-identification requires more detailed feature representations to discriminate different vehicle identities. In the next two chapters, several algorithms are proposed to address the multi-view vehicle re-identification problem.

Chapter 4

Cross-View Image Generation for Vehicle Re-identification

This chapter will discuss another widely-explored research topic called re-identification (re-ID) for intelligent video surveillance. The video surveillance and security management plays a significant role in current public transportation systems. In some applications, we usually need to locate a target object such as vehicle or person in a large city area or track it within a long-term duration. Object re-identification aims to address this problem of matching a query target with a huge number of gallery candidates captured from many other non-overlapping cameras. In the past decade, person re-ID has attracted large attention and been investigated extensively. However, vehicle-related researches only focus on vehicle detection and classification. Vehicle re-ID is an area that has received far less attention in the computer vision community than the prevalent person re-ID. Possible reasons for this slow progress are the lack of appropriate research data and the special 3D structure of a vehicle. A few of existing vehicle re-ID methods have generally focused on some specific views (e.g. front), but these methods are less effective in realistic scenarios where vehicles usually appear in arbitrary views to cameras. Moreover, some researchers prefer to exploit license plate and spatial-temporal information to achieve high-performance re-ID, but such data is hard to be accessed in real applications. Therefore, we aim to propose more general vision-based solutions to address the multi-view vehicle re-ID problem.

4.1 Introduction and Motivation

Intelligent vehicle surveillance techniques have been widely explored in the past decades. Most researches focus on vehicle detection [130, 134, 9] and recognition [153, 30, 171]

tasks. However, a more interesting and challenging problem, called vehicle re-identification (re-ID), has not achieved much progress. Vehicle re-ID solves the problem of searching a target vehicle in numerous non-overlapping cameras in the public surveillance system. It can be applied to many practical scenarios such as urban surveillance and security. However, due to the special 3D structure, a vehicle usually looks highly different in varying viewpoints. On the contrary, two similar but different vehicles in the same viewpoint always look more similar than two different viewpoints of the same vehicle by machine vision. Thus, the cross-view vehicle matching task is considered extremely challenging.

Compared with a similar problem called person re-ID [42] in which many great algorithms have been proposed, vehicle re-ID suffers from more difficulties and challenges. As shown in Figure 4.1, images of the same human usually have overlapped appearance even though with large viewpoint variations since a human body is usually upright, and the texture or color of his/her wear will not vary severely in different viewpoints. However, visual features of vehicles in varying views can be totally different. For instance, there is no overlap on visual pattern of the vehicle across the front, side and rear viewpoints. Conventional person re-ID methods, in which multi-view processing was not a large consideration, are not reasonable to be transferred directly to vehicle re-ID.

Person re-ID and vehicle re-ID are two highly similar problems but just target on different objects. Since person re-ID has achieved more solid investigations, we first give a brief introduction of person re-ID methods. Previous person re-ID methods can be categorized into four main groups: feature extraction, distance metric learning, subspace learning and deep learning. The first group focuses on designing robust features. For person, they usually vertically split the human body into several parts to extract local features [47, 31, 162, 88], but neglect the horizontal changes across different viewpoints. The distance metric [25, 147, 148, 74] and subspace [159, 89, 169, 157] learning methods optimize models in brute-force ways to shorten the distance between the same person and enlarge that between different people, which can only achieve limited improvements since the feature spaces across different viewpoints are not subject to the same manifold. Moreover, recent widespread deep learning methods [85, 1, 17, 161] aim to solve feature learning and distance metric learning simultaneously within one end-to-end model.

Researches on vehicle re-ID, particularly vision-based algorithms, have obtained increasing attention in the past two years. Vehicle re-ID can be classified into two cases: single-view and multi-view. Some early works have been presented in [30, 158, 68] to address the front-view re-ID or side-view re-ID. The one-view re-ID task is easy that features of the given image pair can be employed directly to measure distance since no viewpoint variation needs to be considered. Unfortunately, in most realistic scenarios, surveillance

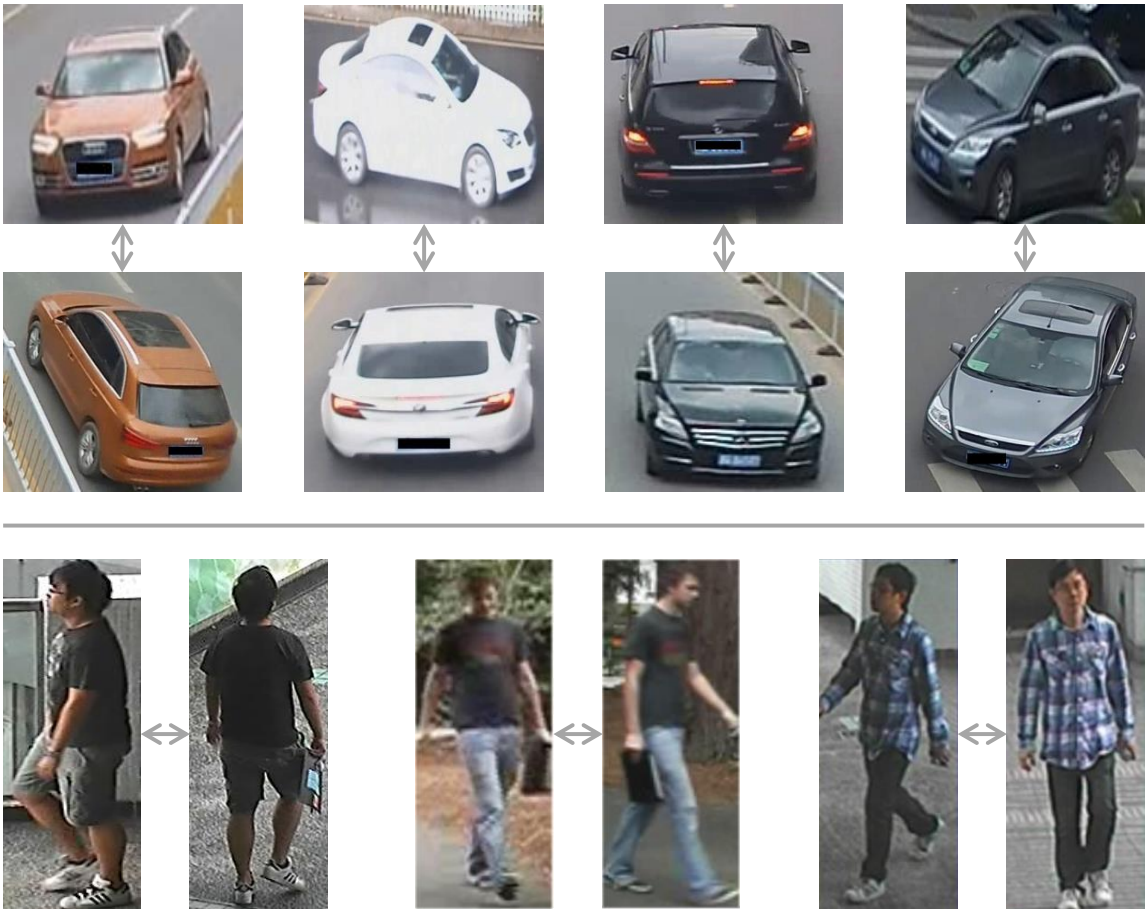


Fig. 4.1 Comparison of vehicle and person re-ID. The variation of visual pattern across different views of a vehicle is much larger than that of a person.

cameras installed at different positions cannot guarantee each vehicle captured is in the same viewpoint. The problem goes difficult for the multi-view vehicle re-ID. Most works [99, 145, 126] also mentioned the challenges thus preferred to exploit spatial and temporal data as auxiliary information to largely reduce the gallery set size then improve the re-ID performance. Since the access of spatial and temporal data is restricted, we aim to address the problem by exploring more general vision-based methods.

Learning distance metrics directly based on features across different viewpoints will make the model confused. Our proposed framework to solve the multi-view vehicle re-ID task is demonstrated in Figure 4.2. We aim to infer images in different viewpoints of the input vehicle with only one captured view, thus distance metrics can be better learned on the generated multi-view feature space which is viewpoint-invariant. In this chapter, we propose two methods of cross-view image generation for vehicle re-ID. One adopts a convolutional

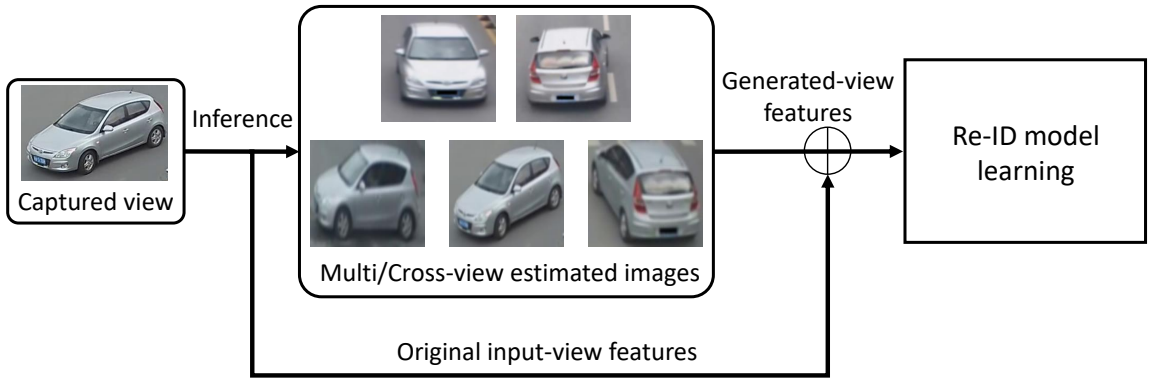


Fig. 4.2 A sketch of our proposed multi-view vehicle re-ID framework. Cross-view images are generated based on the input image, thus distance metric learning can be conducted on the viewpoint-invariant multi-view feature space rather than the single-view one.

encoder-decoder architecture and the other one is built on a conditional generative adversarial network. Our works make the following contributions:

* We present a Spatially Concatenated ConvNet (SCCN) to learn transformations across different viewpoints of a vehicle separately, and then spatially concatenate all the feature maps and map them to a global feature representation to learn the distance metrics.

* A novel deep Cross-View Generative Adversarial Network (XVGAN) is proposed for generating cross-view vehicle images from an input view. Moreover, the model is extended to solve the multi-view vehicle re-ID problem.

* More reasonable distance metrics can be learned on a viewpoint-invariant feature space rather than spaces for cross-view modalities.

* Extensive experiments are carried out to show the superiority of the proposed SCCN and XVGAN on both the vehicle image generation quality and re-ID performance.

4.2 Related Work

4.2.1 Vehicle Re-identification

Most typical vehicle re-ID algorithms are usually based on license plate recognition or multi-sensor fusion. Theoretically, license plate-based re-ID methods [80, 84] should be highly stable and accurate due to the uniqueness of license number. However, license plates are usually not clear enough (e.g. low-resolution and occluded) to identify vehicles in most cases, except some special areas such as toll station where equipped with HD cameras. In addition, license plate is only visible in the front and rear views of a vehicle, which is

restricted to be widely applied to realistic scenarios that a vehicle can appear in arbitrary viewpoints to a surveillance camera. Sensor-based approaches are also well explored for vehicle re-ID. Multi-detector fusion [132] is proved to yield better results than a single detector. Wireless magnetic sensors are adopted in [78] to help to estimate arterial travel time. In [137], vehicle detector waveform is introduced and analyzed in vehicle re-ID. Although sensor-based methods can also achieve satisfactory results, the high costs of professional sensors are not preferred in practical applications. Therefore, the cheaper appearance-based approaches are more potential and deserved to be explored.

Some purely vision-based vehicle re-ID methods and datasets have been proposed in the past two years. Deep relative distance learning [94], adopting coupled clusters loss and mixed difference network structure, is designed for learning the difference between similar vehicles. A new VehicleID dataset is introduced, which contains 10,319 vehicle models. In [158], a vehicle is first extracted by a 3D bounding-box and then warped into a plane as a normalized image. Finally, HOG and color histogram features are computed and linear SVM is learned to match the vehicles. A new vehicle dataset BoxCars which consists of 21,250 different vehicles, is proposed as well. However, the VehicleID dataset mainly studies the front and rear viewpoints, while the BoxCars dataset limitedly contains the front-side and rear-side view. These vehicle re-ID methods only consider some specific views which cannot be applied to realistic applications. Moreover, [98, 99] introduces the VeRi dataset which contains 776 vehicles over 50,000 images with more views and employs visual feature, license plate and spatial-temporal information to explore the re-ID problem. Kanaci *et al.* [68] proposed a cross-level vehicle recognition method exploiting the strong capacity of a Siamese deep model to implement fine-grained re-ID. However, they only explored the front and rear viewpoints. More recently, Wang *et al.* [145] proposed an orientation invariant feature embedding method and adopted spatial-temporal regularization to refine the results. Shen *et al.* [126] mainly focused on the spatio-temporal relations between vehicle images to predict path proposals and the results can be highly improved.

4.2.2 Image Generation

A basic interest in image generation networks is to adopt deep convolutional neural networks (CNNs) that building convolutional layers as image encoders and functioning deconvolutional layers as decoders. Such a general framework has usually been employed to construct view synthesis models. Tatarchenko *et al.* [136] presented a deep CNN to infer multi-view images of 3D models from single image input. Alternatively, rather than generating RGB images of the target view, appearance flow vector is focused in [170]. Although the backbone of the model is still based on a convolutional encoder-decoder fashion, the model does not

need to generate all the pixels from scratch but copy some of them from the input image. Moreover, Park *et al.* [113] proposed a transformation-grounded image generation network that introduces symmetry-aware visibility map to enhance previous models. The visibility map encodes the viewpoint relationship between the input and target views. All these models are mainly optimized using reconstruction loss and evaluated on the 3D car and chairs datasets and achieved promising results.

In addition to the encoder-decoder architectures for intuitively addressing image generation, generative modeling is more worth to be studied. Generative Adversarial Network (GAN) and Variational AutoEncoder (VAE) are the most two successful generative models. The original GAN [44] is proposed with a deconvolutional network for generating images from the noise and a convolutional network for discriminating real or fake samples. DCGAN [117], InfoGAN [18], AC-GAN [110] and WGAN [3] are all excellent follow-up works to investigate conditional GAN variants. Moreover, many researchers have extended GAN to implement different image generation tasks. Reed *et al.* [119] proposed a text-to-image GAN to synthesize realistic images from text descriptions. Image translation is another successful application which gains much improvement by GANs such as [63, 174]. Alternatively, VAE [72] aims to optimize the encoder and decoder networks by minimizing KL divergence between the generated images and the posterior distribution. Gregor *et al.* [49] introduced a deep recurrent attentive neural network for image generation which combines a spatial attention model with a sequential variational auto-encoder framework. In [151], visual attributes are used as conditions to generate images. The proposed model is a layered generative model with disentangled latent variables and optimized by a variational auto-encoder.

4.3 Spatially Concatenated ConvNet for View Estimation

Given a query vehicle image and a set of gallery images, one method is to synthesize images of different viewpoints for each single-view input image and fuse all the feature maps to a global feature which can be considered as a strong descriptive representation containing all-view information. In detail, we can first extract features of the original vehicle image that only contains one-view visual content. Then, transformation models can be learned to estimate images of the vehicle in other viewpoints. Finally, we can fuse all the feature maps in different views, and adopt such final representations to do the distance metric learning for re-ID. A brief overview of this idea is illustrated in Figure 4.3.

In this section, we present our Spatially Concatenated ConvNet (SCCN) which is an end-to-end model. The SCCN, a pure CNN architecture, learns non-linear transformations across different viewpoints of vehicles by separate convolutional and deconvolutional layer

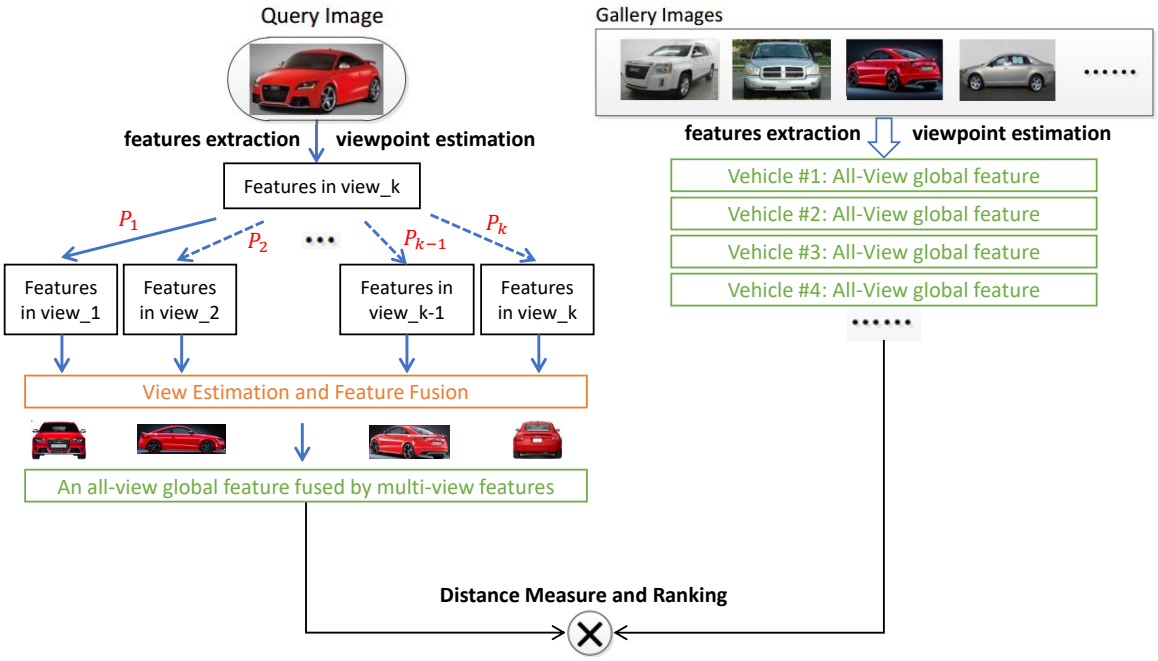


Fig. 4.3 An overview of the SCCN. We aim to infer multiple viewpoints' images of a vehicle from only one visible view and exploit their feature maps to learn the final re-ID model.

sets, and then spatially concatenate the feature maps of all viewpoints to further obtain high-level global features and measure distances with other vehicles.

4.3.1 Problem Formulation

The problem formulation of vehicle re-ID can be similar with that of person re-ID. Define a pair of images $(\mathbf{I}_i^{v_i}, \mathbf{I}_j^{v_j})$ and their corresponding binary label $\mathbf{s}_{ij}$, where v_i and v_j denote the two input images' viewpoint. If $\mathbf{I}_i$ and $\mathbf{I}_j$ are two views from the same vehicle, then $\mathbf{s}_{ij} = 1$, while $\mathbf{s}_{ij} = 0$ if they are from different vehicles. For each single-view input image $\mathbf{I}^v$, we aim to map it to a multi-view representations $\mathbf{f}$ by the following function:

$$\mathbf{f} = G(\{\mathbf{x}_k\}_{k=1}^{k=V}), \ \textit{where} \ \{\mathbf{x}_k\}_{k=1}^{k=V} = T(\mathbf{I}^v). \tag{4.1}$$

The operator $T(\cdot)$ denotes the transformations from the input view image $\mathbf{I}^v$ to the inferred hidden-view features $\{\mathbf{x}_k\}_{k=1}^{k=V}$, where V is our defined number of viewpoints. The operator $G(\cdot)$ is a further mapping process to fuse features $\{\mathbf{x}_k\}_{k=1}^{k=V}$ of different viewpoints into a global multi-view feature $\mathbf{f}$ of the input vehicle.

After modeling $\mathbf{f}$, we aim to minimize a loss function $\mathscr{L}_{reid}$ to shorten the distance between $\mathbf{f}_i$ and $\mathbf{f}_j$ when $\mathbf{s}_{ij} = 1$ and maximize that when $\mathbf{s}_{ij} = 0$ by adopting the contrastive loss [54] as follows:

$$\mathscr{L}_{\text{reid}} = \frac{1}{2N} \sum_{n=1}^{N} [\mathbf{s}_{ij} \|\mathbf{f}_i - \mathbf{f}_j\|_2^2 + (1 - \mathbf{s}_{ij}) \max(0, m^2 - \|\mathbf{f}_i - \mathbf{f}_j\|_2^2)]. \tag{4.2}$$

Where N is the number of samples. The first term is to penalize the distance when a positive pair of vehicle images is too separated, while the second term penalizes a negative pair which is closer than a margin m. In addition, no penalty happens if the distance between a negative pair $\mathbf{f}_i$ and $\mathbf{f}_j$ is already larger than m (The input $\mathbf{f}$ is normalized and m is set as 1.0 in our experiments). After the model is well trained, the Euclidean distances between the query vehicle and the gallery ones can be simply computed based on the top level features $\mathbf{f}$ to get the final ranking. Therefore, the most significant factor to achieve high-performance re-ID is how to design and learn the $T(\cdot)$ and $G(\cdot)$.

4.3.2 Network Architecture

The Spatially Concatenated ConvNet (SCCN) combines two functional parts into one end-to-end model. Fig. 4.4 illustrates an overview of the architecture of SCCN. The first part of SCCN aims to learn a set of feature maps containing information of the input vehicle in V viewpoints from the input visible view. Formally, we implement the $T(\cdot)$ by:

$$\{\mathbf{x}_k\}_{k=1}^{k=V} = P_k(\mathbf{I}^v), \tag{4.3}$$

Where P_k is the transformation function from the original input view v to the k^{th} defined view. After spatially concatenation of $\{\mathbf{x}_k\}_{k=1}^{k=V}$, the second part follows a Siamese-like network, taking $\{\mathbf{x}_k\}_{k=1}^{k=V}$ as inputs to learn feature fusion by $G(\cdot)$ for measuring distance. These two parts can be easily combined into one end-to-end model and optimized simultaneously.

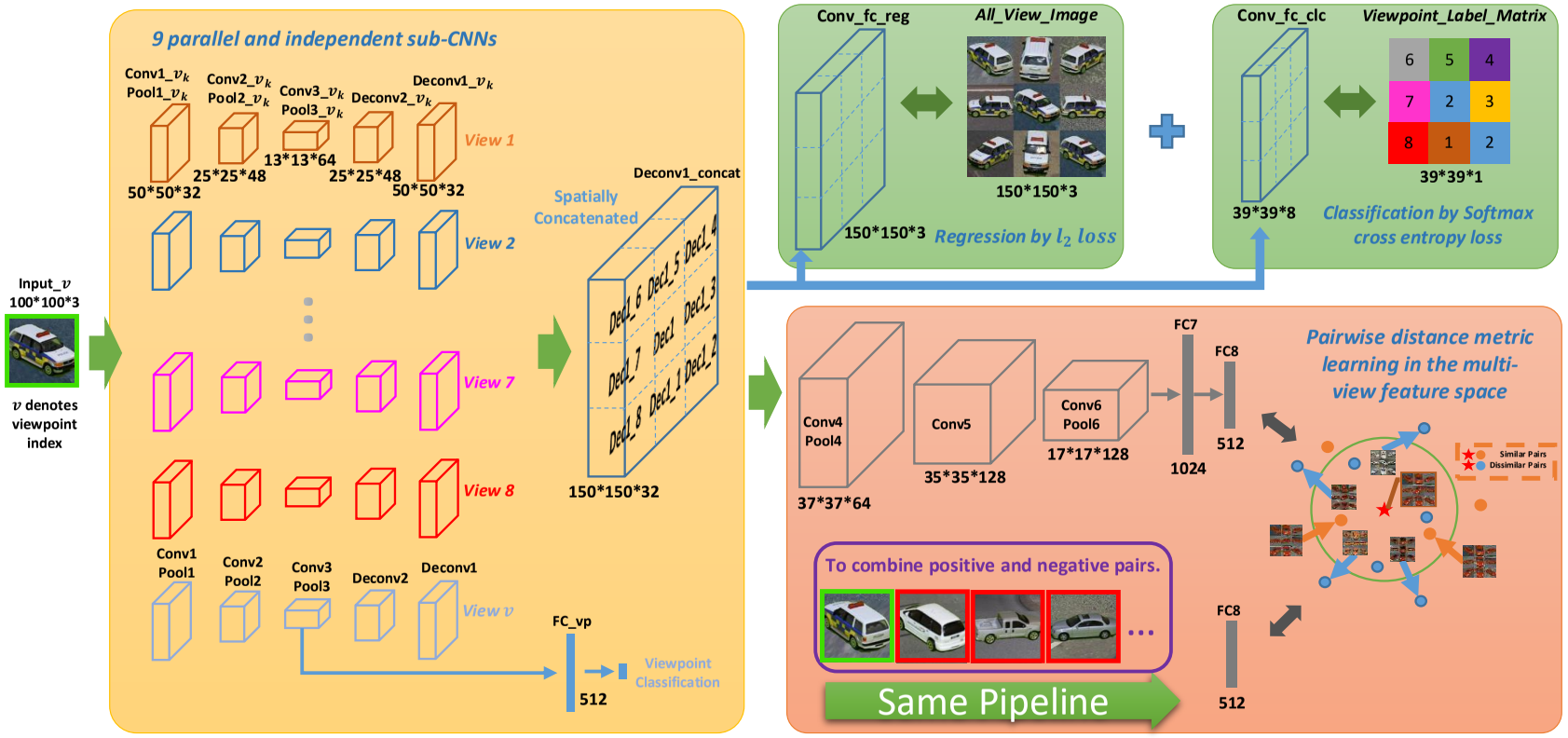


Fig. 4.4 An overview of the architecture of SCCN which consists of two sub-networks shaded in the color of orange and pink. The first part employs nine parallel sets of convolutional and deconvolutional layers to infer transformations between the input visible view and other hidden views of a vehicle. Training the spatially concatenated *Deconv*1*_concat* layer is first supervised by two kinds of ground truth label matrices shaded in the color of green, which are *All_View_Image* and *Viewpoint_Label_Matrix* for regression and classification, respectively. The second sub-network further adopts convolutional and fully-connected layers to learn non-linear mappings from the concatenated feature maps to a global multi-view feature representation of the input vehicle. (Better viewed in color.)

Layer	kernel size	Output number	pad	stride
Conv1_v_k & Conv1	5×5	32	2	1
Pool1_v_k & Pool1	2×2	32	0	2
Conv2_v_k & Conv2	5×5	48	2	1
Pool2_v_k & Pool2	2×2	48	0	2
Conv3_v_k & Conv3	3×3	64	1	1
Pool3_v_k & Pool3	2×2	64	0	2
Deconv2_v_k & Deconv2	3×3	48	1	2
Deconv1_v_k & Deconv1	4×4	32	1	2
FC_vp	-	512	-	-
Conv_fc_reg	5×5	3	2	1
Conv_fc_clc	5×5	8	2	4
Conv4	5×5	64	0	2
Pool4	2×2	64	0	2
Conv5	3×3	128	0	1
Conv6	3×3	128	0	1
Pool6	2×2	128	0	2
FC7	-	1024	-	-
FC8	-	512	-	-

Table 4.1 Parameter settings of the SCCN. Leaky-ReLU activation is set after each convolutional and fully-connected layer.

In SCCN, we define V as 8. The inputs of the SCCN are pairwise vehicle images whose sizes are rescaled to $100 \times 100 \times 3$. Based on the only visible view of a vehicle, the first five sets of convolutional and deconvolutional layers *Conv1_v_k*, *Conv2_v_k*, *Conv3_v_k*, *Deconv2_v_k* and *Deconv1_v_k* are designed as 8 parallel and independent sub-networks to implement non-linear transformation for learning 8 pre-defined v_k views' information. The deconvolution layers act as upsampling and convolution. In addition, another individual sub-network by *Conv1*, *Conv2* and *Conv3*, is used for learning the original viewpoint index and increasing the proportion of the original view in the concatenated all-view feature maps. As shown in Figure 4.4, the *Deconv1_concat* layer, which is spatially concatenated by *Deconv1_v_k* and *Deconv1*, is connected to a multi-loss supervision to learn the transformations across multiple views. Thus, to some extent, the obtained *Deconv1_concat* can infer latent information of a vehicle in all views from only one visible view and can be regarded as a global feature map to describe a vehicle comprehensively. To build an end-to-end architecture, the *Deconv1_concat* further goes through the Siamese-like network to learn the high-level global features by pairwise comparison for computing the distances with other vehicles from different views. The entire network of two paths in the Siamese architecture adopts the same pipeline sharing all the parameters. The model parameter details of the SCCN are listed in Table 4.1. Leaky-

ReLU activations are configured with the slope value of 0.1 after all the convolution and fully-connection layers.

4.3.3 Cross-View Transformation Multi-Loss

In order to train the first five sets of convolutional and deconvolutional layers to extract latent features in hidden views of a vehicle based on only one visible view, we employ a multi-loss to supervise the network with all views' information of a vehicle. First, for each vehicle image in the training set, we generate an *All_View_Image* by placing the original image in the center and the corresponding 8 views around it (see Figure 4.4). This kind of *All_View_Image* is normalized between 0 and 1, and used as the regression label matrix $\mathbf{Y}_{reg}$ for learning by Euclidean loss. In addition, we make a corresponding *Viewpoint_Label_Matrix* $\mathbf{Y}_{clc}$, which is used for viewpoint classification by a softmax function, to enhance the semantic information for learning cross-view transformations. Therefore, after spatial concatenation to the *Deconv*1*_concat* layer, the *Conv_fc_reg* and *Conv_fc_clc* layers are configured for the real-valued predictions $\widehat{\mathbf{Y}}_{\text{reg}}$ and softmax output class probabilities $\widehat{\mathbf{Y}}_{\text{clc}}$, respectively. The multi-loss function of the cross-view transformation is defined as follow:

$$\mathscr{L}_{\text{crossview}} = \frac{1}{2N}\sum_{n=1}^{N}\|\widehat{\mathbf{Y}}_{\text{reg}} - \mathbf{Y}_{\text{reg}}\|_2^2 - \frac{1}{N}\sum_{n=1}^{N}\log(\widehat{\mathbf{Y}}_{\text{clc}}, \mathbf{Y}_{\text{clc}}), \tag{4.4}$$

Where N is the number of samples. In the training phase, the input view of a vehicle is one of the 8 viewpoints, which means the original view usually constitutes two-ninths of information in the concatenated *Deconv*1*_concat*. In other words, we still expect the original view image can be slightly dominant because its visual pattern is intrinsic and lossless.

After learning the *Deconv*1*_concat* layer, which can infer more viewpoints' information of a vehicle, the following network of the SCCN is designed for discriminatively learning the distance metrics between positive or negative vehicle pairs. The 512-*dim FC*8 layer is adopted as the final multi-view feature which is connected with $\mathscr{L}_{reid}$. The multi-view feature is a viewpoint-invariant representation on which distance metrics can be better learned. Moreover, $\mathscr{L}_{\text{crossview}}$ and $\mathscr{L}_{reid}$ are optimized adopting the same loss weight.

4.3.4 Optimization

We first pre-trained the SCCN using the large-scale multi-view RGB-D Object Dataset proposed in [79], since the diversity of this object dataset is abundant for better learning our models with respect to transformations across multi-views of a 3D object. The RGB-D

Object Dataset contains 300 common household objects' images captured for one whole rotation. We only use the RGB data, and averagely sample 8 views for each object at different heights. After 40 epochs pre-training, we adopt the training set of our Toy Car RE-ID dataset to fine-tune our models.

According to some empirical parameters, we specifically set the best-performed value of momentum of 0.9 and weight decay of 0.0005 for training the SCCN. The learning rate is initially scheduled as 5×10^{-4}, and then reduced by 10 when the loss stops decreasing. The training is iterated by 20 epochs.

4.3.5 Experiments and Results

Datasets and Evaluation Protocols

Toy Car RE-ID Dataset. To better explore the multi-view re-ID problem, our motivation is to build a vehicle dataset where densely sampled viewpoints are available for each vehicle since no such kind of dataset exists in this research area. However, collecting an all-view vehicle dataset for re-ID in realistic surveillance is difficult due to restricted access of recordings and likelihood of certain views not being available. To address these issues, we collect a toy car dataset which contains many common vehicle types such as sedan, SUV, hatchback, van, and pickup, of different colors. As we expect to reduce the gap in appearance between toy cars and real cars, we mainly select those metal toy cars as real as possible to construct the dataset. In addition, we provide lighting to simulate illumination by the sun.

This vehicle re-ID dataset consists of 200 different models of toy car. As illustrated in Fig. 4.5, we first adopted a rotation stage to collect sequences of vehicles as they rotated by 360 degrees. We set cameras at three angles: 30°, 60° and 90° to capture data with different altitudes. In each angle, we averagely sampled 50 viewpoints and cropped all the vehicles to generate the raw dataset containing 30,000 images in total. In the second phase, we replace the green background with random road patterns to synthesize our final toy car dataset. In this chapter, we only use data in 30° and 60° angles to evaluate our models. We combine these two angles to make our models better trained with the compatibility of learning vehicles in more flexible vertical angles, since a vehicle far from a camera and one close to a camera are captured in different angles in the real world. The details of the use of data and split of training and test sets are explained in the experiments section. The whole dataset containing 30°, 60° and 90° can be also used for studying other tasks such as UAV vehicle detection. We don't include 90° in our experiments because such an angle is not realistic for vehicle re-ID on the ground.

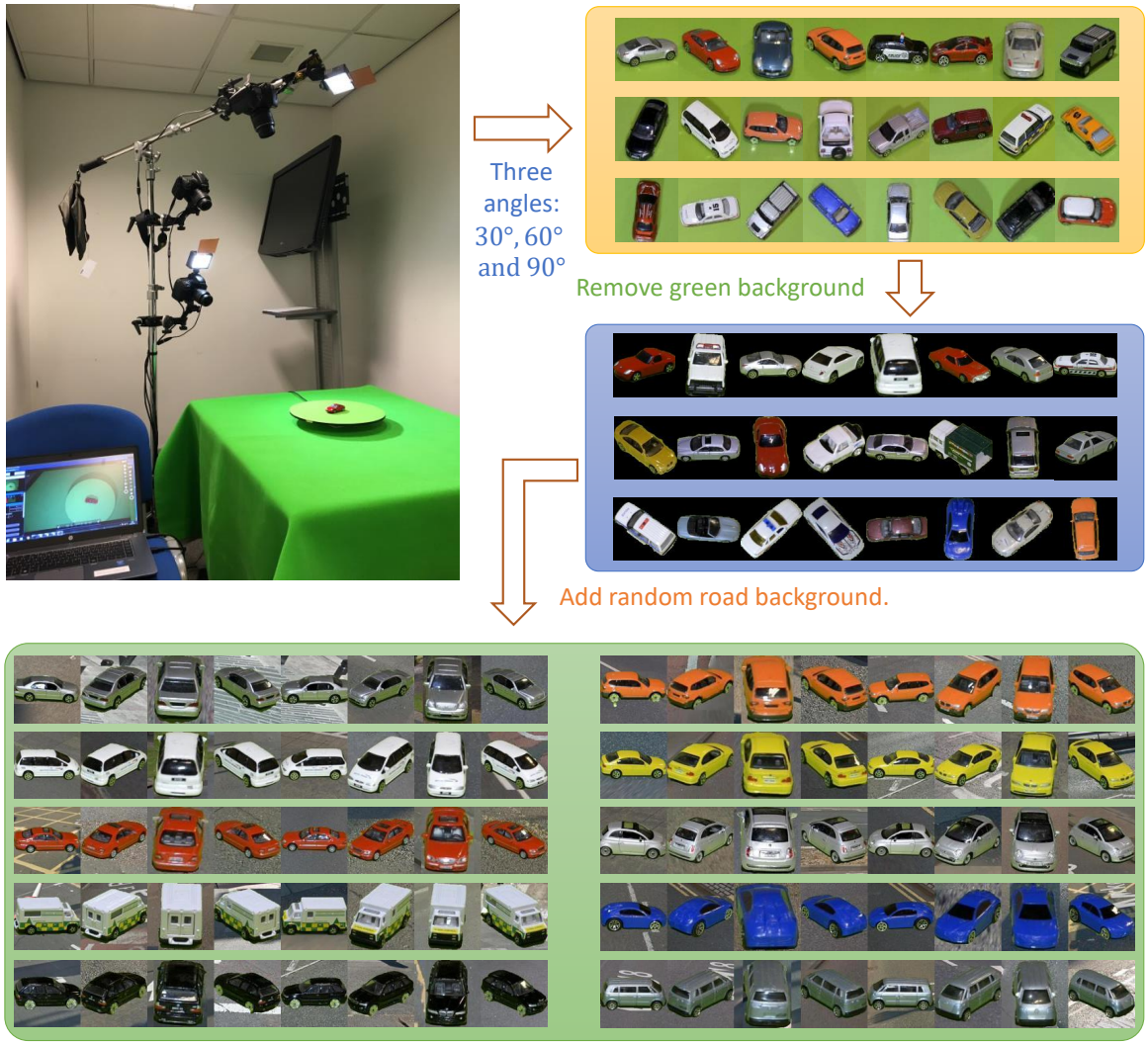


Fig. 4.5 Collection of the Toy Car RE-ID Dataset. Three angles 30°, 60° and 90°, and 50 views in each angle are available to be used. The picture at the bottom gives a glance of the final synthesized data in angle 30°with 8 views for each vehicle.

We select 8 main viewpoints of each vehicle for evaluating SCCN. Images in angles 30°and 60°are mixed to make our models invariant for vertical angles' variation to some extent. Since the majority of our toy car data is captured under the same illumination, we augment the data by adjusting the value channel into three different levels in HSV space, which has been evaluated to improve the final results. In the experiments, we split the dataset into 150 vehicles for training and 50 vehicles for testing. To train our models, we randomly compose 200 view pairs for each vehicle and finally generate 30,000 positive pairs (150 vehicles $\times$ 200 different combinations in viewpoint, angle and illumination). Besides, 60,000 negative pairs are also randomly composed of different vehicles. In the test phase, any arbitrary viewpoint in the dataset can be used (not limited to 8 or 16 viewpoints). We randomly select one image from 100 views (50 viewpoints $\times$ 2 angles) of each vehicle into the gallery set, which was repeated 10 times in our experiments to obtain the final CMC curves.

Multi-view Car Dataset. To make our methods more convincing and show their generalizability, we also evaluate our models on real car datasets. Multi-view Car dataset [111] is the only real vehicle dataset that each car has all-view images. However, it contains sequences of only 20 vehicles rotated by 360 degrees. We use it as a small test set for evaluating all models trained on our Toy Car RE-ID dataset. Since the backgrounds of all images for one vehicle are the same but different between vehicles in this dataset, we tightly crop the vehicle in each image and add random backgrounds to avoid features being dominated by background patterns. In our experiments, we averagely sample 16 viewpoints of each vehicle and similarly select a random one of them into the gallery set.

VehicleID Dataset. VehicleID [94] is a large-scale realistic surveillance dataset, but only contains images of two viewpoints: the front and rear view. It consists of 26267 vehicles in total and is split into 13134 vehicles for training and 13133 vehicles for testing. We strictly follow their evaluation protocol, even though it is not fair for our multi-view methods since many vehicles in the dataset only contain images in one viewpoint. In the test set, we randomly take one image of each vehicle into the gallery set (gallery size = 800). Other images are used for querying. The final results are obtained by repeating the experiments 10 times.

VeRi Dataset. The VeRi dataset [99] contains 776 vehicles (about 9000 tracks) with more varied viewpoints. According to [99], the dataset is split into 576 vehicles for training and 200 vehicles for a test. A newly defined evaluation method, mean average precision (mAP) for re-ID, is also introduced in [99], and image-to-track search is conducted instead of conventional image-to-image search. Moreover, since some vehicles in this dataset have more available viewpoints but not all, we carefully select 255 vehicles containing our defined

Table 4.2 Evaluation (%) of effectiveness of each component proposed in the SCCN. The left part studies the proposed multi-view inference. The right part studies the architecture of the SCCN.

		Multi-view inference		Architecture of the SCCN			
Datasets	Methods	SCCN	Single-View Feature	SCCN-w/o-Deconv1	SCCN-w/o-VLM	SCCN-w/o-AVI	One-CNN
ToyCar	r=1	41.69	21.36	38.26	35.39	23.81	29.64
	r=5	80.47	62.04	78.54	75.66	65.30	69.82
	r=20	100.00	93.57	99.81	97.23	94.11	95.30
Multi-View	r=1	66.14	38.24	64.92	60.73	37.92	46.54
	r=5	97.37	77.91	96.13	92.71	78.65	84.11
	r=10	100.00	98.16	100.00	98.05	97.95	97.62
VehicleID	r=1	45.14	32.59	44.39	42.29	29.68	34.36
	r=10	69.07	42.66	68.50	67.41	38.54	50.20
	r=50	85.37	57.01	84.82	83.26	55.22	67.15
VeRi	mAP	15.39	12.96	15.28	15.02	9.56	13.79
	r=1	44.85	35.29	44.59	43.91	26.89	38.64
	r=5	58.61	49.30	58.47	58.13	41.77	53.15
	r=20	71.73	64.15	71.62	71.07	59.20	66.99

8 different viewpoints in the training set to finetune our model, which refers to SCCN-Ft in experiments, and then evaluate on the same whole test set.

Ablation Studies

Hidden View Generation by SCCN. Before evaluating re-ID performance, some generated examples of the *Conv_fc_reg* are visualized in Fig. 4.6. The upper four sample pairs are results by the model trained on our Toy Car RE-ID dataset. The bottom four sample pairs are results by the model fine-tuned on the VeRi dataset. The SCCN model infers eight different views as well as the original visible one for each test vehicle, and can moderately filter useless background. From inferred samples, we can observe that most of the shape and color information could be successfully generated. Although the inferred views look blurry, the learned features of these views can already contribute to the final multi-view vehicle re-ID accuracies.

Effect of Multi-view Inference. The primary contribution we need to investigate is the effectiveness of the multi-view feature inference for vehicle re-ID. In this baseline, we configure $\mathscr{L}_{reid}$ after *FC_vp* layer to learn distance metrics only based on the input one-view features, which refers to Single-View Feature in Table 4.2.

As shown in the left part of Table 4.2, SCCN can make consistently significant improvements compared with the Single-View Feature baseline on four vehicle datasets. Particularly for the Toy Car RE-ID and Multi-View Car datasets where each vehicle has images of densely sampled viewpoints, the rank-1 accuracies can be increased by over 20% by our model. To better demonstrate the effectiveness of multi-view inference, we visualize the multi-view feature embeddings compared with the Single-View Feature ones directly extracted from

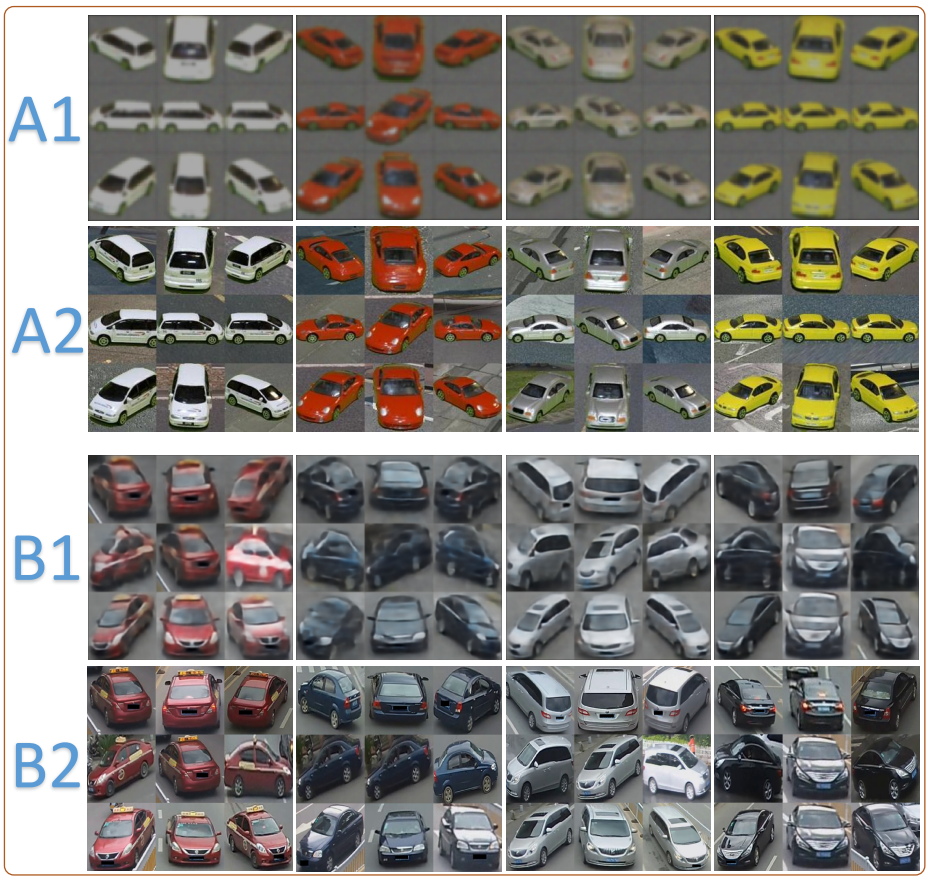
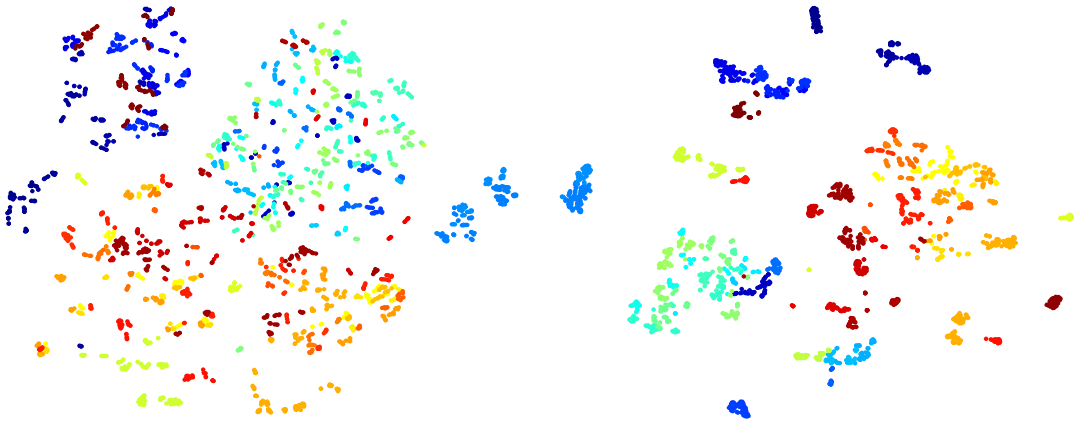


Fig. 4.6 Visualization examples of the *Conv_fc_reg* inference. Row A1 and B1 show samples inferred from the only one input view. The *All_View_Image* ground truths are compared in Row B1 and B2.



(a) Single-View Feature (b) Multi-View Feature

Fig. 4.7 t-SNE demonstration of multi-view features compared with single-view features of samples from 50 test vehicles in the VeRi dataset. It shows our learned multi-view features are more viewpoint-invariant.

the input view. We select VeRi dataset to show the results since it is a realistic vehicle dataset containing different viewpoints' images. In Fig. 4.7, images of 50 vehicles in the test set of VeRi dataset are distributed in the 2-D space. It shows samples of the same vehicle in different viewpoints are scattered by the Single-View Feature, but better clustered after multi-view feature inference.

Studies on SCCN. We first evaluate each component proposed in SCCN with the following baselines.

SCCN-w/o-Deconv1. (drop the sub-CNN for *Deconv*1**)** The branch of producing *Deconv*1 is proposed to increase the proportion of original input view in the concatenated feature maps since in the test phase the viewpoint of the input vehicle can be arbitrary which is not limited to the defined eight viewpoints for training. To explore its effectiveness, we compare it with a baseline that directly concatenates two same sub-CNNs for the input viewpoint in the concatenated *Deconv*1*_concat*.

SCCN-w/o-VLM. (drop the *Viewpoint_Label_Matrix* **(VLM))** The layer *Conv_fc_clc* supervised by the *Viewpoint_Label_Matrix* is auxiliary to provide more semantic viewpoint labels for better training the SCCN. It contributes to faster and more stable convergence for getting incremental accuracies. In this baseline, we drop it for evaluating its effectiveness.

SCCN-w/o-AVI. (drop the *All_View_Image* **(AVI))** We design the layer *Conv_fc_reg* for learning transformations across different viewpoints optimized by l_2 loss with *All_View_Image*. This is the most essential part of modeling vision pattern relations between the input viewpoint and the target viewpoint. If we drop this supervision, training the SCCN with only

discrete labels of *Viewpoint_Label_Matrix* does not make sense to transform the input view to different views, thus cannot learn effective multi-view features.

One-CNN (for learning multi-view image generation). To design view-specific parallel networks separately for learning transformations for each viewpoint is because each sub-CNN only needs to focus on optimizing parameters for one viewpoint which achieves better performance. To explore the necessity of this design, we set a One-CNN model as a baseline. We adopt one encoder-decoder structure with the number of channels increased by eight times for each layer compared with one branch in the SCCN. The pre-computed average feature from *FC_vp* for each viewpoint is replicated spatially and concatenated in depth with the bottleneck *Conv* layer as the viewpoint conditional embedding. Other parts are kept same as the SCCN.

The right part of Table 4.2 illustrates that adding the additional sub-CNN for modeling the original input viewpoint can have a slight increase for the final result compared with SCCN-w/o-Deconv1. Moreover, with the training by auxiliary *Viewpoint_Label_Matrix*, we can moderately improve our models' performance on all the four datasets. Dropping the indispensable *All_View_Image* will make our model meaningless and degrade the accuracies same or even lower than results by Single-View Feature. Additionally, adopting one CNN to learn transformations across different viewpoint pairs severely decreases the performance although it can reduce the model's complexity.

Effect of Data Augmentation by Illumination Synthesis. To moderately make our models robust for varying lighting conditions in the real world, we conduct data augmentation by illumination synthesis. More specifically, we augment the data by adjusting the value channel into three different levels in HSV space. Some synthesised examples are demonstrated in Figure 4.8. Although it is not a natural way to provide the different lighting environments, our ablation experiment disabling the illumination synthesis has proved that such data augmentation can indeed benefit the final results. As shown in Table 4.3, particularly for the real surveillance datasets VehicleID and VeRi with diverse lighting conditions, our synthesis method can averagely get about 1.0% increase of rank-1 accuracy.

Table 4.3 Rank-1 rate (%) of Data Augmentation by Illumination Synthesis.

Datasets	Illumination Synthesis	No Illumination Synthesis
Toy Car RE-ID	41.69	41.25
Multi-View Car	66.14	65.95
VehicleID	45.14	43.92
VeRi	44.85	43.96

Fig. 4.8 Examples of illumination synthesis on three different levels.

Comparisons with State-of-the-arts

We compared our proposed methods with state-of-the-art person and vehicle re-ID methods. Local Maximal Occurrence Representation (LOMO) [88] is a hand-crafted local feature first proposed for person re-ID. It aims to address the problem against viewpoint and illumination variations. Moreover, XQDA [88], LDNS [159] and MLAPG [90] are used for learning subspaces based on the LOMO features. KISSME [75] is a distance metric learning method from equivalence constraints and DVDL [69] aims to learn viewpoint invariant dictionaries to discriminate different people. Moreover, we train deep models, which strictly follow methods proposed in DeepReID [85], Improved deep [1], SIR + CIR [143], DGD [149] and DRDL [94]. One significant difference between person and vehicle re-ID is that the aspect ratio of a cropped image of human is usually less than 1:2, while a vehicle image is suitable with 1:1. Thus, for training those deep models originally designed for person, we first vertically concatenate the RGB and LAB images of each vehicle and then resize them to the input sizes defined in their papers to avoid the distortion of vehicle shapes.

Fig. 4.9 illustrates the CMC curves of different re-ID methods performed on the Toy Car RE-ID, Multi-View Car, and VehicleID datasets. Our model evaluated on these three datasets are only trained using the proposed Toy Car RE-ID datasets. It can be observed that our method can achieve great improvements compared with state-of-the-art re-ID algorithms, which validates the feasibility to use toy car images instead of real vehicle images for training. For the VehicleID dataset, our model is beaten by certain previous deep models, since a large amount of training data is available and only two viewpoints are required to be matched in the test set. We train all the state-of-the-arts using the combination of our Toy Car dataset and the VehicleID dataset. However, our SCCN still adopts the relatively small Toy Car RE-ID dataset for training.

Table 4.4 Comparisons (%) with state-of-the-art re-ID methods.

Datasets	Toy Car RE-ID			Multi-View Car			VehicleID		
Methods	r = 1	r = 5	r = 20	r = 1	r = 5	r = 10	r = 1	r = 10	r = 50
KISSME [75]	6.88	29.61	73.21	17.02	55.18	78.46	35.09	44.69	58.74
LOMO+XQDA [88]	12.80	41.35	82.52	26.48	69.49	87.01	37.01	49.97	66.06
DVDL [69]	17.58	50.36	88.12	35.45	75.19	95.16	36.40	52.35	69.68
LOMO+MLAPG [90]	18.31	56.09	89.56	48.48	83.46	97.49	37.62	57.15	77.58
LOMO+LDNS [159]	27.73	63.92	96.60	37.39	78.17	97.02	35.95	54.30	73.91
DeepReID [85]	25.38	61.09	95.25	46.13	78.26	98.42	37.22	57.92	77.23
Improved Deep [1]	30.05	66.68	96.53	51.04	93.12	100.00	40.14	61.51	82.36
SIR+CIR [143]	32.75	71.85	98.14	53.46	88.19	100.00	47.31	72.12	87.29
DGD [149]	34.16	74.59	99.79	57.13	90.67	100.00	48.33	80.85	93.32
DRDL [94]	33.09	72.52	99.44	58.27	95.36	100.00	**49.92**	**83.23**	**95.44**
SCCN (Ours)	**41.69**	**80.47**	**100.00**	**66.14**	**97.37**	**100.00**	45.14	69.07	85.37

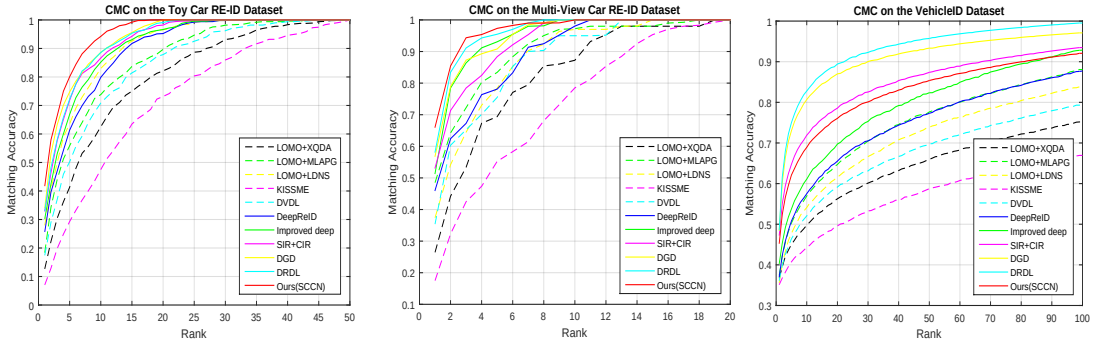


Fig. 4.9 CMC curves comparisons of different re-ID methods on the Toy Car RE-ID, Multi-view Car and VehicleID datasets.

Detailed rank-*r* results are compared in Table 4.4. For the Toy Car RE-ID dataset, the SCCN outperforms the best previous model DGD [149] with a big gap of 7.53% at rank-1. Moreover, we can observe that although those one-view based deep models generally achieve better results than non-deep learning methods, their performance is still much poorer than our proposed multi-view inference. For the Multi-View Car dataset, our method obtains consistent improvements over other algorithms and can almost achieve perfection within the top-5 positions. For the VehicleID datasets, the DRDL [94] obtains the best rank-1 accuracy of 49.92%, which beats our SCCN with 4.78%. Our method does not get superior performance since the advantage of our networks designed for multiple views is not fully exploited in this two-view setting. The aim of SCCN is to learn the networks with transformations across arbitrary-view pairs of a vehicle, and then the models can be evaluated on arbitrary-view re-ID. Thus, our model can be enhanced if large-scale realistic multi-view vehicle training data is available. This point has been proved on the VeRi dataset.

Table 4.5 Evaluation (%) on the VeRi dataset.

Methods	mAP	r=1	r=5	r=20
LOMO [88]	9.03	23.89	40.32	58.61
GoogLeNet [153]	17.58	51.98	66.79	78.77
FACT [99]	18.54	52.35	67.16	79.97
SCCN-Ft (Ours)	**20.13**	**55.46**	**70.02**	**82.94**

Since the VeRi dataset contains more available viewpoints for each vehicle, we fine-tune the SCCN on the VeRi dataset which can significantly improve the results. Moreover, we compare the fine-tuned model with existing re-ID approaches evaluated on this dataset. One is the GoogLeNet fine-tuned on the CompCars dataset [153] can be used for extracting great visual descriptors containing rich semantic features for vehicles. Fusion of Attributes, and Color FeaTures (FACT) [99], consisting of SIFT, Color Name and GoogLeNet features, is proposed to discriminate vehicles in joint domains.

Our method is purely based on vehicle appearances, we do not compare them to those methods adopting license plate and spatio-temporal information in [99]. As shown in Table 4.5, the SCCN-Ft gets 1.59% increase in terms of mAP than that of the FACT [99]. Our model trained using less training data of the VeRi dataset, obtains better results, which strongly illustrates the effectiveness of multi-view inference.

4.4 Cross-View Generative Adversarial Network for Image Generation

Image generation by convolutional generative adversarial networks (GANs), which has obtained breakthrough progress on generating real images, inspires us to generate vehicles in different viewpoints from only one visible view to tackle vehicle re-ID. In this section, we propose another new deep architecture, called *Cross-View Generative Adversarial Network* (XVGAN), to learn the features of vehicle images captured by cameras with disjoint views, and take the features as conditional variables to effectively infer cross-view images. Finally, the features of the original images are combined with the features of generated images in other views to learn distance metrics for vehicle re-ID. Figure 4.10 (a) sketches an overview of the XVGAN which comprises three sub-networks: Classification (C), Generative (G) and Discriminative (D) Nets. C Net can learn intrinsic features of the input vehicle image. Conditioned on these features as well as a pre-computed viewpoint feature of the expected view, G and D Nets aim to generate real images of the same vehicle in other

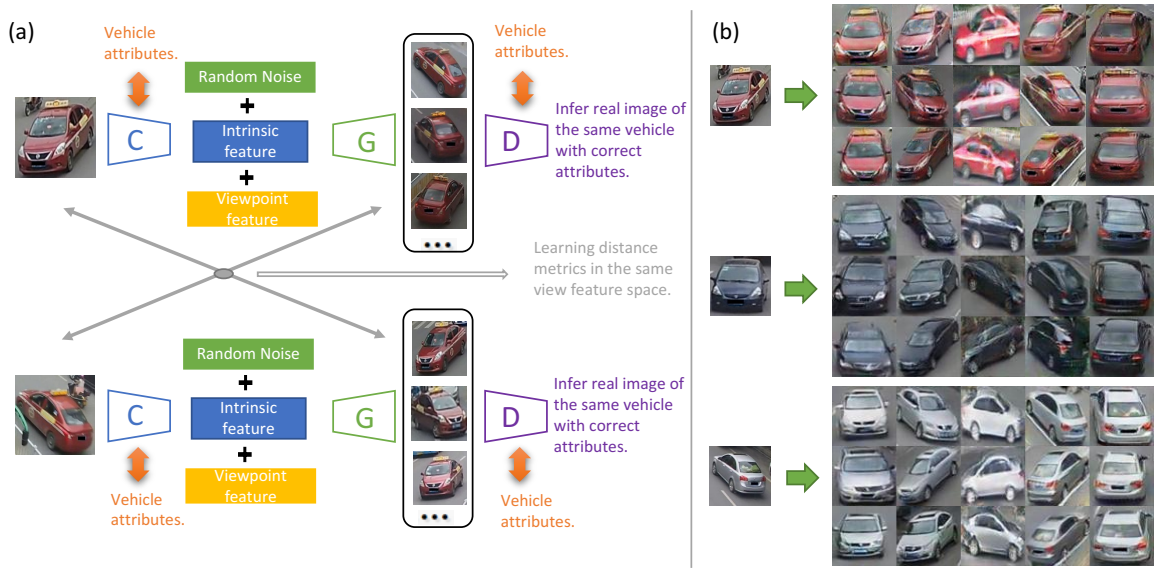


Fig. 4.10 **(a)** Overview of the proposed XVGAN. A Classification Net is first used for learning vehicles' intrinsic features containing model, color and type information. Besides, viewpoint features are also learned. The Generative Net then takes a vehicle's intrinsic feature of the visible view, the average feature of the expected viewpoint and a random noise vector as inputs to infer images of the same vehicle in other views. The Discriminative Net distinguishes real images from synthetic samples while keeping images generated with correct vehicle attributes. Finally, the inferred vehicle images from cross-view pair data contribute to learning distance metrics for re-ID. **(b)** Generated image examples in different viewpoints for the input vehicle.

viewpoints. Examples of the inferred samples by XVGAN are illustrated in Figure 4.10 (b). Our model can successfully generate realistic images in different views of the same vehicle, and contribute to re-ID on two public datasets: VeRi and VehicleID.

4.4.1 Generative Adversarial Nets

GAN is an unsupervised machine learning method, which achieves great success in image generation tasks. It consists of a generative model G and a discriminative model D competing against each other in a two-player min-max game. The generative network takes a latent random vector z from a uniform distribution as input to generate samples. The $p_z(z)$ is expected to converge to a target true data distribution $p_{data}(x)$, where x is a real image. Meanwhile, the discriminative network aims to distinguish the real data from synthesized samples. These two networks are simultaneously optimized by the following problem:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x\sim p_z(z)}[\log(1-D(G(z)))]. \tag{4.5}$$

It has been proved [44] that a global optimum can be obtained when p_G well converges to p_{data}, if G and D have enough capacity. Compared to other generative models, GAN has few restrictions (e.g. no Markov chain and variational bound is needed relative to Boltzmann machines and VAEs). Moreover, since G is very poor in the early training stage and D can easily reject synthesized samples with high confidence, $\log D(G(z))$ is maximized for better training G rather than minimizing $\log(1-D(G(z)))$.

4.4.2 XVGAN

We propose a new cross-view GAN (XVGAN), which consists of three main networks, to generate vehicles across different viewpoints. A Classification Net aims to learn vehicle features including type, color, some unique patterns and the viewpoint information. Conditioned on these features, the Generative Net takes the intrinsic vehicle features of input views, random vectors and central viewpoint features of expected views as inputs to synthesize vehicle images in certain views of the same input vehicles. The Discriminative Net distinguishes the generated samples from the real images and simultaneously tries to match the inferred vehicle images with correct attributes and viewpoints. Finally, the features of original views are concatenated with the features of inferred views to further learn distance metrics for re-ID. The network architecture is illustrated in Figure 4.11. The final learned *ReidFeat* can be directly adopted for measuring distances between vehicles across different views.

Feature learning and viewpoint clustering. Formally, x^A denotes the input vehicle images in camera A. The trunk architecture of the Classification Net consists of 4 convolutional layers (kernel size = 5, padding = 2 and stride = 2) and 2 fully-connected layers. The Leaky-ReLU is set after each layer. Then, we configure two layers for learning the 256-dimensional x^A_{attr} by multi-attributes classification and the 128-dimensional x^A_{vp} by viewpoint classification separately, since we expect viewpoint information of view A weakened in x^A_{attr}, but strengthened in x^A_{vp}. The viewpoints of all vehicles are coarsely categorized into five groups: front, rear, side, front-side, and rear-side. During training the Classification Net, the loss of viewpoint classification can be fast and well converged. Thus, we can easily learn five viewpoints' feature clusters from all the training data by k-means clustering, and compute the feature in the center of each cluster, x^B_{cvp}, as a condition to generate views in camera B.

Matching-aware conditional vehicle GAN. The conditional generator is defined as G: $\mathbb{R}^F \times \mathbb{R}^Z \times \mathbb{R}^V \to \mathbb{R}^I$, where F is the dimension of x^A_{attr}, Z is for random noise, V is for x^B_{cvp} and I is for images. Besides, the discriminator is denoted as D: $\mathbb{R}^I \to \{0,1\} \times \prod L_i$, where

$i = \{1 : ID,\ 2 : Type,\ 3 : Color,\ 4 : Viewpoint\}$. l_i is the attribute label and L_i denotes the range of each label. The optimization of the G and D in Eq. 4.5 can be reformulated as:

$$\begin{aligned} \mathscr{L}_D &= \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] - \sum_{i=1}^{4} \log(D(x), l_i), \\ \mathscr{L}_G &= \mathbb{E}_{x\sim p_z(z);\ x^A_{attr},\ x^B_{cvp}\sim p_{data}(x^A_{attr},\ x^B_{cvp})}[\log(1 - D(G(x^A_{attr}, z, x^B_{cvp})))]. \end{aligned} \tag{4.6}$$

The input of the generator G is the concatenation of the x^A_{attr}, x^B_{cvp} and a random noise prior $z \sim \mathscr{N}(0,1)$. x^A_{attr} can be regarded as an intrinsic feature learned from the original view A without much viewpoint information, while x^B_{cvp} is a central viewpoint feature in the expected view B. A fully-connected layer is set for better fusing the three vectors and then four deconvolutional layers are adopted for generating synthesized vehicle samples. The hyper-parameter settings of the Generative Net are reverse to that of the Classification Net. Moreover, batch normalization is operated on all the layers.

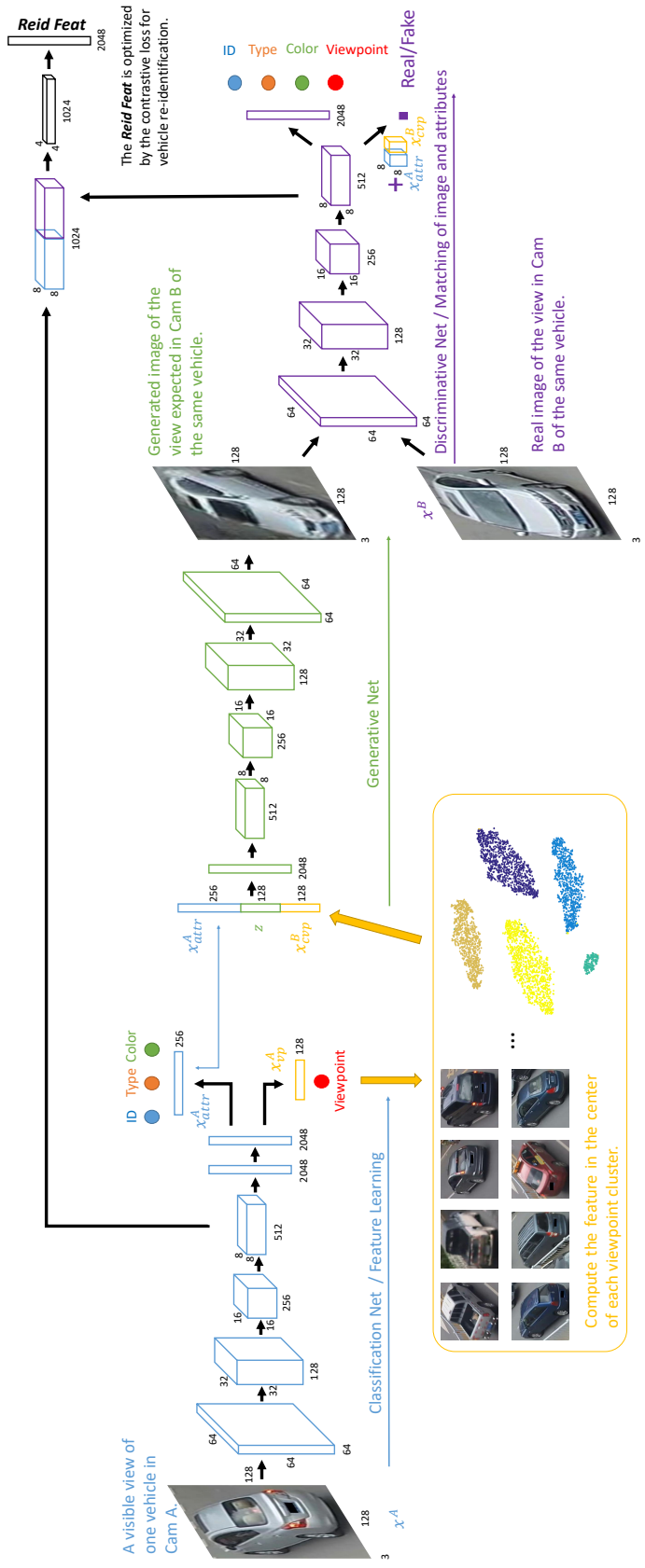


Fig. 4.11 Network Architecture of the XVGAN. The yellow part is for learning the central features of five main viewpoints by k-means clustering. The Discriminative Net is not only for generative adversarial training but also supervised by vehicles' multi-attributes to help the Generative Net infer real images with correct vehicle attributes in certain views. The final convolutional layers in the Classification Net and the Discriminative Net are concatenated in depth for further learning to measure distances by contrastive loss.

The discriminator D takes the generated samples and the real images in view B as inputs. The main trunk of the Discriminator has the similar structure in Classification Net. Meanwhile, to match the inferred images with the same attributes of original vehicles in view A and correct viewpoint in view B, we add a fully-connected layer and simultaneously optimize the whole Discriminative Net by multi-label classification. The viewpoint label is for view B, which is different from the viewpoint in Classification Net for view A. Batch normalization and Leaky-ReLU are adopted for all the layers in the discriminator as well. Moreover, for better optimizing the conditioned G and D, we also replicate the x^A_{attr} and x^B_{cvp} embeddings spatially and do concatenation in depth when the spatial size is 8×8, and perform a 1×1 convolution afterward.

Distance learning for re-ID. In addition to training the XVGAN for image generation, we extend the architecture to simultaneously optimize for the vehicle re-ID problem. Define a pair of images (x^A_q, x^B_p) as positive if they are two views from the same vehicle and (x^A_q, x^B_n) as negative if they are from different vehicles. We feed forward the image pairs to our XVGAN in a Siamese-like way. Take a positive pair as an example, the mapped feature pair (f^A_q, f^B_p) from the last convolutional layer in Classification Net and the inferred $(\hat{f}^B_q, \hat{f}^A_p)$ from the last convolutional layer in Discriminative Net are concatenated as $(f_q = f^A_q + \hat{f}^B_q, f_p = f^B_p + \hat{f}^A_p)$. Conversely, (f_q, f_n) is for negative pairs. Then, f_q, f_p and f_n, including both the features from the original images and the inferred samples, can be further adopted for learning distance metrics by minimizing a contrastive loss $\mathscr{L}_{reid}$ [54] to shorten the distance between the same vehicle and maximize that between different vehicles. Moreover, a convolutional layer (kernel size = 3, padding = 1 and stride = 2) and a fully-connected layer are configured at the end. Our distance metric learning is more reasonable since it is optimized in the same feature manifold by combining the original and generated views.

In the inference phase, the Classification Net first predicts viewpoints of the query vehicle and each candidate in the gallery set. Then, the corresponding central viewpoint features for each other as well as the intrinsic features are adopted to generate cross-view images. Finally, the 2048-dimensional *ReidFeat* is used to measure distance for ranking.

Implementation Settings. The random noise z is set as 128-dimensional, sampled from the uniform distribution. The spatial size of generated images is 128×128. Similar to [117], we adopt the ADAM Optimizer with the learning rate of 0.0002 and the momentum of 0.5. We set the mini-batch size as 64 and trained our model for 500 epochs of the VeRi dataset on a GPU server configured with four GTX TITAN X cards.

4.4.3 Analysis of GAN Compared to Variational Approximations

The reason we build our vehicle image generation network based on GAN is that GAN [43] has fewer disadvantages compared to other generative models. Most generative models follow the principle of maximum likelihood which aims to provide models with estimates of probability distributions. Assume a model is parameterized by θ and a training dataset has N data samples x^i ("training data" usually refers to all the observable samples points on an unknown data distribution), the likelihood is defined as the probability the model distributes over the data: $\prod_{i=1}^{N} p_\theta(x^i;\theta)$. Then, the maximum likelihood is to find the parameters of the model to maximize the likelihood of the training data as:

$$\begin{aligned}\theta^* &= \arg\max_\theta \prod_{i=1}^{N} p_\theta(x^i;\theta) \\ &= \arg\max_\theta \sum_{i=1}^{N} \log p_\theta(x^i;\theta).\end{aligned} \tag{4.7}$$

More specifically, the maximum likelihood process consists of taking observable samples from the data generating distribution to form a training set, then increasing the probability the models assigns to those points, so as to maximize the likelihood of the training data. The maximum likelihood estimation can be also considered as minimizing the KL divergence [77] between the model and the data distribution:

$$\theta^* = \arg\min_\theta D_{KL}(p_{data}(x) \parallel p_\theta(x;\theta)). \tag{4.8}$$

Recent deep generative models can be mainly categorized into explicit density models and implicit density models. The variational approximation is preferred in optimizing explicit density models in which the most successful one is Variational Auto-Encoder (VAE). On the other side, GAN which belongs to implicit density models is designed to be unbiased. The Nash equilibrium aims to optimize GAN by recovering p_{data} as exact as possible. Compared with GAN, variational approximation has some drawbacks. Variational methods used to have a lower bound for their models:

$$\mathscr{L}(x;\theta) \leq \log p_\theta(x;\theta). \tag{4.9}$$

The maximization of $\mathscr{L}$ should be higher than that of a pre-defined $\mathscr{L}$ which is usually computationally tractable. The main drawback of these methods is that when either the prior distribution or the approximate posterior distribution is too weak, the gap between the true likelihood and $\mathscr{L}$ cannot make the p_θ learning exactly according to the true p_{data}. We did exploration to design VAE-based model for vehicle image generation, but the result is not as good as that by XVGAN. Evaluation is shown in the next sub-section.

4.4.4 Experiments and Results

We first qualitatively evaluate the generation ability of the XVGAN compared to three baselines. Moreover, we explore the performance on vehicle re-ID compared to some state-of-the-arts on two public vehicle re-ID datasets. All the experiments are implemented based on the deep learning framework TensorFlow [117]. The TensorFlow platform is also introduced in Appendix B.2.

Cross-View Generation with Correct Attributes and Viewpoints

Before evaluating the final re-ID performance, we first present the vehicle image generation results by our XVGAN compared to some baselines. To validate the effectiveness of each designed component in the XVGAN, we carefully evaluate three corresponding baselines explained in detail as follows.

VAE Architecture. To explore that GAN based framework can better recover the training distribution and be asymptotically consistent, we first set the Variational Auto-Encoder (VAE) architecture as a baseline. In this method, the Discriminative Net is discarded, and the Classification Net and Generative Net work as the encoder and decoder, respectively. All the convolutional, fully-connected and deconvolutional layers have exactly same settings with the XVGAN for a fair comparison. In addition to the KL divergence, we adopt ℓ_2 norm for optimizing the decoder loss.

Purely Attributes-conditioned GAN. Conventional GANs usually condition on class labels or semantic attributes. However, more descriptive intrinsic visual features are not taken advantage of if the Generative Net only takes noise and some discrete values of attributes as inputs. To validate the superiority of our cross-view image-conditioned GAN, we evaluate a baseline of GAN purely conditioned on discrete attributes. In this method, after predicting the attribute labels of a vehicle in an original view image by the Classification Net, the ID, type, color and the expected viewpoint labels are directly encoded to a 384-dimensional (256+128) vector and then concatenated with the random noise vector. The structures of the Generative and Discriminative Nets are same as that of the XVGAN.

XVGAN without Matching-awareness. The goal of XVGAN is not only to generate real vehicle images, but also to infer images with correct attributes and viewpoints. A generated image with mismatched vehicle attributes or viewpoints cannot contribute to the final re-ID performance. The multi-attributes learning of generated views is configured for the constraint in the Discriminative Net. To prove the effectiveness of this design, an ablation experiment of dropping the multi-attributes learning is conducted.

Figure 4.12 demonstrates some qualitative examples of generated vehicle images by our XVGAN compared to three baselines. According to the results, we have the following observations and analysis. First, the VAE architecture generated much more blurred images compared to GAN-based frameworks, even though most correct attributes and viewpoints can be successfully generated. For the attributes-conditioned GAN, we find that the intra-variations within the synthesized same color and type vehicles are large since the detailed intrinsic features of the visible original view images are not exploited. In other words, from an input view image, the attributes-conditioned GAN can generate many similar vehicles rather than exactly the same target, thus, its improvement for the re-ID task is also limited. Moreover, the XVGAN without the matching-awareness constraint can generate real vehicle images, however, some attributes and viewpoints of the inferred views frequently mismatch the original vehicle. Finally, our XVGAN can synthesize highly real vehicle images with correct attributes and viewpoints in most cases. Its effectiveness for vehicle re-ID is further investigated in the next two sections.

Re-identification on the VeRi Dataset

To evaluate the re-ID performance, in addition to the image generation based models, we also compare to five traditional one-view based methods which only exploit the features of original views to measure distances across different views. One is simply adopting the second fully-connected layer in the Classification Net of XVGAN. The LOMO [88] feature is a highly successful handcrafted feature adopted for person re-ID. Moreover, the deep person re-ID model of domain guided dropout (DGD) [149] is also transferred to vehicles by re-training on [94, 99]. The GoogLeNet feature extracted from the model fine-tuned on vehicles in [153], is a solid deep representation containing rich semantic vehicle attributes information. A weighted combination of SIFT, Color Name, and GoogLeNet features, proposed as FACT in [99, 98], can well discriminate vehicles in joint domains. Besides, since the VAE baseline does not have the Discriminative Net, we concatenate the last convolution layer of the encoder and the first convolution layer of the decoder instead, and then learn the *ReidFeat* by the contrastive loss. Furthermore, we also compare a view synthesis method

Fig. 4.12 Qualitative examples of generated vehicle images in different viewpoints from only one original visible view. The left column shows the original vehicle images and their corresponding ground truths in five viewpoints. The right four columns show generated samples by VAE, Attributes-conditioned GAN, XVGAN without matching-awareness and XVGAN.

	Methods	mAP	Top-1	Top-5	Top-20	Top-50
One-View	LOMO	9.03	23.89	40.32	58.61	73.96
	GoogLeNet	17.58	51.98	66.79	78.77	86.37
	FACT	18.54	52.35	67.16	79.97	87.09
	XVGAN-C	18.42	51.14	64.50	79.42	91.51
	DGD	17.96	50.51	68.86	80.05	87.62
Multi-View	AppFlow	18.91	53.09	69.78	80.56	88.93
	VAE	16.68	46.52	61.41	74.94	84.25
	Attr-GAN	17.53	52.75	66.13	78.16	86.36
	XVGAN-w/o-M	20.07	55.21	69.85	82.87	91.76
	XVGAN	**24.65**	**60.20**	**77.03**	**88.14**	**93.95**

Table 4.6 mAP and matching rate (%) at rank-1, 5, 20 and 50 on VeRi Dataset. Attr-GAN denotes Attribute-conditioned GAN, and w/o-M is abbreviation of without Matching-awareness. XVGAN-C only adopts the feature in the Classification Net.

by appearance flow (AppFlow) [170]. Since only one visible input view is available in the vehicle re-ID test phase, we adopt the single-input view network of AppFlow.

Table 4.6 illustrates the mAP and matching rate results by different methods. The FACT feature achieves the highest mAP of 18.54% among the five one-view based methods. However, our XVGAN beats FACT by 6.11%. The improvement shows the effectiveness of the cross-view generation can indeed contribute to the vehicle re-ID problem in disjoint views. Moreover, without the design of matching-aware multi-label supervision in the Discriminative Net, the mAP of XVGAN decreases by 4.58%. Also, neither of VAE-based generation model and purely attribute-conditioned GAN achieves satisfactory performance. Thus, each component of design in the XVGAN is proved to be significant for re-ID. The matching rates of XVGAN at top-1, 5, 20, 50 are consistently higher than those of other baselines. The detailed comparison of CMC curves is shown in Figure 4.13. Besides, Figure 4.14 demonstrates qualitative examples of top-20 ranks for some query vehicles. We can observe that images of the same vehicle with large viewpoint variations compared to the query one can be successfully distinguished from many similar candidates in most cases. However, some false hits still exist usually caused by homogeneous visual patterns from very similar candidates in the same viewpoint.

Re-identification on the VehicleID Dataset

To make our method more convincing and show its generalizability, we evaluate the XVGAN on another large-scale vehicle dataset: VehicleID [94]. All the vehicles in the VehicleID dataset are captured in up to only two viewpoints: front and rear. The dataset is divided into

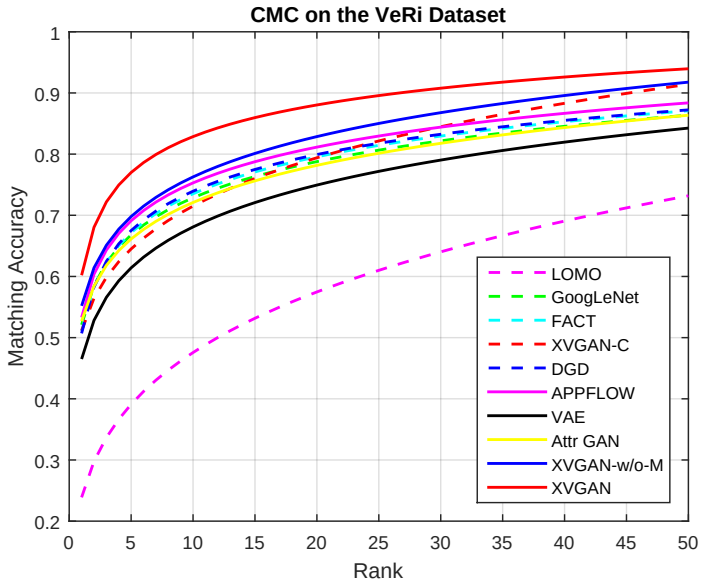


Fig. 4.13 CMC results evaluated on the VeRi dataset. The solid lines denote image generation models, while the dashed lines are methods only exploiting the original one view.

Fig. 4.14 Qualitative success and failure examples of top-20 rank on the VeRi dataset. Green boxes mean correct hits, while red boxes denote wrong ones.

	Methods	Top-1	Top-5	Top-20	Top-50
One-View	VGG+CCL	43.62	64.84	80.12	89.29
	Mixed Diff+CCL	48.93	75.65	88.47	93.37
	FACT	39.85	58.47	74.98	86.36
	XVGAN-C	42.31	60.26	76.77	88.24
	DGD	44.72	66.68	81.35	90.31
Multi-View	AppFlow	45.52	69.45	82.02	90.15
	VAE	37.62	56.84	75.36	88.38
	Attr-GAN	45.83	74.33	85.94	90.23
	XVGAN-w/o-M	44.91	73.48	87.04	92.58
	XVGAN	**52.89**	**80.84**	**91.86**	**95.83**

Table 4.7 Matching accuracies (%) at rank-1, 5, 20 and 50 on the two-view VehicleID Dataset.

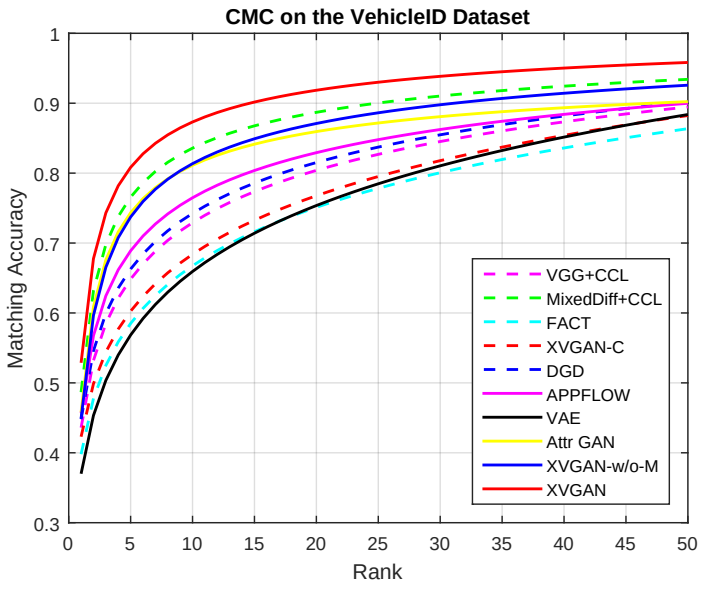


Fig. 4.15 CMC results evaluated on the VehicleID dataset. The solid lines denote image generation models, while the dashed lines are methods only exploiting the original one view.

the training set with 110,178 images of 13,134 vehicles and the test set with 111,585 images of 13,133 vehicles. A coupled clusters loss (CCL) and a mixed difference network structure (Mixed Diff) for vehicle re-ID are also introduced in [94]. Following its evaluation protocol, we conduct the image-to-image search. One image is randomly selected for each vehicle in the test set to construct the gallery set with the size of 800. Other images are adopted as query ones. The experiment is carried out 10 times to obtain the final results.

CMC curves are illustrated in Figure 4.15. As shown in Table 4.7, XVGAN increases the top-1 and 5 matching rates by 3.96% and 5.19%, respectively, compared to the second place Mixed Diff+CCL. The FACT feature performs poorly on this dataset since neither of its components can be discriminative for small inter-variations between vehicles in the single viewpoint. The large margin between XVGAN and XVGAN-C strongly proves the significance of the contribution by the cross-view inference. Moreover, VAE also gets low accuracies because the generated cross-view images are blurred, losing details. The top-1 rate by Attr-GAN or XVGAN-w/o-M is 7.06% or 7.98% lower than that of XVGAN, respectively. Therefore, GAN models, conditioned on only discrete attribute labels or without matching-awareness design, are ineffective for vehicle re-ID.

4.5 Conclusions

In this chapter, we presented two end-to-end deep networks: the SCCN and XVGAN to address the challenging multi-view vehicle re-ID task. The SCCN adopts a convolutional encoder-decoder architecture to learn transformations across different viewpoint pairs of a vehicle. The XVGAN is designed for cross-view vehicle generation by a conditional generative adversarial network. Extensive experimental results showed that our models could achieve promising results and outperform state-of-the-art methods for vehicle re-ID. However, these two models are both based on image generation so that their performance for re-ID will be limited by the quality of generated images. In the next chapter, we will present the feature-level transformations of multi-view processing for vehicle re-ID which do not require image synthesis.

Chapter 5

Multi-View Feature Transformation for Vehicle Re-identification

5.1 Introduction and Motivation

As discussed in Chapter 4, the motivation of multi-view vehicle re-identification (re-ID) is due to the two main obstacles. One inherent difficulty is that a vehicle captured in different viewpoints usually has dramatically varied visual appearances. In contrast, two different vehicles of the same color and type have a similar appearance from the same viewpoint. The subtle inter-instance discrepancy between images of different vehicles and the large intra-instance difference between images of the same vehicle make the matching problem addressed in an unsatisfactory way by existing vision models.

Methods presented in Chapter 4 aim to address multi-view problem through synthesizing cross-view images. However, the re-ID results are limited by the blurred image quality and low resolution of generated samples. In this chapter, we extend our attention to vehicle re-ID from view synthesis to multi-view feature-level transformation. Generally speaking, features of an input image is first extracted that only contains one view's visual content. Then, transformations can be learned for mining correlations between the visible view and other hidden views. Note that the transformations here are in terms of features rather than explicit geometric transformations. Finally, we can fuse all the inferred features in different views and adopt this final representation for distance metric learning where distance metrics can be learned in the generated viewpoint-invariant multi-view feature space.

Our main contributions of this chapter are highlighted as follows:

* We propose an adversarial bi-directional long short-term memory (LSTM) network to learn transformations between features in continuously changing viewpoints of vehicles.

Given only one input view of each vehicle, contrastive learning and adversarial learning are jointly employed to infer discriminative multi-view features to distinguish vehicles. Extensive ablation studies and comparison experiments conducted on the VehicleID and VeRi datasets have demonstrated the effectiveness and superiority of our method over state-of-the-art vehicle re-ID methods.

* A viewpoint-aware attention model is proposed to obtain attention maps from the input image. The high-scored region of each map shows the overlapped appearance between the input vehicle's view and a target viewpoint. For instance, to infer the side view feature from a front-side view input image, the VAMI will only pay attention to the vehicle's side pattern while ignoring the front region. Given the attentive features of a single-view input, we design a conditional multi-view generative network to infer a global feature containing different viewpoints' information of the input vehicle. The adversarial training mechanism and auxiliary vehicle attribute classifiers are combined to achieve effective feature generation. In addition to inferring multi-view features, we embed pairwise distance metric learning in the network to place the same vehicle together and push different vehicles away.

5.2 Related Work

5.2.1 Long Short-Term Memory Convolutional Neural Networks

Long Short-Term Memory (LSTM) networks are a special variant of RNN, capable of learning long-term dependencies. In addition to its success in the applications in natural language processing, LSTM models have also been widely explored in the following vision tasks.

The most common idea is related to models for action recognition performed on video sequences. A long-term recurrent convolutional network (LRCN) is proposed by Jeff *et al.* [26] for large-scale visual learning with an end-to-end architecture. LRCN extracts features from the variable-length visual input with a CNN and then feeds the outputs into a number of LSTM layers which finally produce a variable-length prediction. Ma *et al.* [102] aim to address activity detection by learning activity progression in LSTMs. Given the visual features obtained by a CNN, LSTM with a linear layer that is adopted to compute the detection scores based on the CNN features of the current step and the hidden states and memory from the previous step. A ranking loss is designed at the end to train the entire network.

Image captioning is another research area where recurrent LSTMs can hugely improve the performance. Image captioning connects computer vision and natural language processing,

addressing a problem that generates natural sentences to describe an image. Oriol *et al.* [140] introduced a model which is trained to maximize the likelihood of the target description sentence given the training image. The LSTM-based sentence generator combines the CNN image representation and word embeddings. Moreover, an alignment model [70] is presented to align visual and language data. The method performs CNN over image regions and bi-directional RNN over sentences. An alignment objective adopts a multimodal embedding to align the two modalities.

5.2.2 Adversarial Learning

Adversarial learning is one of the most important areas of deep learning. With the goal of modeling the real data distribution, the generator learns to generate realistic samples of data while the discriminator learns to determine if these samples are real or not. Generative adversarial networks (GANs) [44] have achieved great success on many vision tasks such as image generation and image translation. In essence, the design of the adversarial learning leads to the success of GAN, mainly because it forces the generated samples to be indistinguishable from real data.

In applications, image generation models have been extensively explored. Alec Radford *et al.* [117] introduced the deep convolutional generative adversarial network which gets promising results on the LSUN bedroom dataset and human face dataset. A text-conditional convolutional GAN architecture [119] is proposed to automatically synthesize realistic images from text. Moreover, Phillip *et al.* [63] presented a conditional adversarial network to translate an input image into a corresponding output image. For instance, translate gray images to color images, translate sketch images to real images.

GANs are trained to minimize the distance between the generated and true data distributions. Initially, the Jensen-Shannon divergence was used as this distance metric. However, Wasserstein GAN (wGAN) [3] provides extensive theoretical work and shows empirically that minimizing a reasonable and efficient approximation of the Earth Mover's (EM) distance is a theoretically sound optimization problem that cures the main drawbacks of GANs. For this approximation of the EM distance to be valid, wGAN imposes weight clipping constraints on the critic (referred to as the discriminator pre-Wasserstein) which causes some training failures. Moreover, improved training of Wasserstein GANs [51] enables very stable GAN training by penalizing the norm of the gradient of the critic with respect to its input instead of clipping weights. This gradient penalty is simply added to the Wasserstein distance for the total loss.

5.2.3 Visual Attention Mechanism

Visual attention mechanisms aim to automatically focus on the core regions of image inputs and ignore the useless parts. Existing attention mechanisms can be mainly categorized into two groups. One is the fully-differentiable soft attention model which aims to learn attention maps to weight different regions of an image. The other is the hard attention model which is a stochastic process sampling hidden states with probabilities. Hard attention models are not differentiable and usually learned by reinforcement learning.

The visual attention models' ability of selective feature extraction has been extensively explored in many applications including image classification, fine-grained image recognition, image captioning and VQA. Volodymyr *et al.* [108] proposed the recurrent models of visual attention that adaptively select and process a sequence of regions, and evaluated the models for image classification. A multi-attention convolutional neural network [165] is designed for fine-grained image recognition, which consists of convolution, channel grouping, and part classification sub-networks. Moreover, in visual question answering models, visual attention plays an important role to assign weights to image regions based on their relevance to a question. Chen *et al.* [173] proposed a grid-structured CRF to model the visual attention as a multivariate distribution.

5.3 Adversarial Bi-directional LSTM Network

Based on the great advantages of feature learning and sequential data modeling by CNN and LSTM, we propose a deep network - Adversarial Bi-directional LSTM Network (ABLN) - to address the arbitrary-view vehicle re-ID problem. Figure 5.1 sketches an overview of the ABLN whose training pipeline can be partitioned into three sub-tasks. The first one adopts a CNN to learn features of vehicles' model and viewpoint, and then aims to infer multi-view features by an LSTM from only one input view. The LSTM module is bi-directional, whose details will be discussed later. A simple reconstruction loss is used for optimization. The second part adopts an adversarial training architecture that treats the LSTM (LSTM-G) as a feature generator, and adds another LSTM (LSTM-D) working as the discriminator to differentiate the real multi-view features and the inferred ones. The optimization goal of this sub-task is to converge the inferred feature distribution to the target real feature distribution. After generating the global all-view features, we employ one more re-ID loss to place the same vehicle nearby and keep different vehicles apart as far as possible in the desired same feature space.

The problem formulation is similar to that in Sec. 4.3.1. Learning an input view's feature is easily addressed by training a deep CNN using available vehicle-related labels.

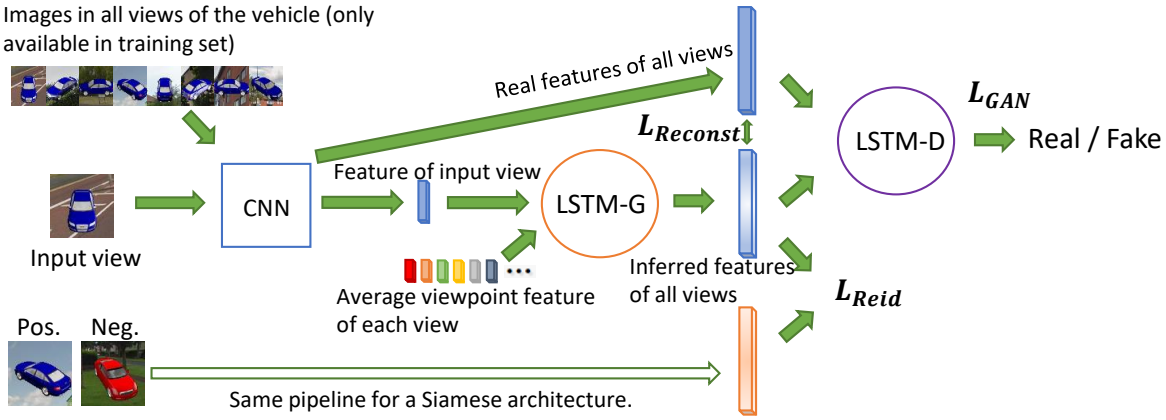


Fig. 5.1 Main components of our approach: The CNN extracts input vehicle image's feature. The LSTM-G infers multi-view features from only one input view. The LSTM-D discriminates whether the multi-view features are real or generated. A Siamese architecture is also built for learning distance metrics, given a vehicle image pair.

However, learning transformations to infer effective features of invisible views is more difficult. Figure 5.2 illustrates the architecture of the proposed ABLN. The left part is a CNN module for learning input vehicles' model and viewpoint features. The LSTM-G aims to learn transformations from the input view feature to the hidden view features. The LSTM-D, which discriminates the real multi-view features and the generated features, is optimized against the LSTM-G in an adversarial training architecture. In the training phase, we input the one-view feature of a vehicle for the LSTM-G. To learn the transformations, we take the input vehicle's other view features both as the supervision for a reconstruction loss $\mathscr{L}_{Reconst}$ in LSTM-G and the real data inputs of LSTM-D to train an adversarial loss $\mathscr{L}_{Advers}$ against the generated data from LSTM-G. Moreover, given an image pair, a re-ID loss $\mathscr{L}_{Reid}$ is configured at the end of the LSTM-G for distance metric learning. In the test phase, the input one-view image only needs to forward pass the CNN and LSTM-G modules, and then all the inferred features in different viewpoints can be simply concatenated for computing the distances with other vehicles' images. We clearly explain the details of each component in the following three sub-sections.

5.3.1 Feature Extraction and Viewpoint Estimation

The CNN part of the ABLN, for extracting vehicle features and estimating corresponding viewpoints, deploys six convolutional layers and two fully-connected layers. The detailed structure and the hyper-parameter settings are illustrated in the left part of Figure 5.2. The input image size is $128 \times 128 \times 3$. We configure 512-*dim* fc_attr to learn vehicle features

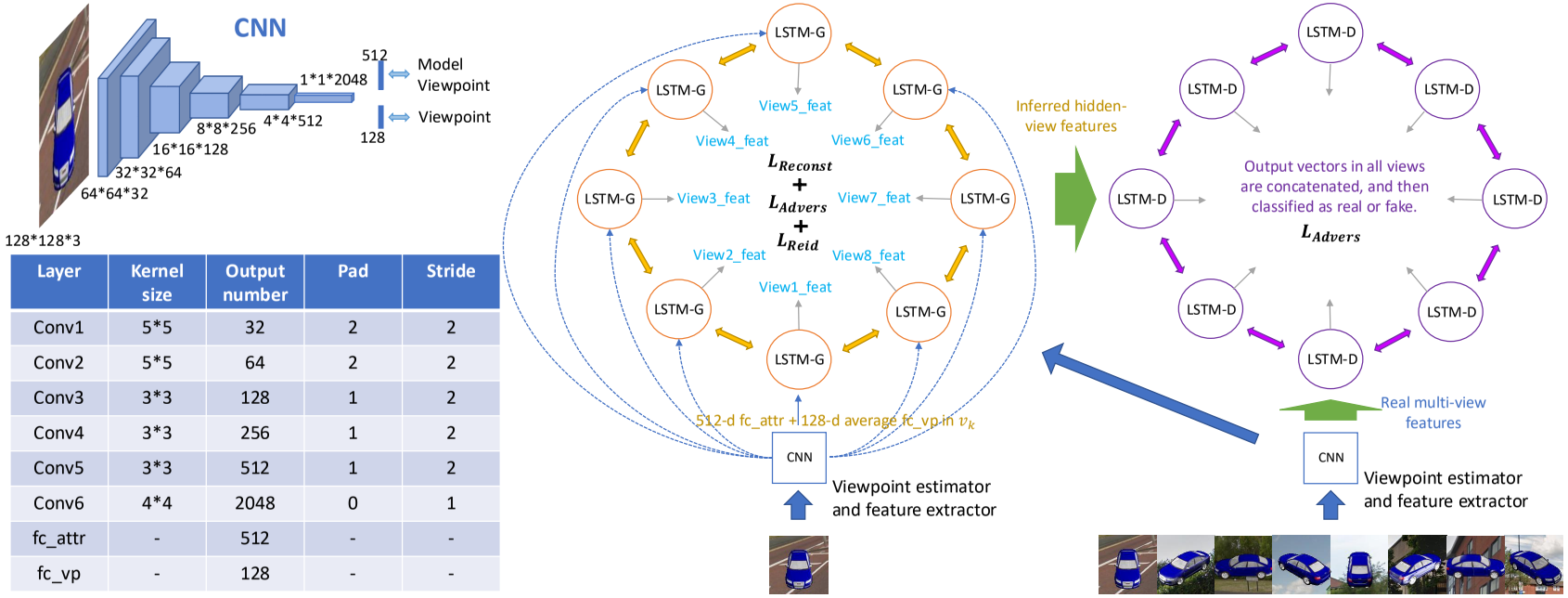
by multi-classification using vehicles' model and viewpoint labels, while set 128-dim fc_vp to learn viewpoint features only using viewpoint labels. Thus, fc_attr can well describe the intrinsic vehicle feature of the input view, while fc_vp focuses on discriminative patterns for viewpoint estimation.

Once the CNN module is pre-trained, the features of layer fc_attr are set for the inputs and outputs of LSTM-G to learn feature transformations by the reconstruction loss. Moreover, they are also adopted as the real data of the LSTM-D for adversarial learning. In addition, we extract the features of layer fc_vp and compute the average feature $\overline{\mathbf{v}_v}$ of each viewpoint v on the whole training set. These pre-computed average viewpoint features are taken as conditional embeddings for the input of LSTM-G. To better adopt the cross-entropy loss to optimize $\mathscr{L}_{Reconst}$ for LSTM-G, we normalize the CNN features within the range [0,1]. Moreover, the viewpoint of each input vehicle image can be estimated to determine the index of the average viewpoint feature at each view step of LSTM-G in the test phase.

5.3.2 Bi-directional LSTM Inference

LSTM, which aims to learn long-term dependencies, is a variant of the recurrent neural network (RNN). Its architecture enables the network to add new information and discard previous useless messages by gates in the cell state. Due to the great success of LSTM applied in modeling sequential data such as activity recognition and image captioning, we expect that the visual appearance variations across continuously changing viewpoints of vehicles can be well modeled by LSTM. Instead of remembering information for long periods of time, we investigate LSTM to learn the spatial change of a 3D vehicle structure. Therefore, a bi-directional LSTM loop is proposed for learning the transformed features in V continuous viewpoints, where V equals 8 as demonstrated in Figure 5.2 and Figure 5.3.

An apparent drawback of traditional LSTMs is that they only exploit the previous (one-side) information to generate the outputs. However, in the case of multi-view vehicle re-ID, the feature in one view of a vehicle is closely related to the two neighboring views. Therefore, the core component of the proposed LSTM loop comprises two separate LSTM modules: one processes data in the clockwise direction, while the other is set for the anti-clockwise direction.



Layer	Kernel size	Output number	Pad	Stride
Conv1	5*5	32	2	2
Conv2	5*5	64	2	2
Conv3	3*3	128	1	2
Conv4	3*3	256	1	2
Conv5	3*3	512	1	2
Conv6	4*4	2048	0	1
fc_attr	-	512	-	-
fc_vp	-	128	-	-

Fig. 5.2 An overview of the architecture of ABLN. The left part is a CNN for learning input vehicles' model and viewpoint features. The LSTM-G module aims to learn feature transformations between the input view and other invisible views. We design three different losses to learn these transformations. The first one is to adopt all the different viewpoint features of the input vehicle as supervision information to optimize a reconstruction loss $\mathscr{L}_{Reconst}$. Additionally, we build another LSTM-D module to discriminate the real different view features and the inferred ones. The two LSTM modules can be optimized as a generator and a discriminator by an adversarial loss $\mathscr{L}_{Advers}$. Moreover, given an image pair, a re-ID loss $\mathscr{L}_{Reid}$ is configured at the end of the LSTM-G for distance metric learning. Note that both the LSTM-G and LSTM-D are bi-directional.

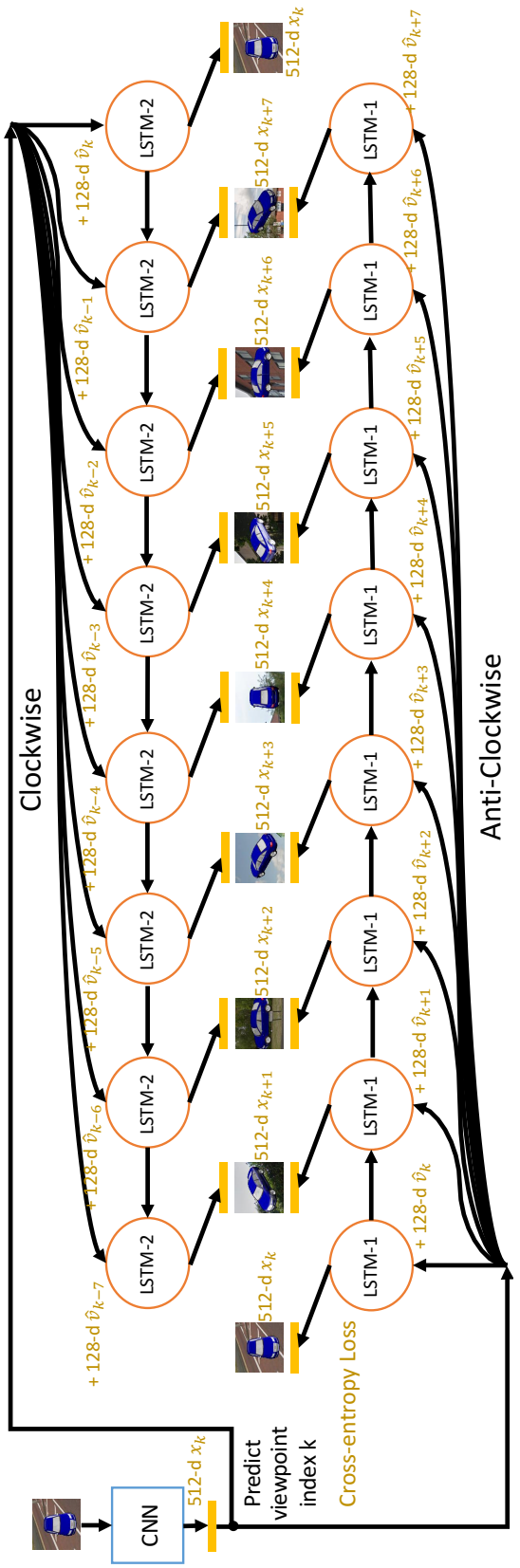


Fig. 5.3 Details of the unwrapped LSTM-G module. At each view step of the LSTM unit, the input is a feature vector concatenated by the feature of the input view and the average viewpoint feature of that certain view. The outputs are supervised by the features of the corresponding views of the input vehicle using a cross-entropy loss.

Given a group of features in different views surrounding a vehicle $\mathbf{x} = \{\mathbf{x}_v\}_{v=1}^{v=V}$, we select one feature $\mathbf{x}_k$ in arbitrary viewpoint index k as the input, and the remaining features (also including $\mathbf{x}_k$) are used as supervision vectors for learning the transformations between different views. The details of the bi-directional LSTM loop architecture can be unwrapped in Figure 5.3. The LSTM-1 and LSTM-2 layers learn the models in two directions separately. At each view step, we concatenate the 512-*dim* input view feature $\mathbf{x}_k$ with the 128-*dim* average viewpoint feature $\hat{\mathbf{v}}_v$ in that view as the input. Thus, the inputs of two LSTM modules are 640-*dim* feature vectors as follows:

$$\mathbf{X}_v = \text{concat}(\mathbf{x}_k, \hat{\mathbf{v}}_v), \quad \forall v \in \{1, \ldots, V\}, \tag{5.1}$$

Where $v = 1, \ldots, V$ is a cycle with V viewpoints. As shown in Figure 5.3, if the input viewpoint k is 1, the sequential $\hat{\mathbf{v}}_v$ will be $\hat{\mathbf{v}}_1$,$\hat{\mathbf{v}}_2$,...,$\hat{\mathbf{v}}_7$ and $\hat{\mathbf{v}}_8$ for the anti-clockwise direction, and $\hat{\mathbf{v}}_1$,$\hat{\mathbf{v}}_8$,...,$\hat{\mathbf{v}}_3$ and $\hat{\mathbf{v}}_2$ for the clockwise direction. The LSTMs encode the inputs $\mathbf{X}_v$ to hidden states $\mathbf{h}_v^d$ ($d \in \{\text{ck}, \text{ack}\}$, where ck is short for clockwise and ack is short for anti-clockwise), and hidden states to output the vectors $\mathbf{y}_v^d$ in different views by iterating the following equations:

$$\mathbf{h}_v^d = H(\mathbf{W}_{xh}^d \mathbf{X}_v + \mathbf{W}_{hh}^d \mathbf{h}_{v-1}^d + \mathbf{b}_h^d), \tag{5.2}$$

$$\mathbf{y}_v^d = \mathbf{W}_{hy}^d \mathbf{h}_v^d + \mathbf{b}_y^d, \tag{5.3}$$

Where H denotes the formulation of the LSTM modules. In our model, we implement H as the following functions:

$$\begin{aligned}
\mathbf{i}_v^d &= \sigma(\mathbf{W}_{xi}^d \mathbf{X}_v + \mathbf{W}_{hi}^d \mathbf{h}_{v-1}^d + \mathbf{b}_i^d), \\
\mathbf{f}_v^d &= \sigma(\mathbf{W}_{xf}^d \mathbf{X}_v + \mathbf{W}_{hf}^d \mathbf{h}_{v-1}^d + \mathbf{b}_f^d), \\
\mathbf{c}_v^d &= \mathbf{f}_v^d \cdot \mathbf{c}_{v-1}^d + \mathbf{i}_v^d \cdot \tanh(\mathbf{W}_{xc}^d \mathbf{X}_v + \mathbf{W}_{hc}^d \mathbf{h}_{v-1}^d + \mathbf{b}_c^d), \\
\mathbf{o}_v^d &= \sigma(\mathbf{W}_{xo}^d \mathbf{X}_v + \mathbf{W}_{ho}^d h_{v-1}^d + \mathbf{b}_o^d), \\
\mathbf{h}_v^d &= \mathbf{o}_v^d \cdot \tanh(\mathbf{c}_v^d),
\end{aligned} \tag{5.4}$$

Where the intermediate variables $\mathbf{i}_v^d, \mathbf{f}_v^d, \mathbf{c}_v^d$ and $\mathbf{o}_v^d$ denote the input gate, forget gate, memory cell and output gate in the two opposite LSTM units, respectively. In addition, $\sigma(x) = (1 + e^{-x})^{-1}$ and $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. Element-wise product operation is denoted by $\cdot$. For

both the LSTM-1 and LSTM-2 modules, we configured 3 layers (each with 512 hidden units) which empirically shows the best performance to learn feature transformations across neighboring views of a vehicle.

The outputs of LSTM-1 and LSTM-2 are denoted as 512-*dim* vectors $\hat{\mathbf{x}}_v^{ack}$ and $\hat{\mathbf{x}}_v^{ck}$, respectively. Furthermore, the output vector $\hat{\mathbf{x}}_v$ in each view is fused by $\mathbf{W}_v^{ack}\hat{\mathbf{x}}_v^{ack} + \mathbf{W}_v^{ck}\hat{\mathbf{x}}_v^{ck} + \mathbf{b}_v$, where $\mathbf{W}_v^{ack}$ and $\mathbf{W}^{ck}$ are learnable weights. At each view step, the reconstruction error in the LSTM-G module is optimized by a cross-entropy loss as:

$$\mathscr{L}_{Reconst} = -\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_v^i \log \hat{\mathbf{x}}_v^i + (1-\mathbf{x}_v^i)\log(1-\hat{\mathbf{x}}_v^i)). \tag{5.5}$$

Moreover, $\hat{\mathbf{x}}_v$ in all viewpoints are concatenated into a $512 \times V$ dimensional feature vector $\mathbf{f}$, which can be regarded as a global representation to describe a vehicle comprehensively. Finally, we configure one more fully-connected layer reducing the feature dimension by 4 times for further optimizing the re-ID loss $\mathscr{L}_{\text{reid}}$ in a Siamese way.

Adversarial LSTM Training

A generative adversarial network (GAN) consists of a generative net and a discriminative net which are two competing neural network models. The generative net usually takes a latent random vector z from a uniform or Gaussian distribution as input to generate samples, while the discriminative net aims to distinguish the real data x from inferred samples. The $p_z(z)$ is expected to converge to a target true data distribution $p_{data}(x)$. In this sub-section, we provide the noise in the form of dropout on the vehicle feature embeddings as inputs for the generative net LSTM-G. To make the inferred hidden-view features as close as the real multi-view features, we design the LSTM-D to discriminate the real or inferred features for adversarial training against LSTM-G.

As shown in the right part of Figure 5.2, the LSTM-D is also a bi-directional LSTM loop, where the input at each view step is the corresponding inferred $\hat{\mathbf{x}}_v$ or real $\mathbf{x}_v$. In both directions, we also configure 3 LSTM layers whose unit sizes are 512. Similar to the LSTM-G, the outputs of the LSTM-D at each view step in two directions are fused by learnable weights, and then concatenated in all viewpoints to be classified as real or fake. The adversarial loss is defined as:

$$\mathscr{L}_{Advers} = \mathbb{E}[\log(LSTMD(\{\mathbf{x}_v\}_{v=1}^{v=V}))] + \mathbb{E}[\log(1 - LSTMD(\{\hat{\mathbf{x}}_v\}_{v=1}^{v=V}))], \quad (5.6)$$
$$where\ \{\hat{\mathbf{x}}_v\}_{v=1}^{v=V} = LSTMG(\mathbf{x}_{v=k}, \{\hat{\mathbf{v}}_v\}_{v=1}^{v=V}).$$

Algorithm 5.1 Train LSTM-G and LSTM-D

Inputs: input single-view features $\mathbf{x}_{v=k}$, corresponding real multi-view features $\{\mathbf{x}_v\}_{v=1}^{v=V}$ and average viewpoint features in each view $\{\hat{\mathbf{v}}_v\}_{v=1}^{v=V}$
Outputs: learned parameters $\{\theta_{lstmG}, \theta_{lstmD}\}$
Initialize all parameters $\{\theta_{lstmG}, \theta_{lstmD}\}$
for i in max number of iterations **do**
    $X_S \leftarrow$ mini-batch $\mathbf{x}_{v=k}$ % k is input view index
    $X_M \leftarrow$ mini-batch $\{\mathbf{x}_v\}_{v=1}^{v=V}$
    $\hat{X}_M \leftarrow$ LSTM-G(X_S, $\{\hat{\mathbf{v}}_v\}_{v=1}^{v=V}$)
    Update LSTM-D by ascending its stochastic gradient: $\theta_{lstmD} \leftarrow \nabla\mathscr{L}_{Advers}$
    Update LSTM-G by descending its stochastic gradient: $\theta_{lstmG} \leftarrow \nabla(\mathscr{L}_{Advers} + \mathscr{L}_{Reconst})$
    if $i > C$ **then** % after C iterations, add $\mathscr{L}_{Reid}$ for training
        Combine input single-view features into positive and negative pairs
        Update LSTM-G by descending its stochastic gradient: $\theta_{lstmG} \leftarrow \nabla\mathscr{L}_{Reid}$
    end if
end for

5.3.3 Optimization

The CNN model is pre-trained by multi-classification using vehicles' model and viewpoint labels. Then the CNN features are extracted to train the LSTM-G and LSTM-D. Given the definitions of $\mathscr{L}_{Advers}$ and $\mathscr{L}_{Reconst}$, as well as $\mathscr{L}_{Reid}$ defined in Eq. 4.2, we adopt the Stochastic Gradient to train the LSTM-G and LSTM-D. Algorithm 5.1 summarizes the steps of optimization.

5.3.4 Experiments and Results

In this sub-section, we evaluate our ABLN by ablation studies and comparisons with state-of-the-art person and vehicle re-ID methods. All the experiments are implemented based on the deep learning framework TensorFlow.

Datasets and Evaluation Protocols

To train the proposed ABLN requires a vehicle dataset where each vehicle has images in densely sampled views. We adopt the ShapeNet dataset [12] to render 2D images from 7,497 3D car models. For each car model, we averagely sample 32 azimuthal views ranging from 0°to 355°, and set the elevation angle as 30°with a fixed camera distance, since most of the test vehicle samples in realistic surveillance datasets are captured around this angle. Random backgrounds are overlaid for the rendered images. To reduce the appearance gaps between the 3D models and real images, we also simulate different lightings. Moreover, we randomly combine 40,000 positive pairs and 80,000 negative pairs to optimize the contrastive loss for re-ID.

We first pre-train the CNN module by 30 epochs. Afterward, we train the LSTM-G and LSTM-D by optimizing only $\mathscr{L}_{Advers}$ and $\mathscr{L}_{Reconst}$ in the first 50 epochs, and then include $\mathscr{L}_{Reid}$ for a joint training by another 150 epochs, since at the early training stage the inferred multi-view features are extremely poor, thus the $\mathscr{L}_{Reid}$ cannot contribute to the optimization. We adopt the ADAM Optimizer with the learning rate of 0.0002 and the momentum of 0.5. The mini-batch size is set as 64.

In the test phase, we do evaluation on two large real vehicle surveillance datasets: VeRi [99] and VehicleID [94].

Ablation Studies

Before comparing the ABLN to state-of-the-art methods, we first do ablation studies to explore the effectiveness of each significant component designed in our model. In this part, all the variants of our model are only trained on the ShapeNet and then evaluated on the two real vehicle datasets without fine-tuning for comparisons.

Effect of Number of Inferred Views

To explore the effectiveness of hidden view inference, the re-ID performance versus the number of inferred views is investigated. We train our ABLN configured with a different V number of inferred views. Results are reported when V equals 4, 8, 16 or 32. All the views are averagely sampled with an equal interval (e.g. Figure 5.2 and Figure 5.3 illustrate the version when V equals 8.). Moreover, we also compare with the version when V equals 0 which means only the input view feature is used for computing the re-ID distance.

As shown in Figure 5.4, the more views the model infers, the smoother feature transitions the ABLN can learn, thus improving the re-ID accuracy. Details of the results can be found in Table 5.1. We can observe that the top-1 rate by ABLN-32 outperforms that by ABLN-0 (no view inference) with big gaps of 13.68% and 9.53% on the VeRi and VehicleID datasets,

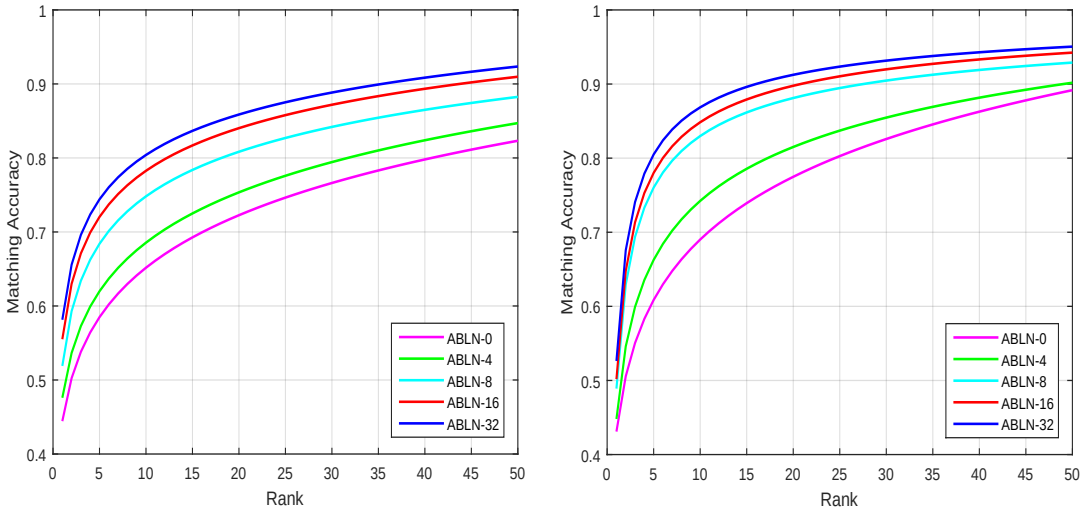


Fig. 5.4 CMC results of models trained by a different number of inferred views. (a) Comparisons on the VeRi dataset. (b) Comparisons on the VehicleID dataset.

Table 5.1 Results (%) of a different V numbers of inferred views.

	V	0	4	8	16	32
VeRi	mAP	15.21	16.85	18.67	20.81	**22.91**
	rank-1	44.46	47.60	51.92	55.51	**58.14**
	rank-5	58.49	61.99	68.40	72.03	**74.41**
	rank-20	72.26	75.35	80.84	84.02	**85.87**
	rank-50	82.32	84.71	88.25	90.97	**92.35**
VehicleID	rank-1	43.10	44.82	48.95	50.18	**52.63**
	rank-5	60.87	66.27	76.03	78.02	**80.51**
	rank-20	77.48	81.51	88.11	89.77	**91.25**
	rank-50	89.17	90.18	92.89	94.24	**95.05**

respectively. Moreover, some qualitative results are demonstrated on the VeRi dataset in Figure 5.5 to show the effectiveness of view inference for multi-view vehicle re-ID. It can be observed that ABLN-0 which does not infer other views' features can only rank those viewpoints similar to the input view in top positions. However, with the increase of the number of inferred views, we can find more candidates with different viewpoints are proposed in top positions, and correct hits in the top-10 ranks become increasingly more as well. To better illustrate the reconstructed multi-view features by our model, the t-SNE is also adopted to show the difference between the single-view features and the multi-view features in the right column of Figure 5.5. The features of images of 20 test vehicles are visualized that

Fig. 5.5 Qualitative top-10 rank comparisons of the ABLN-V trained by a different V number of inferred views. Blue boxes are query vehicles, while the green and red boxes denote correct hits and wrong ones, respectively. The right part visualizes the single-view features obtained by the ABLN-0 without any viewpoint inference and the multi-view features inferred by the ABLN-32.

samples of the same vehicle can be better clustered after multi-view inference compared with single-view features, since the multi-view features are more viewpoint-invariant.

Bi-directional vs. Single-directional

One arbitrary view of a vehicle usually shares some overlapping visual patterns with its neighboring views in opposite directions. If we only design a single-directional LSTM module to infer continuous view variations in one direction, the output feature at the last view step of the LSTM might be poor due to loss of information transmission. For example, given a left front-side input view, a single-directional LSTM infers view transformations in the clockwise direction. The feature of the left side view will be learned much better than that of the front view, even though the front view is also close to the input view. To address this problem, we propose the bi-directional design and fuse the features in two directions for final computation. Therefore, we explore the single-directional LSTM as a baseline. We fix the number of inferred views as 32 for comparison. Table 5.2 illustrates the results which prove the effectiveness of the bi-directional design. Moreover, we also compare our learnable fusion strategy of two directions with simply doing average and maximum operations.

Effects of Different Losses

Table 5.2 Comparisons (%) of bi/single-directional LSTM. ASLN is the abbreviation of Adversarial Single-directional LSTM Network. The number of inferred views is fixed as 32.

	Methods	rank-1	rank-5	rank-20	rank-50	mAP
VeRi	A**B**LN	**58.14**	**74.41**	**85.87**	**92.35**	**22.91**
	A**B**LN-ave	55.36	69.87	82.34	91.03	20.18
	A**B**LN-max	54.57	69.12	81.45	90.62	20.03
	A**S**LN	54.16	69.38	81.11	90.16	19.96
VehicleID	A**B**LN	**52.63**	**80.51**	**91.25**	**95.05**	-
	A**B**LN-ave	50.47	77.02	90.13	93.86	-
	A**B**LN-max	50.23	76.64	89.80	93.22	-
	A**S**LN	51.48	79.23	90.68	94.30	-

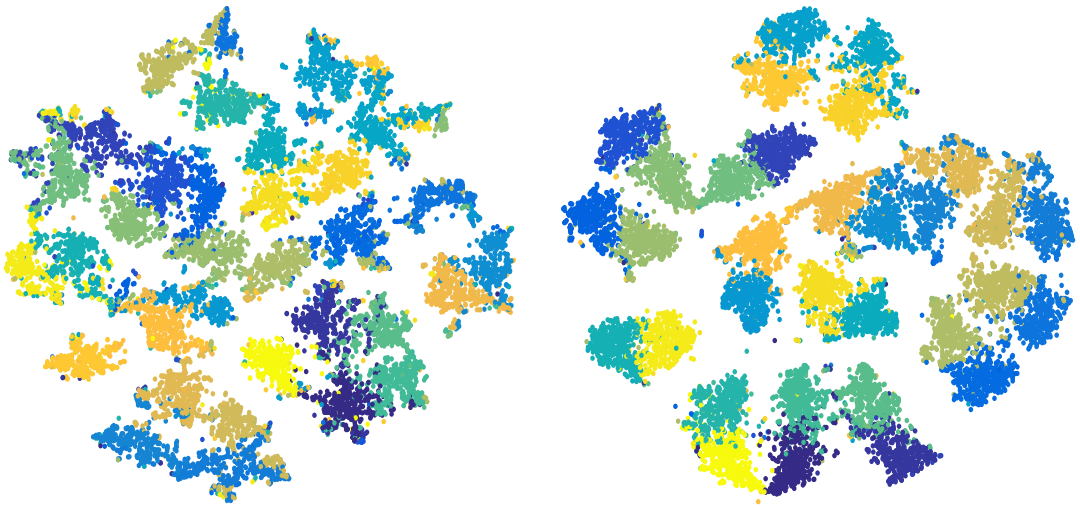
To investigate contributions by different losses for the re-ID accuracy, we drop the reconstruction, adversarial training, and contrastive losses, respectively. As illustrated in Table 5.3, dropping the adversarial training loss severely decreases the performance on both datasets, which proves $\mathscr{L}_{Advers}$ is essential to help the LSTM-G learning more real features in different views for each vehicle. We also demonstrate the inferred feature space by t-SNE [103] in Figure 5.6. It shows adversarial training can make the outputs at each view step of LSTM-G generate better view-specific features.

Table 5.3 Comparisons (%) of dropping different losses. The number of inferred views is fixed as 32.

	Losses	rank-1	rank-5	rank-20	rank-50	mAP
VeRi	$\mathscr{L}_{Reconst}+\mathscr{L}_{Advers}+\mathscr{L}_{Reid}$	**58.14**	**74.41**	**85.87**	**92.35**	**22.91**
	$\mathscr{L}_{Advers}+\mathscr{L}_{Reid}$	56.03	70.29	83.68	92.05	20.45
	$\mathscr{L}_{Reconst}+\mathscr{L}_{Reid}$	51.89	66.52	77.95	86.40	17.32
	$\mathscr{L}_{Reconst}+\mathscr{L}_{Advers}$	56.76	71.28	83.80	92.44	20.87
VehicleID	$\mathscr{L}_{Reconst}+\mathscr{L}_{Advers}+\mathscr{L}_{Reid}$	**52.63**	**80.51**	**91.25**	**95.05**	-
	$\mathscr{L}_{Advers}+\mathscr{L}_{Reid}$	50.95	77.68	90.30	94.11	-
	$\mathscr{L}_{Reconst}+\mathscr{L}_{Reid}$	45.37	69.10	81.86	89.97	-
	$\mathscr{L}_{Reconst}+\mathscr{L}_{Advers}$	51.56	78.81	90.66	94.36	-

Comparisons with State-of-the-arts

Finally, the ABLN model is compared with several state-of-the-art methods of person or vehicle re-ID. Improved Deep [1] and DGD [149] are deep models proposed for person re-ID. We transfer them to vehicles by re-training using the VeRi and VehicleID datasets. The handcrafted LOMO [88] feature also achieves great success on person re-ID. Moreover, the GoogLeNet feature extracted from the model fine-tuned on vehicles in [153], is a solid deep



(a) Training without $\mathscr{L}_{Advers}$ (b) Training with $\mathscr{L}_{Advers}$

Fig. 5.6 Visualization of the output feature space at different view steps of the LSTM-G.

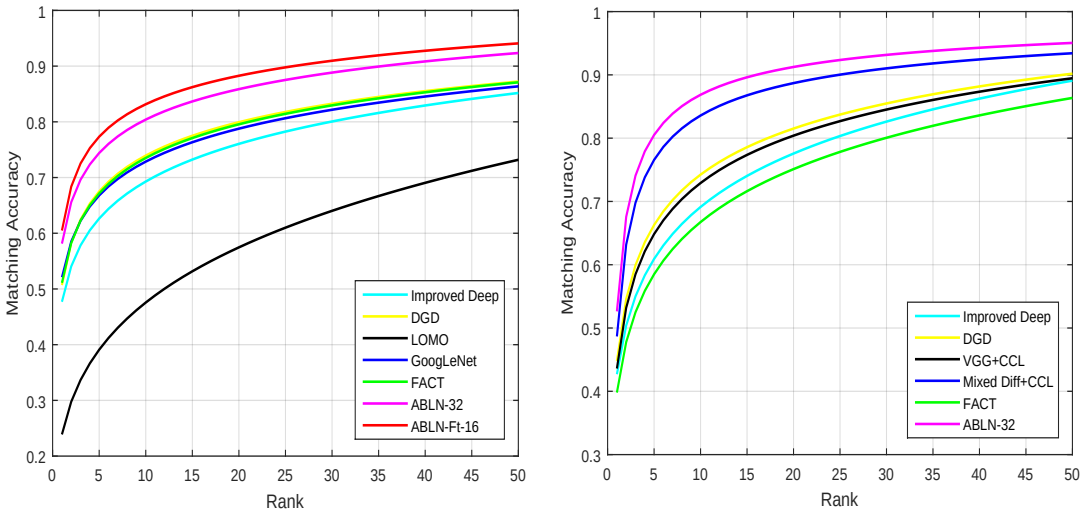
representation containing rich semantic vehicle attributes information. FACT [99], consisting of SIFT, Color Name, and GoogLeNet features, is proposed to discriminate vehicles in joint domains. Besides, for the VehicleID dataset, a coupled cluster loss (CCL) and a mixed difference network structure (Mixed Diff) are introduced in [94].

Since the VeRi dataset has multiple available views of some vehicles in the training set, in addition to the model trained on the ShapeNet, we also fine-tune the ABLN-16 version using selected 255 vehicles with 16 views. We report the results as ABLN-Ft-16. Moreover, due to the only two available views in the VehicleID dataset, we only adopt the ABLN-32 without fine-tuning for comparison.

As demonstrated in Figure 5.7, our methods are dominant compared to state-of-the-art re-ID models. It can be seen from Table 5.4 that, for the VeRi dataset, ABLN-Ft-16 obtains the highest mAP which increases 4.11% compared with ABLN-16. It means there still exists huge improvement space if large multi-view real vehicle data are available for training our model. Without fine-tuning, ABLN-32 outperforms previous vehicle re-ID methods FACT and GoogLeNet by 4.37% and 5.33% in mAP, respectively, since they do not address the huge variations across different views of a vehicle. None of Improved Deep, DGD or LOMO designed for person re-ID gets satisfied results. Moreover, consistent improvement by our method is shown on the VehicleID dataset. ABLN-32 increases the rank-1 and rank-5 matching accuracies by 3.7% and 4.86%, respectively, compared to the second place Mixed Diff+CCL. FACT performs poorly, since it fails to discriminate small inter-variations between vehicles in the same viewpoint.

Table 5.4 Comparisons (%) of our method with state-of-the-arts.

	Methods	rank-1	rank-5	rank-20	rank-50	mAP
VeRi	Improved Deep [1]	47.70	62.65	76.05	85.18	17.13
	DGD [149]	50.51	68.86	80.05	87.62	17.96
	LOMO [88]	23.89	40.32	58.61	73.96	9.03
	GoogLeNet [153]	51.98	66.79	78.77	86.37	17.58
	FACT [99]	52.35	67.16	79.97	87.09	18.54
	ABLN-32	58.14	74.41	85.87	92.35	22.91
	ABLN-16	55.51	72.03	84.02	90.97	20.81
	ABLN-Ft-16	**60.49**	**77.33**	**88.27**	**94.08**	**24.92**
VehicleID	Improved Deep [1]	42.73	60.92	77.57	89.09	-
	DGD [149]	44.72	66.68	81.35	90.31	-
	VGG+CCL [94]	43.62	64.84	80.12	89.29	-
	Mixed Diff+CCL [94]	48.93	75.65	88.47	93.37	-
	FACT [99]	39.85	58.47	74.98	86.36	-
	ABLN-32	**52.63**	**80.51**	**91.25**	**95.05**	-



(a) (b)

Fig. 5.7 Comparisons of CMC curves with state-of-the-arts. (a) Results on the VeRi dataset. (b) Results on the VehicleID dataset.

5.4 Viewpoint-aware Attentive Multi-view Inference

In this section, another model called viewpoint-aware attentive multi-view inference (VAMI) is presented to infer multi-view features from single-view image inputs. Then, distance metrics can be learned on the generated viewpoint-invariant multi-view feature space. The main contributions of the VAMI are highlighted as follows: (1) A viewpoint-aware attention model is proposed to obtain attention maps from the input image. The high-scored region of each map shows the overlapped appearance between the input vehicle's view and a target viewpoint. For instance, to infer the side view feature from a front-side view input image, the VAMI will only pay attention to the vehicle's side pattern while ignoring the front region. (2) Given the attentive features of a single-view input, we design a conditional multi-view generative network to infer a global feature containing different viewpoints' information of the input vehicle. The adversarial training mechanism and auxiliary vehicle attribute classifiers are combined to achieve effective feature generation.

5.4.1 Problem Formulation

The Eq. 4.1 is modified as the following the function:

$$\mathbf{f} = T(\text{concat}(\{\mathbf{x}_v\}_{v=1}^{V})) = T(\text{concat}(F(\mathbf{I}) \cdot \{\alpha_v\}_{v=1}^{V})). \tag{5.7}$$

The operator $F(\cdot)$ is to extract the feature of the input image $\mathbf{I}$. $\{\alpha_v\}_{v=1}^{V}$ is obtained by the viewpoint-aware attention model to select overlapped regions between the input view and a target viewpoint v, where V is the defined number of viewpoints. Moreover, the operator $T(\cdot)$ denotes the transformation from the concatenated attentive single-view features $\{\mathbf{x}_v\}_{v=1}^{V}$ to the inferred multi-view features. Therefore, the most essential point to achieve effective re-ID is how to design and optimize the $F(\cdot)$, α and $T(\cdot)$.

5.4.2 Network Architecture

Our proposed viewpoint-aware attentive multi-view inference network mainly consists of four important components. The network architecture is illustrated in Fig. 5.8. Learning $F(\cdot)$ for extracting vehicles' single-view features is first addressed by training a deep CNN using vehicles' attribute labels. To obtain viewpoint-aware attention maps α for extracting core regions targeting at different viewpoints from the input view, corresponding viewpoint embeddings are adopted to attend to one intermediate layer of the F Net. Exploiting the

attentive feature maps for different viewpoints as conditions, we aim to generate multi-view features by $T(\cdot)$ with an adversarial training architecture. During training, features extracted from real images in various viewpoints of the input vehicle are used for the real data branch, but this branch is no longer needed in the testing phase. The discriminator simultaneously distinguishes the generated multi-view features from the real ones and adopts auxiliary vehicle classifiers to help match the inferred features with the correct input vehicle's identities. Finally, given pairwise image inputs, a contrastive loss is configured at the end to optimize the network embedded with distance metric learning. The details of each component are clearly explained in the following four sub-sections.

5.4.3 Vehicle Feature Learning

The F Net is built with a deep CNN module for learning vehicles' intrinsic features containing vehicles' model, color and type information. Its backbone deploys five convolutional (*conv*) layers and two fully-connected (fc) layers. The first two *conv* layers are configured with 5×5 kernels, while the following three *conv* layers are set with 3×3 kernels. Stride is set to 4 for the first *conv* layer and 2 for the remaining *conv* layers. The Leaky-ReLU is set after each layer with the leak of 0.2. Detailed hyper-parameters' settings are illustrated in the bottom-left part of Fig. 5.8.

In addition to two 1024-dimensional fc layers connected with multi-attributes classification, one more 256-dimensional fc layer is configured for viewpoint classification. All the vehicle images are coarsely categorized into five viewpoints ($V = 5$) as front, rear, side, front-side and rear-side which are enough to describe a vehicle comprehensively. After training the F Net, we can extract viewpoint features over all the training data and easily learn five viewpoints' feature clusters by k-means clustering, thus the feature in the center of each cluster called central viewpoint feature can be obtained. These central viewpoint features are used for learning the viewpoint-aware attention model.

5.4.4 Viewpoint-aware Attention Mechanism

Visual attention models can automatically select salient regions and drop useless information from features. In the vehicle re-ID problem, our model requires focussing on the overlapped visual pattern of vehicles between the input viewpoint and the target viewpoint. For instance, to tell the difference between two similar vehicles from the front-side and rear-side viewpoints, humans usually will pay attention to their shared side appearance to discriminate whether the two vehicles are the same or not. The top-right part of Fig. 5.9 shows some examples. Thus, we aim to address this problem by proposing a viewpoint-aware attention model.

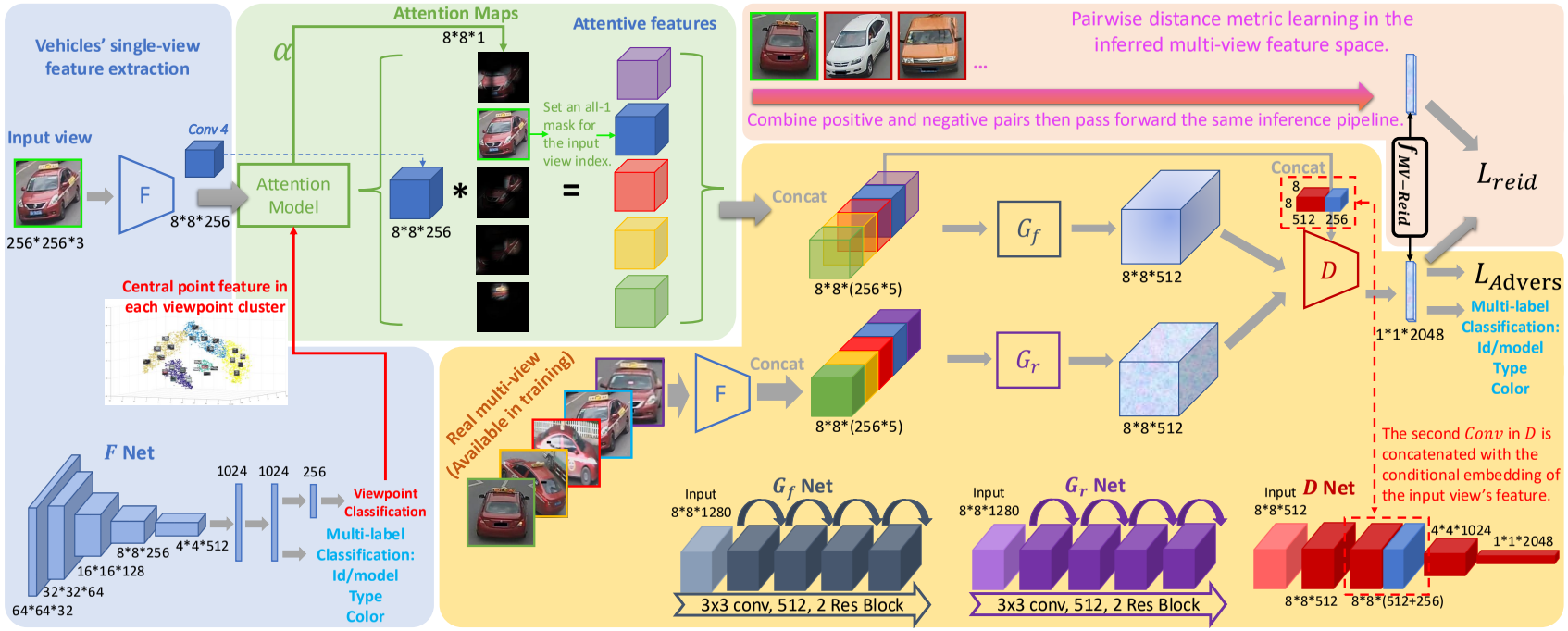


Fig. 5.8 An overview of the architecture of VAMI. The F Net is for learning single-view features containing vehicles' intrinsic information such as model, color and type. Moreover, viewpoint features can be also learned so that the central point feature of each viewpoint cluster over the whole training set can be obtained and used for attention learning. The attention model aims to output viewpoint-aware attention maps from the input view image targeting at different viewpoints. To infer multi-view features from the obtained attentive single-view features, we design a conditional generative network trained by an adversarial architecture. The networks of the real data branch are only available in the training phase. Auxiliary vehicle classifiers are configured at the end of D to help match the inferred multi-view features with correct input vehicles' identities. Finally, given positive and negative vehicle pairs, a contrastive loss is designed to optimize the network for distance metric learning. (Best viewed in color.)

Fig. 5.9 illustrates the underlying design of our attention mechanism. In order to extract feature vectors of different regions, we select the *Conv*4 layer of the F Net since it has high-level perceptrons and keeps a large enough spatial size. Thus, the input image is represented as $\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N\}$, where N is the number of image regions and $\mathbf{u}_n$ is a 256-dimensional feature vector of the n-th region. Our model performs viewpoint-aware attentions by multiple steps. Attention mechanism at each step can be considered as a building block. An attention map can be produced by learning a context vector weakly supervised by labels indicating shared appearance between the input and target viewpoints.

The context vector at step t can attend to certain regions of the input view by the following equation:

$$\mathbf{c}^t = \mathbf{attention}(\mathbf{c}^{t-1}, \{\mathbf{u}_n\}_{n=1}^N, \mathbf{v}), \tag{5.8}$$

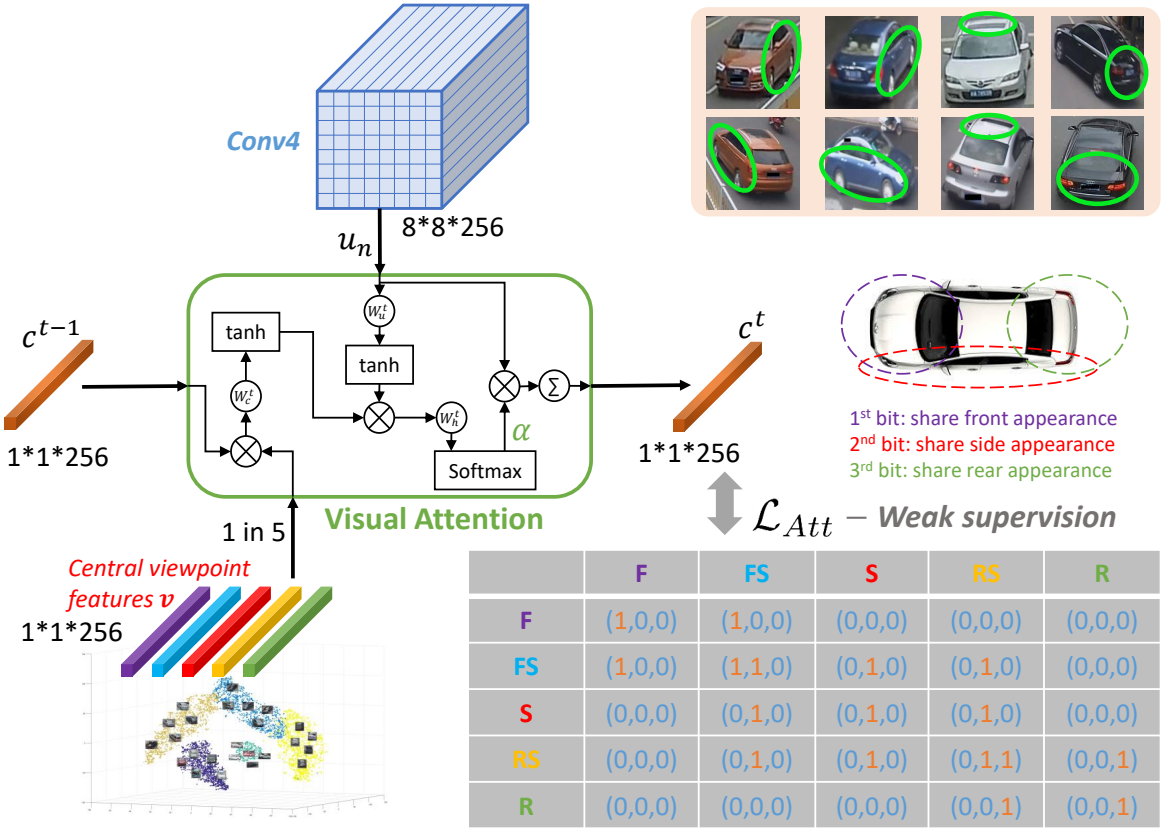
Where $\mathbf{c}^{t-1}$ is the context vector at step $t-1$ and $\mathbf{v}$ denotes one of the five central viewpoint features. The soft attention mechanism is adopted that a weighted average of all the input feature vectors is used for computing the context vector. The attention weights $\{\alpha_n^t\}_{n=1}^N$ are calculated through two layer non-linear transformations and the softmax function:

$$\begin{aligned}
&\mathbf{e}^{t-1} = \mathbf{c}^{t-1} \cdot \mathbf{v}, \\
&\mathbf{h}_n^t = \tanh(\mathbf{W}_c^t \mathbf{e}^{t-1} + \mathbf{b}_c^t) \cdot \tanh(\mathbf{W}_u^t \mathbf{u}_n + \mathbf{b}_u^t), \\
&\alpha_n^t = \mathrm{softmax}(\mathbf{W}_h^t \mathbf{h}_n^t + \mathbf{b}_h^t), \\
&\mathbf{c}^t = \sum_{n=1}^{N} \alpha_n^t \mathbf{u}_n,
\end{aligned} \tag{5.9}$$

Where $\mathbf{W}_c^t$, $\mathbf{W}_u^t$, $\mathbf{W}_h^t$ and bias terms are learnable parameters. $\mathbf{h}_n^t$ is the hidden state and $\cdot$ denotes the element-wise multiplication. The context vector $\mathbf{c}^0 (t=0)$ is initialized by:

$$\mathbf{c}^0 = \frac{1}{N} \sum_{n=1}^{N} \mathbf{u}_n. \tag{5.10}$$

Learning this viewpoint-aware attention model is mainly weakly supervised by the shared appearance region's labels between the input and target viewpoints. We design three-bit binary codes to encode the view-overlap information as shown in the bottom-right matrix of Fig. 5.9. The first bit is set as 1 when the two viewpoints share the front appearance,



	F	FS	S	RS	R
F	(1,0,0)	(1,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
FS	(1,0,0)	(1,1,0)	(0,1,0)	(0,1,0)	(0,0,0)
S	(0,0,0)	(0,1,0)	(0,1,0)	(0,1,0)	(0,0,0)
RS	(0,0,0)	(0,1,0)	(0,1,0)	(0,1,1)	(0,0,1)
R	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,1)	(0,0,1)

Fig. 5.9 The details of the viewpoint-aware attention model. The top-right part gives examples of overlapped regions of certain arbitrary viewpoint pairs.

while the second and third bits denote whether the side and rear appearances are shared or not, respectively. The attention loss $\mathscr{L}_{Att}$ is optimized by the cross-sentropy. Specifically, if the input vehicle image is the front-side viewpoint and the target viewpoint is rear-side, the central viewpoint feature of rear-side will be adopted as the $\mathbf{v}$ and the supervision codes will be $(0,1,0)$ since the two viewpoints only share the side appearance region. Then, once the attention model is trained, it outputs an attention map only giving a high response on the side appearance of the input vehicle. Moreover, for certain cases where none of the front, side or rear appearance is overlapped between viewpoint pairs (i.e. $(0,0,0)$), it is surprisingly observed that the top appearance would be attended, which is shown in the sub-section of experiments.

Since the target is to infer multi-view features containing all the five viewpoints' information from the input view, as illustrated in the green curly brackets of Fig. 5.8, we extract the input view's *Conv*4 feature maps and output corresponding attention maps $\{\alpha_v\}_{v=1}^{V}$ for other four viewpoints. The feature maps of the input view are masked by different viewpoints'

attention maps. Then, these intermediate attentive feature maps $\{\mathbf{x}_v\}_{v=1}^{V}$ are concatenated as conditional embeddings to further infer multi-view features.

5.4.5 Adversarial Multi-view Feature Learning

Instead of generating real images by normal GANs, our model aims to transform single-view features into multi-view features by a generative model. Two networks for both the fake path and the real path are designed as G_f and G_r, respectively. The input of G_f is the concatenated attentive feature $\{\mathbf{x}_v\}_{v=1}^{V}$ of the input single-view image in which the noise is embedded in the form of dropout. The input of G_r is the real features $\{\bar{\mathbf{x}}_v\}_{v=1}^{V}$ of images from different viewpoints of the same vehicle identity with G_f. The G_r is designed mainly for better fusing and learning a real high-level multi-view feature of the input vehicle.

Since we do not need to generate images by gradually enlarging the spatial size of feature maps but infer high-level multi-view features, G_f and G_r are proposed with residual transformation modules rather than adopting deconvolutional layers. The residual transformation module consists of four residual blocks whose hyper-parameters are shown in Fig. 5.8. The advantage of using residual blocks is that the networks can better learn the transformation functions and fuse features of different viewpoints by a deeper perceptron. Moreover, G_f and G_r have the same architecture but do not share the parameters since they are set for different purposes. We tried to set G_f and G_r sharing parameters, but the model failed to converge since the inputs of the two paths have a huge difference.

The discriminative net D employs a general fully convolutional network to distinguish the real multi-view features from the generated ones. Rather than maximizing the output of the discriminator for generated data, the objective of feature matching [123] is employed to optimize G_f to match the statistics of features in an intermediate layer of D. The adversarial loss is defined in the following equation:

$$\begin{aligned}\mathscr{L}_{Advers} &= \max_{D}(\mathbb{E}(\log(D(G_r(\{\bar{\mathbf{x}}_v\}_{v=1}^{V}))))) \\ &+ \mathbb{E}(\log(1-D(G_f(\{\mathbf{x}_v\}_{v=1}^{V}))))) \\ &+ \min_{G_f}||\mathbb{E}(D_m(G_r(\{\bar{\mathbf{x}}_v\}_{v=1}^{V}))) - \mathbb{E}(D_m(G_f(\{\mathbf{x}_v\}_{v=1}^{V})))||_2^2,\end{aligned} \tag{5.11}$$

Where m means the m^{th} layer in D ($m = 4$ in our setting). Moreover, D is trained with auxiliary vehicles' multi-attributes classification to better match inferred multi-view features with input vehicles' identities. The architecture of D is shown in Fig. 5.8. The second *conv* layer is concatenated with the input single-view feature maps to better optimize the

conditioned G_f and D. Then, we apply two more *conv* layers to output the final multi-view feature $\mathbf{f}_{MV\_Reid}$ which is a 2048-dimensional feature vector. The final *conv* layer deploys the 4×4 kernels while others use 3×3 kernels. For all the *conv* layers in G_f, G_r and D, we adopt Leaky-ReLU activation and batch normalization. The pre-activation proposed in [59] is implemented for residual blocks.

In the training phase, in addition to optimizing the $\mathscr{L}_{Advers}$, the $\mathscr{L}_{Reid}$ defined in Sec. 4.3.1 is configured to make the model learning with distance metrics given positive and negative vehicle image pairs. Learning $\mathscr{L}_{Reid}$ is based on the $\mathbf{f}_{MV\_Reid}$ inferred from the single-view input rather than corresponding real multi-view inputs. Our distance metric learning is more reasonable since the generated multi-view feature space is viewpoint-invariant. In the testing phase, only single-view inputs are available. Given any image pair in arbitrary viewpoints, each image can pass forward the F, G_f and D to infer the $\mathbf{f}_{MV\_Reid}$ containing all viewpoints' information of the input vehicle, then the Euclidean distance between the pair can be computed for the final re-ID ranking.

5.4.6 Optimization

The training scheme for VAMI consists of four steps. In the first step, the F Net for vehicle feature learning is trained using softmax classifiers. Then, the computed five central viewpoint features are used for training the viewpoint-aware attention model by $\mathscr{L}_{Att}$. In the second step, the G_r for learning the real multi-view features from five viewpoints' inputs needs to be pre-trained by auxiliary vehicles' multi-attributes classification together with D. Otherwise, optimizing the G_f, G_r and D together at the early stage will make the $\mathscr{L}_{Advers}$ unstable since the fused real data distribution in the adversarial architecture has not been shaped. Once the G_r is trained, we fix it. In the following step, the conditioned G_f and D nets can be optimized by $\mathscr{L}_{Advers}$ to infer multi-view features from single-view inputs. Finally, the pairwise loss $\mathscr{L}_{Reid}$ is added to fine-tune the whole network except for F and G_r to learn distance metrics, since at the early training stage the inferred multi-view features are poor so that the $\mathscr{L}_{Reid}$ cannot contribute to the optimization.

5.4.7 Experiments and Results

We first qualitatively demonstrate the viewpoint-aware attention model. Then, ablation studies and comparisons with state-of-the-art vehicle re-ID methods are evaluated on the VeRi [99] and VehicleID [94] datasets.

VeRi-776 and Training Details

Experiments are mainly conducted on the VeRi-776 dataset since each vehicle has multiple viewpoints' images so that we can fully evaluate the effectiveness of our VAMI. The VeRi dataset contains 776 different vehicles captured in 20 cameras along a circular road within a city area. The whole dataset is split into 576 vehicles with 37,778 images for training and 200 vehicles with 11,579 images for testing. An additional set of 1,678 images selected from the test vehicles are used as query images. We strictly follow the evaluation protocol proposed in [99]. Since an image-to-track search is proposed, in addition to the Cumulative Matching Characteristic (CMC) curve, a mean average precision (mAP) is also adopted for evaluation [99].

To train the model, the ADAM Optimizer is adopted with the empirical learning rate of 0.0002 and the momentum of 0.5. The mini-batch size is set as 128. Training of the F Net and viewpoint-aware attention model are stopped after 30 and 35 epochs, respectively, when the losses converge to stable values. Moreover, we first pre-train the G_r and D by 50 epochs and then start the adversarial learning with the G_f for 200 epochs. Finally, we randomly combine 10k positive pairs and 30k negative pairs and add the $\mathscr{L}_{Reid}$ loss for a joint training by additional 50 epochs.

Qualitative Attention Map Results

Before evaluating the re-ID results, we first qualitatively demonstrate the effectiveness of the viewpoint-aware attention model. Fig. 5.10 shows some examples of attention maps achieved by our model. For instance, if the viewpoint of the input image is front-side and the target viewpoint is side, the central viewpoint feature of the side view will be used to attend to the side appearance region of the input view image. Then, only the feature in this region is selected for further multi-view feature inference. The effectiveness of this attention model for multi-view vehicle re-ID has been evaluated by the ablation study.

Ablation Studies

Effect of Multi-View Inference

The primary contribution needed to be investigated is the effectiveness of the multi-view feature inference for vehicle re-ID. We compare the VAMI with three baselines. The first one simply adopts the feature of the original input view image extracted from the second fully-connected layer of the F Net. The second one adds a $\mathscr{L}_{Reid}$ to learn distance metrics based on the single-view features. Moreover, we also drop the $\mathscr{L}_{Reid}$ of the VAMI as a baseline

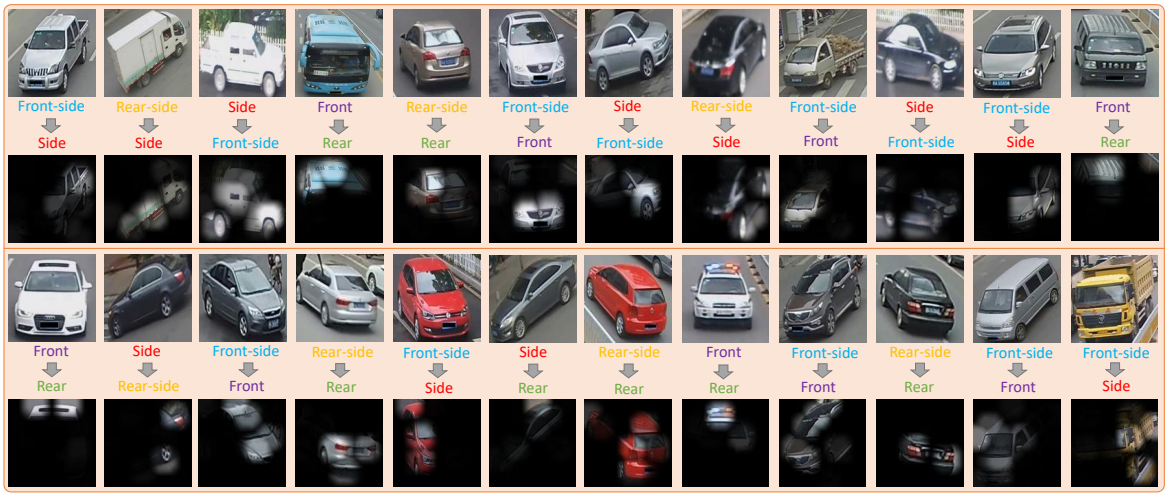


Fig. 5.10 Viewpoint-aware attention maps. The upper row shows the input images and the bottom row shows the output attention maps. The highly-responded region is obtained by the input view attended with the central viewpoint feature of the target viewpoint.

to explore the improvement by metric learning on the multi-view features. Fig. 5.12(a) illustrates CMC curves of different approaches.

As shown in the upper half of Table 5.5, the mAP increases 13.3% by inferred multi-view features compared with original single-view features. Such a huge improvement shows the proposed multi-view inference indeed benefits the vehicle re-ID from arbitrary viewpoints. Optimizing $\mathscr{L}_{Reid}$ on the multi-view feature space has a further gain of 8.19% which shows the distance metric learning performed on the viewpoint-invariant feature space is more suitable. The ranking results demonstrate the similar tendency. Moreover, Fig. 5.11 compares qualitative results that most gallery candidates ranked in top positions by directly adopting single-view feature based learning usually have similar viewpoints with query ones, but more candidates with different viewpoints can be proposed by our VAMI and correct hits in the top-10 ranks become much more as well. The last two rows in the column of results by VAMI give the failure cases where some gallery candidates have highly similar appearances from the same viewpoint with the query vehicle images. In Fig. 5.11, the features of images of 20 test vehicles are also visualized. It shows samples of the same vehicle in different viewpoints are scattered based on the single-view features, but clustered after multi-view inference by VAMI.

Effect of Attention Model

To transform features across different viewpoints, we only need to attend to regions of the input view containing the appearance overlapped with target viewpoints while ignoring other useless regions. The viewpoint-aware attention model is dropped to explore its significance

Table 5.5 Evaluation (%) of effectiveness of the multi-view inference (MV Infer.) and adversarial network (Advers. Net.).

	Baselines	mAP	r=1	r=5	r=20
MV Infer.	Single-view feat	28.64	63.52	78.69	87.13
	Single-view feat + $\mathscr{L}_{Reid}$	32.59	66.21	80.63	89.86
	Multi-view feat	41.94	71.51	85.69	93.66
	Multi-view feat + $\mathscr{L}_{Reid}$	**50.13**	**77.03**	**90.82**	**97.16**
Advers. Net.	Regular objective for G_f	41.59	74.26	86.51	92.55
	No auxiliary classifiers for D	33.43	69.54	79.34	88.46
	Regular CNN for G_f and G_r	42.89	72.95	84.66	92.82
	ℓ_2 loss	34.96	67.92	82.60	91.48
	VAMI	**50.13**	**77.03**	**90.82**	**97.16**

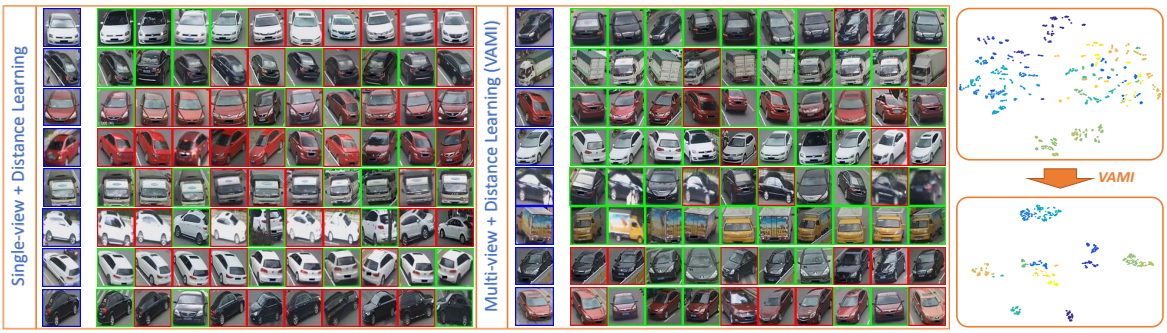
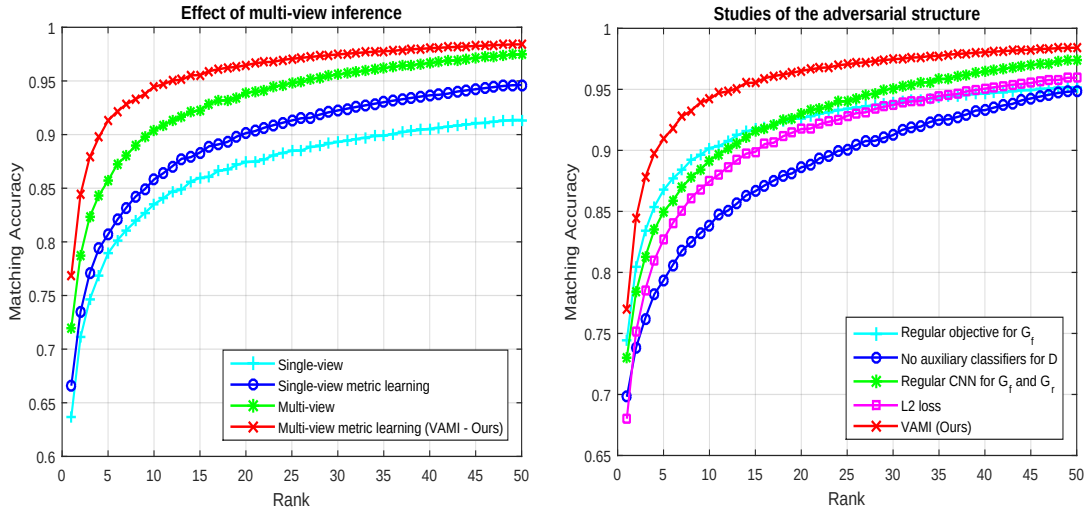


Fig. 5.11 The left part compares qualitative results (Top-10 ranks) of single-view feature+$\mathscr{L}_{Reid}$ and our VAMI. Blue boxes are query vehicles, while the green and red boxes denote correct hits and incorrect ones, respectively. The right part visualizes the spaces of the original single-view features and the multi-view features inferred by our VAMI.



(a) (b)

Fig. 5.12 (a) CMC results of evaluation of the multi-view inference. (b) CMC results of studies of the adversarial structure.

in this baseline. The input view's *Conv*4 feature maps are simply concatenated with a same size noise volume as the input for the G_f. Table 5.6 shows the mAP largely decreases 10.01% if we drop the attention model. However, the non-attentive multi-view feature inference module can still outperform the single-view based baselines.

Our attention model can be built by k steps in depth. Thus, the variable k is evaluated to explore the best performance. Table 5.6 shows the highest mAP is achieved when k equals 2. Moreover, since the noise is provided in the form of dropout for G_f, we also study its effect on the attentive multi-view feature inference by varying the dropout rate n. Through experiments, dropout rate as 0.3 gets the best re-ID results. Too small noise embedding makes the generator become deterministic, while too much noise weakens the attentive features degrading the model to the non-attentive version. Neither of such two cases gets satisfactory performance.

Adversarial Structure Studies

To study each designed component in the adversarial multi-view feature generation module, we explore them individually to compare the results.

Regular objective for G_f. Traditional objective for updating the generator in an adversarial architecture is to directly maximize the output of the discriminator, which usually overtrains the discriminator. The training of the G_f and D is unstable. We compare it as a baseline with our objective of feature matching.

No auxiliary classifiers for D. Configuring auxiliary vehicles' classifiers can make the adversarial networks learned with vehicles' intrinsic features. It helps to match the inferred

Table 5.6 Evaluation (%) of attention model. k is the number of the attention step. n is the noise rate in the form of dropout.

Baselines	mAP	r = 1	r = 5	r = 20
VAMI w/o attention	40.12	69.31	82.81	91.34
VAMI	**50.13**	**77.03**	**90.82**	**97.16**
k = 1	43.78	73.86	88.27	95.42
k = 2	**50.13**	**77.03**	**90.82**	**97.16**
k = 3	45.60	74.54	88.31	95.72
k = 4	41.05	70.23	83.44	91.90
n = 0.1	42.33	71.76	86.13	93.94
n = 0.2	43.25	73.32	87.98	95.10
n = 0.3	**50.13**	**77.03**	**90.82**	**97.16**
n = 0.4	47.68	75.19	88.97	96.04
n = 0.5	45.54	74.62	88.42	95.68
n = 0.6	41.59	71.49	85.45	93.30

multi-view features with correct identities of the input vehicles. Otherwise, the generated features can be close to real features but do not match input identities, which is a destructive weakness for the re-ID task.

Regular CNN for G_f and G_r. To evaluate the advantage of the designed residual transformation module, we compare it with a regular CNN without identity mapping. In this baseline, we configure convolutional layers with the same settings of hyper-parameters both for the G_f and G_r.

ℓ_2 loss. Our overall aim is to transform single-view features into multi-view features. To explore the ability to generate real features by our adversarial network, an ℓ_2 loss is adopted instead of $\mathscr{L}_{Advers}$ at f_{MV_Reid} to minimize the distance between features of the single-view and real multi-view paths. Noise embedding in G_f is dropped in this case.

Fig. 5.12(b) compares the CMC curves of different baselines. As shown in the bottom half of Table 5.5, the final proposed VAMI outperforms all the baselines by large margins, which validates each carefully designed component is effective to benefit the multi-view vehicle re-ID task.

Comparisons with State-of-the-arts

Evaluation on the VeRi-776 dataset

We compare the VAMI with state-of-the-art vehicle re-ID methods. LOMO [88] is a hand-crafted local feature first proposed for person re-ID. It aims to address the problem against viewpoint and illumination variations. DGD [149] is a method which can learn generic and robust deep features with data from multiple domains. We transfer the model

from humans to vehicles by re-training on the VeRi and VehicleID datasets. The GoogLeNet fine-tuned on the CompCars dataset [153] can be used for extracting great visual descriptors containing rich semantic features for vehicles. FACT [99], consisting of SIFT, Color Name, and GoogLeNet features, is proposed to discriminate vehicles in joint domains. XVGAN in Chapter 4 proposes to generate cross-view images from the input view of a vehicle, then combines the original and generated views to compute distances. Siamese-Visual [126] is proposed to learn vehicles' features computing a pairwise visual similarity by classification cross-entropy loss. Moreover, OIFE [145] aims to align local region features of different viewpoints based on key points.

The CMC curves are shown in Fig. 5.13 and detailed mAP results are listed in Table 5.7. In all the vision-based methods, our VAMI achieves the best performance over the second place which also has an orientation-based region module, with 2.13% mAP increase. The key point alignment of OIFE does not work well for large viewpoint variations. The Siamese-Visual simply adopts a pairwise deep CNN for distance metric learning but does not include vehicles' semantic attributes learning. XVGAN focuses on image generation so that the re-ID results are limited by the blurred image quality and small resolution. All the other methods are hugely beaten by the VAMI. Moreover, we also combine our model with the spatial-temporal relations (STR [99]), which still outperforms other ST-based methods.

Evaluation on the VehicleID dataset

The VehicleID dataset consists of the training set with 110,178 images of 13,134 vehicles and the test set with 111,585 images of 13,133 vehicles. However, the dataset only contains two viewpoints: front and rear. Thus, we drop the attention model and transfer the multi-view feature inference module into a two-view version. The *Conv*4 feature maps of the input view are concatenated with a same size noise volume for the input of G_f. Corresponding real images of each vehicle's two viewpoints are set for the G_r in the training phase. Table 5.7 shows our proposed multi-view feature inference obtains dominant performance over other approaches consistently in three settings of the gallery size.

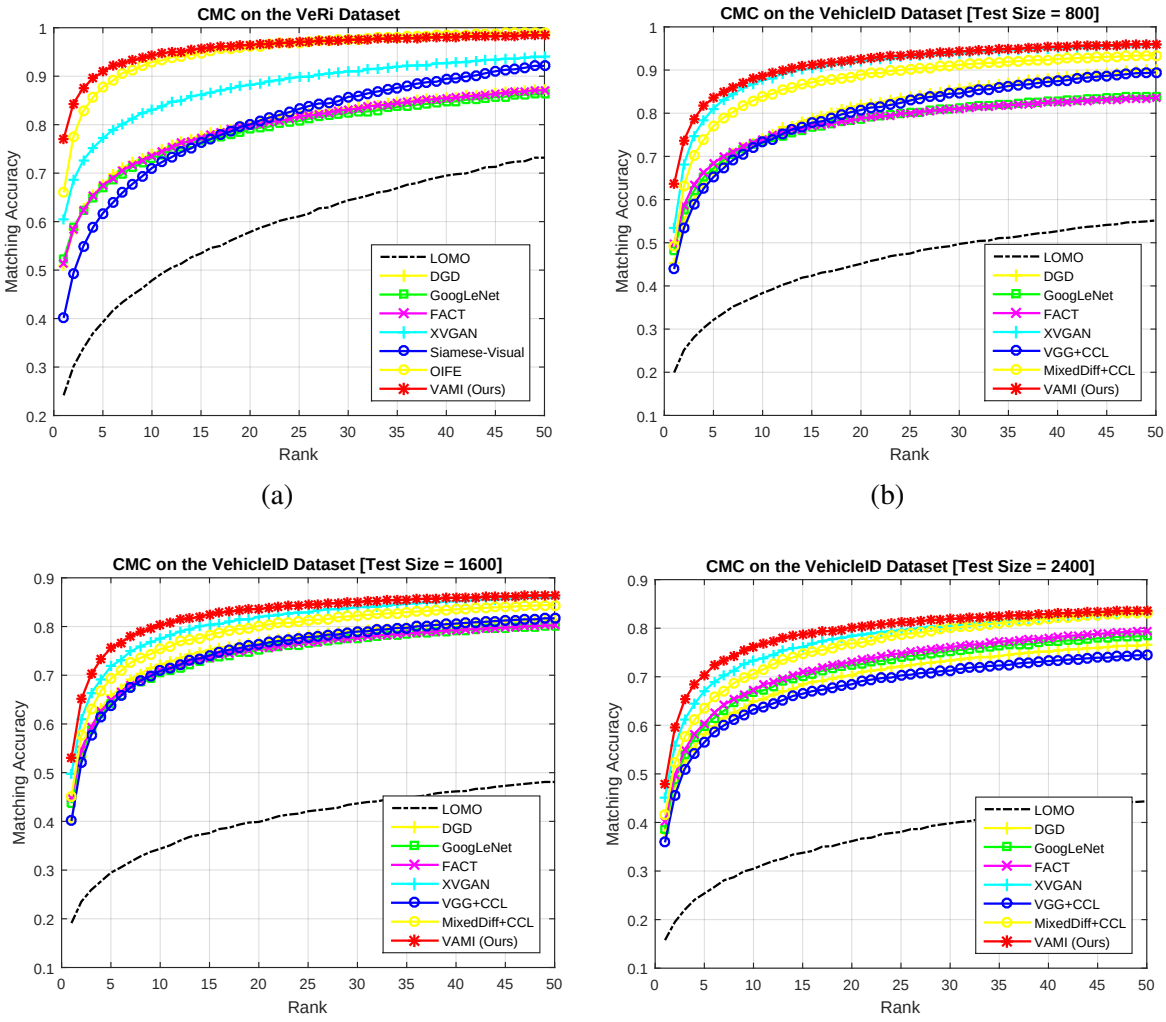


Fig. 5.13 CMC curve comparisons of different vision-based vehicle re-ID methods.

Table 5.7 Comparisons (%) with state-of-the-art re-ID methods. Methods in the last three rows include spatial-temporal (ST) information.

VeRi					VehicleID									
Settings	Query = 1678, Test = 11579				Settings	Test Size = 800			Test Size = 1600			Test Size = 2400		
Methods	mAP	r = 1	r = 5	r = 20	Methods	r = 1	r = 5	r = 20	r = 1	r = 5	r = 20	r = 1	r = 5	r = 20
LOMO [88]	9.03	23.89	40.32	58.61	LOMO [88]	19.76	32.01	45.04	18.85	29.18	39.87	15.32	25.29	35.99
DGD [149]	17.96	50.51	68.86	80.05	DGD [149]	44.72	66.68	81.35	40.25	65.31	76.76	37.33	57.82	70.25
GoogLeNet [153]	17.58	51.98	66.79	78.77	GoogLeNet [153]	47.88	67.18	78.46	43.40	63.86	74.99	38.27	59.39	72.08
FACT [99]	18.54	52.35	67.16	79.97	FACT [99]	49.53	68.07	78.54	44.59	64.57	75.30	39.92	60.32	72.92
XVGAN [172]	24.65	60.20	77.03	88.14	XVGAN [172]	52.87	80.83	91.86	49.55	71.39	81.73	44.89	66.65	78.04
SiameseVisual [126]	29.48	41.12	60.31	79.87	VGG+CCL [94]	43.62	64.84	80.12	39.94	62.98	76.07	35.68	56.24	68.41
OIFE [145]	48.00	65.92	87.66	96.63	MixedDiff+CCL [94]	48.93	75.65	88.47	45.05	68.85	79.88	41.05	63.38	76.62
VAMI (Ours)	**50.13**	**77.03**	**90.82**	**97.16**	VAMI (Ours)	**63.12**	**83.25**	**92.40**	**52.87**	**75.12**	**83.49**	**47.34**	**70.29**	**79.95**
SiameseCNN+PathLSTM [126]	58.27	83.49	90.04	96.03		-	-	-	-	-	-	-	-	-
SiameseVisual([126])+STR([99])	40.26	54.23	74.97	91.68	No ST information	-	-	-	-	-	-	-	-	-
VAMI (Ours) + STR([99])	**61.32**	**85.92**	**91.84**	**97.70**		-	-	-	-	-	-	-	-	-

5.5 Conclusions

In this chapter, methods of feature-level transformation for addressing multi-view vehicle re-ID are investigated. First, an Adversarial Bi-directional LSTM Network (ABLN) is proposed. The ABLN exploits the great ability of LSTM to model sequential data, and is learned by an adversarial training architecture. Moreover, we designed a Viewpoint-aware Attentive Multi-view Inference Network (VAMI), which adopts a viewpoint-aware attention model and the adversarial training architecture to implement effective multi-view feature inference from single-view input. Extensive experiments show that our models can achieve promising results and outperform state-of-the-art vehicle re-ID methods. In our future work, we expect to explore large-scale real surveillance multi-view vehicle data to enhance our model for better practical applications.

Chapter 6

Conclusions and Future Work

This chapter concludes the essential discoveries of this thesis, connecting the different research topics, and draws some plans for future work.

6.1 Discussion

Deep vision-based algorithms for vehicle-related tasks such as detection, attributes learning and re-identification, have been mainly investigated in this thesis. We studied different deep CNN architectures to do feature learning within task-specific end-to-end systems. Moreover, the idea of adversarial learning is to have two competing neural network models, which has been explored to implement effective image/feature generation. Visual attention model learning is another important finding to weight regions on images to improve the recognition accuracies. In the following sub-sections, I conclude my work with some discussions of these main discoveries.

6.1.1 CNN Feature Learning

In most of the proposed models, Convolutional Neural Network (CNN) is designed for learning vehicle images' features. CNN is a popular deep learning technique for current visual recognition tasks. Like all deep learning techniques, CNN is very sensitive to the scale and quality of the training data. Given a well-prepared dataset, CNNs are capable of surpassing humans at visual recognition tasks. However, they are still not robust to visual artifacts such as glare and noise, which humans are able to cope. The theory of CNN is still being developed and researchers are working to endow it with properties such as active attention and online memory, allowing CNNs to evaluate new items that are vastly different

from what they were trained on. This better emulates the human visual system, thus moving towards a smarter artificial visual recognition system.

6.1.2 Adversarial Learning

Generative adversarial network (GAN) is a recent unsupervised learning technique to train neural networks to generate plausible data using a zero-sum game. In addition to normal image generation, GANs can also be successfully used for many other interesting tasks. For instance, GANs are able to upscale an image [83] in a plausible way, inferring details that do not exist in the original. GANs are also able to infer the next frames in videos [87] in a realistic way, and manage to transform simplistic drawings into photorealistic sceneries [64].

I also would like to discuss some limits of GANs and compare them with other similar models. It is particularly hard to quantify how good their output is: first thing, they run unsupervised so theoretically the loss would be a good metric to watch, however, since GANs are in fact two models fighting with each other, they will both try to have the lowest loss while augmenting the other model loss. They are also exposed to the risk of generating very similar results for different noise input. GANs are very good for their generative abilities but are also used in tasks where other concurrent models are highly strong such as latent modeling in which Variational Autoencoders (VAEs) are already very good at. VAEs are good at reducing dimensions into interpretable dimensions while GANs, as said before are harder to evaluate. However, GANs will, compared to VAEs, generate less blurry output. Extensive experiments show that our adversarial models can achieve promising results and outperform state-of-the-art methods for vehicle re-ID.

6.1.3 Attention Mechanism

Following the deep learning and AI advances in 2015, many researchers have been interested in artificial attention mechanism in neural networks. A particularly studied aspect is visual attention: humans usually focus on specific parts of their visual inputs to compute the adequate responses. This principle has a large impact on neural computation as we need to select the most pertinent piece of information, rather than using all available information, a large part of it being irrelevant to compute the neural response. Therefore, a similar idea, focusing on specific parts of the input, has been applied in deep learning for recognition, image-to-text translation, image reasoning, and visual identification of objects.

In Chapter 5, we presented a soft attention model to obtain viewpoint-aware attention maps for extracting core regions targeting at different viewpoints from the input view. Soft attention is a fully differentiable deterministic mechanism that can be plugged into an existing

system, and the gradients are propagated through the attention mechanism at the same time they are propagated through the rest of the network. The other attention mechanism is the hard one comes with reinforcement learning (RL). A major problem with RL methods such as the reinforcing method is they have a high variance (in terms of the gradient of the expected reward computed) which scales linearly with the number of hidden units in the network. That is an apparent drawback, especially if we are going to build a large network. Hence, researchers prefer to look for differentiable models of attention.

6.2 Future Work

6.2.1 Capsule Networks

Up until now, CNNs have been the state-of-the-art approach of classifying images. CNNs work by accumulating sets of features at each layer. It starts off by finding edges, then shapes, then actual objects. However, the spatial relationship information of all these features is lost. If the images have rotation, tilt or any other different orientation then CNNs have poor performance. Geoffrey Hinton recently introduced Capsule Network [122] that projects human brain have modules called "capsules". These capsules are particularly good at handling different types of visual stimulus and encoding things like pose (position, size, orientation), deformation, velocity, albedo, hue, texture etc. In their paper, CapsNet has only been explored on the MNIST dataset showing a significant increase in performance in case of overlapped digits. In my future work, I would like to look into the capsule module to develop more robust feature learning models for various vision tasks.

6.2.2 Instance-level Segmentation: Beyond Detection

I would also like to extend the detection work presented in Chapter 3 to the instance-level segmentation task. Alternatively, this task is usually called scene understanding. A scene is a view of a real-world environment that contains multiples surfaces and objects, organized in a meaningful way. The goal of scene understanding is to obtain as much semantic knowledge of a given scene image as possible. This includes categorization (labeling the whole scene), object detection (predicting object locations by bounding boxes), and semantic segmentation (labeling each pixel). Due to this very general formulation, there is a wide range of applications, such as urban scene understanding for automotive applications, generic object detection, or inferring semantics of remote sensing data. Since I am focusing on the intelligent transportation systems, the urban scene understanding would be one of my next research points.

6.2.3 Attentive Image Captioning

As mentioned before, attention mechanism is an important research branch in the computer vision community, and also one of my research interests. Image captioning is a topic where attention mechanism can hugely benefit its performance. Image captioning addresses a task of generating a natural language description for an input image. A classic image captioning system would encode the image, using a pre-trained convolutional neural network that would produce a hidden state h. Then, it would decode this hidden state by using a recurrent neural network and recursively generate each word of the caption. A general framework for image captioning is illustrated in Figure 6.1.

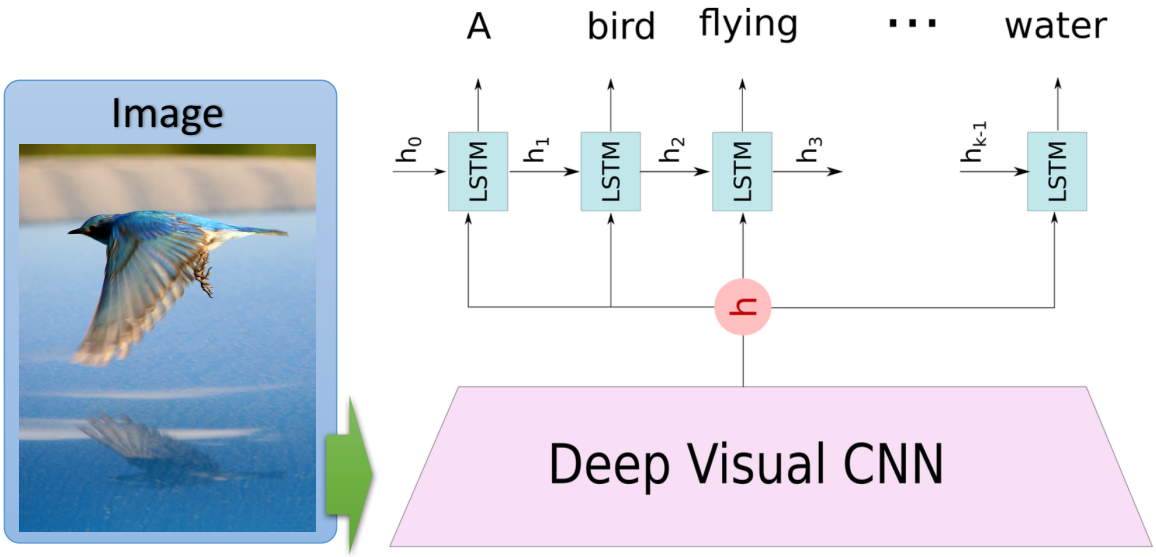


Fig. 6.1 The general framework to address the image captioning task.

The problem with this method is that, when the model is trying to generate the next word of the caption, this word is usually describing only a part of the image. Using the whole representation of the image h to condition the generation of each word cannot efficiently produce different words for different parts of the image. This is exactly where an attention mechanism is helpful. With an attention mechanism, the image is first divided into n parts, and we compute with a CNN representations of each part $h_1, ..., h_n$. When the RNN is generating a new word, the attention mechanism is focusing on the relevant part of the image, so the decoder only uses specific parts of the image. Therefore, more works will be explored in the future.

References

[1] Ahmed, E., Jones, M., and Marks, T. K. (2015). An improved deep learning architecture for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3908–3916.

[2] Ahonen, T., Hadid, A., and Pietikainen, M. (2006). Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2037–2041.

[3] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan. In *Proceedings of The International Conference on Machine Learning*.

[4] Ba, J., Mnih, V., and Kavukcuoglu, K. (2014). Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*.

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

[6] Bas, E., Tekalp, A. M., and Salman, F. S. (2007). Automatic vehicle counting from video for traffic flow analysis. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 392–397. IEEE.

[7] Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. pages 404–417. Springer.

[8] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

[9] Buch, N., Velastin, S. A., and Orwell, J. (2011). A review of computer vision techniques for the analysis of urban traffic. *IEEE Transactions on Intelligent Transportation Systems*, 12(3):920–939.

[10] Burges, C. J. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.

[11] Carreira-Perpinan, M. A. and Hinton, G. E. (2005). On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40.

[12] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.

[13] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[14] Chang, S.-L., Chen, L.-S., Chung, Y.-C., and Chen, S.-W. (2004). Automatic license plate recognition. *IEEE Transactions on Intelligent Transportation Systems*, 5(1):42–53.

[15] Chen, D., Yuan, Z., Chen, B., and Zheng, N. (2016a). Similarity learning with spatial constraints for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1268–1277.

[16] Chen, L.-C., Yang, Y., Wang, J., Xu, W., and Yuille, A. L. (2016b). Attention to scale: Scale-aware semantic image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3640–3649.

[17] Chen, W., Chen, X., Zhang, J., and Huang, K. (2017). Beyond triplet loss: A deep quadruplet network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[18] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. (2016c). Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2172–2180.

[19] Chen, Z. and Ellis, T. (2013). Efficient annotation of video for vehicle type classification. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 59–64. IEEE.

[20] Cheng, D., Gong, Y., Zhou, S., Wang, J., and Zheng, N. (2016). Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1335–1344.

[21] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[22] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[23] Dai, J., Li, Y., He, K., and Sun, J. (2016). R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, pages 379–387.

[24] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893.

[25] Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. (2007). Information-theoretic metric learning. In *Proceedings of The International Conference on Machine Learning*, pages 209–216.

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

[27] Dong, Z., Pei, M., He, Y., Liu, T., Dong, Y., and Jia, Y. (2014). Vehicle type classification using unsupervised convolutional neural network. In *Proceedings of the IEEE International Conference on Pattern Recognition*, pages 172–177.

[28] Douret, J. and Benosman, R. (2004). A multi-cameras 3d volumetric method for outdoor scenes: a road traffic monitoring application. In *Pattern Recognition, 2004. Proceedings of the 17th International Conference on*, volume 3, pages 334–337. IEEE.

[29] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

[30] Fang, J., Zhou, Y., Yu, Y., and Du, S. (2017). Fine-grained vehicle model recognition using a coarse-to-fine convolutional neural network architecture. *IEEE Transactions on Intelligent Transportation Systems*, 18(7):1782–1792.

[31] Farenzena, M., Bazzani, L., Perina, A., Murino, V., and Cristani, M. (2010). Person re-identification by symmetry-driven accumulation of local features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2360–2367. IEEE.

[32] Felzenszwalb, P., McAllester, D., and Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE.

[33] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645.

[34] Freund, Y., Schapire, R., and Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612.

[35] Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163.

[36] Fu, J., Zheng, H., and Mei, T. (2017). Look closer to see better: recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[37] Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448.

[38] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587.

[39] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2016). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142–158.

[Girshick et al.] Girshick, R. B., Felzenszwalb, P. F., and McAllester, D. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/.

[41] Goldberger, J. and Roweis, S. T. (2005). Hierarchical clustering of a mixture model. In *Advances in Neural Information Processing Systems*, pages 505–512.

[42] Gong, S., Cristani, M., Yan, S., and Loy, C. C. (2014). *Person Re-identification*, volume 1.

[43] Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

[44] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

[45] Graves, A., Mohamed, A.-r., and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE.

[46] Gray, D., Brennan, S., and Tao, H. (2007). Evaluating appearance models for recognition, reacquisition, and tracking. In *Proc. IEEE International Workshop on Performance Evaluation for Tracking and Surveillance (PETS)*, volume 3, pages 1–7. Citeseer.

[47] Gray, D. and Tao, H. (2008). Viewpoint invariant pedestrian recognition with an ensemble of localized features. *Proceedings of European Conference on Computer Vision (ECCV)*, pages 262–275.

[48] Greene, D., Cunningham, P., and Mayer, R. (2008). Unsupervised learning and clustering. In *Machine Learning Techniques for Multimedia*, pages 51–90. Springer.

[49] Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. In *Proceedings of The International Conference on Machine Learning*.

[50] Guillaumin, M., Verbeek, J., and Schmid, C. (2009). Is that you? metric learning approaches for face identification. In *Computer Vision, 2009 IEEE 12th international conference on*, pages 498–505. IEEE.

[51] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of wasserstein gans. *arXiv preprint arXiv:1704.00028*.

[52] Gunn, S. R. et al. (1998). Support vector machines for classification and regression. *ISIS technical report*, 14:85–86.

[53] Guo, G., Wang, H., Bell, D., Bi, Y., and Greer, K. (2003). Knn model-based approach in classification. In *CoopIS/DOA/ODBASE*, volume 2003, pages 986–996. Springer.

[54] Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1735–1742.

[55] Han, S., Han, Y., and Hahn, H. (2009). Vehicle detection method using haar-like feature on real time system. *World Academy of Science, Engineering and Technology*, 59:455–459.

[56] Haque, A., Alahi, A., and Fei-Fei, L. (2016). Recurrent attention models for depth-based person identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1229–1238.

[57] He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 346–361. Springer.

[58] He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[59] He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Identity mappings in deep residual networks. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 630–645. Springer.

[60] Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

[61] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

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

[63] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017a). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[64] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017b). Image-to-image translation with conditional adversarial networks. *arXiv preprint*.

[65] Jazayeri, A., Cai, H., Zheng, J. Y., and Tuceryan, M. (2011). Vehicle detection and tracking in car video based on motion model. *IEEE Transactions on Intelligent Transportation Systems*, 12(2):583–595.

[66] Jenatton, R., Mairal, J., Obozinski, G., and Bach, F. (2011). Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12(Jul):2297–2334.

[67] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, pages 675–678.

[68] Kanaci, A., Zhu, X., and Gong, S. (2017). Vehicle re-identification by fine-grained cross-level deep learning. In *British Machine Vision Conference Workshop*. BMVA.

[69] Karanam, S., Li, Y., and Radke, R. J. (2015). Person re-identification with discriminatively trained viewpoint invariant dictionaries. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4516–4524.

[70] Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137.

[71] Kim, S., Oh, S.-Y., Kang, J., Ryu, Y., Kim, K., Park, S.-C., and Park, K. (2005). Front and rear vehicle detection and tracking in the day and night times using vision and sonar sensor fusion. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2173–2178. IEEE.

[72] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations*.

[73] Kingsbury, N. (2001). Complex wavelets for shift invariant analysis and filtering of signals. *Applied and Computational Harmonic Analysis*, 10(3):234–253.

[74] Koestinger, M., Hirzer, M., Wohlhart, P., Roth, P. M., and Bischof, H. (2012). Large scale metric learning from equivalence constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2288–2295. IEEE.

[75] Köstinger, M., Hirzer, M., Wohlhart, P., Roth, P. M., and Bischof, H. (2012). Large scale metric learning from equivalence constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2288–2295.

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

[77] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

[78] Kwong, K., Kavaler, R., Rajagopal, R., and Varaiya, P. (2009). Arterial travel time estimation based on vehicle re-identification using wireless magnetic sensors. *Transportation Research Part C: Emerging Technologies*, 17:586–606.

[79] Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view rgb-d object dataset. In *2011 IEEE International Conference onRobotics and Automation (ICRA)*, pages 1817–1824.

[80] Lalimi, M. A., Ghofrani, S., and McLernon, D. (2013). A vehicle license plate detection method using region and edge based methods. *Computers & Electrical Engineering*, 39:834–845.

[81] Layne, R., Hospedales, T. M., Gong, S., and Mary, Q. (2012). Person re-identification by attributes. In *BMVC*, volume 2, page 8.

[82] LeCun, Y. et al. (2015). Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, page 20.

[83] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al. (2016). Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*.

[84] Lee, R., Hung, K., and Wang, H. (2012). Real time vehicle license plate recognition based on 2d haar discrete wavelet transform. *International Journal of Scientific & Engineering Research*, 3:1–6.

[85] Li, W., Zhao, R., Xiao, T., and Wang, X. (2014). Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 152–159.

[86] Li, X., Zhang, G., Fang, J., Wu, J., and Cui, Z. (2010). Vehicle color recognition using vector matching of template. In *International Symposium on Electronic Commerce and Security*, pages 189–193.

[87] Liang, X., Lee, L., Dai, W., and Xing, E. P. (2017). Dual motion gan for future-flow embedded video prediction. *arXiv preprint*.

[88] Liao, S., Hu, Y., Zhu, X., and Li, S. Z. (2015). Person re-identification by local maximal occurrence representation and metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2197–2206.

[89] Liao, S. and Li, S. Z. (2015a). Efficient psd constrained asymmetric metric learning for person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*.

[90] Liao, S. and Li, S. Z. (2015b). Efficient psd constrained asymmetric metric learning for person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3685–3693.

[91] Liao, S., Zhao, G., Kellokumpu, V., Pietikäinen, M., and Li, S. Z. (2010). Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1301–1306. IEEE.

[92] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 740–755. Springer.

[93] Lipton, Z. C., Berkowitz, J., and Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*.

[94] Liu, H., Tian, Y., Yang, Y., Pang, L., and Huang, T. (2016a). Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2167–2175.

[95] Liu, L., Zhou, Y., and Shao, L. (2017). Dap3d-net: Where, what and how actions occur in videos? In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 138–145. IEEE.

[96] Liu, L., Zhou, Y., and Shao, L. (2018). Deep action parsing in videos with large-scale synthesized data. *IEEE Transactions on Image Processing*.

[97] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016b). Ssd: Single shot multibox detector. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 21–37. Springer.

[98] Liu, X., Liu, W., Ma, H., and Fu, H. (2016c). Large-scale vehicle re-identification in urban surveillance videos. In *Multimedia and Expo (ICME), 2016 IEEE International Conference on*, pages 1–6. IEEE.

[99] Liu, X., Liu, W., Mei, T., and Ma, H. (2016d). A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 869–884. Springer.

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

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

[102] Ma, S., Sigal, L., and Sclaroff, S. (2016). Learning activity progression in lstms for activity detection and early detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1942–1950.

[103] Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, pages 2579–2605.

[104] Malyavej, V., Torteeka, P., Wongkharn, S., and Wiangtong, T. (2009). Pose estimation of unmanned ground vehicle based on dead-reckoning/gps sensor fusion by unscented kalman filter. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2009. ECTI-CON 2009. 6th International Conference on*, volume 1, pages 395–398. IEEE.

[105] Martínez, E., Díaz, M., Melenchón, J., Montero, J. A., Iriondo, I., and Socoró, J. C. (2008). Driving assistance system based on the detection of head-on collisions. In *IEEE Intelligent Vehicles Symposium*, pages 913–918.

[106] Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767.

[107] Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630.

[108] Mnih, V., Heess, N., Graves, A., et al. (2014). Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212.

[109] Nair, V. and Hinton, G. E. (2009). 3d object recognition with deep belief nets. In *Advances in Neural Information Processing Systems*, pages 1339–1347.

[110] Odena, A., Olah, C., and Shlens, J. (2017). Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of The International Conference on Machine Learning*, pages 2642–2651.

[111] Ozuysal, M., Lepetit, V., and Fua, P. (2009). Pose estimation for category specific multiview object localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 778–785.

[112] Palm, R. B. (2012). Prediction as a candidate for learning deep hierarchical models of data.

[113] Park, E., Yang, J., Yumer, E., Ceylan, D., and Berg, A. C. (2017). Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[114] Park, K., Lee, D., and Park, Y. (2007). Video-based detection of street-parking violation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 152–156.

[115] Piateski, G. and Frawley, W. (1991). *Knowledge discovery in databases*. MIT press.

[116] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

[117] Radford, A., Metz, L., and Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. *In International Conference on Learning Representations*.

[118] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

[119] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. In *Proceedings of The International Conference on Machine Learning*, volume 3.

[120] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99.

[121] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533.

[122] Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3857–3867.

[123] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242.

[124] Seber, G. A. and Lee, A. J. (2012). *Linear Regression Analysis*, volume 329. John Wiley & Sons.

[125] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.

[126] Shen, Y., Xiao, T., Li, H., Yi, S., and Wang, X. (2017). Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE.

[127] Shin, P., Jasso, H., Tilak, S., Cotofana, N., Fountain, T., Yan, L., Fraser, M., and Elgamal, A. (2007). Automatic vehicle type classification using strain gauge sensors. In *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pages 425–428. IEEE.

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

[129] Sivaraman, S. and Trivedi, M. M. (2010). A general active-learning framework for on-road vehicle recognition and tracking. *IEEE Transactions on Intelligent Transport System*, 11(2):267–276.

[130] Sivaraman, S. and Trivedi, M. M. (2013). Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795.

[131] Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[132] Sun, C. C., Arr, G. S., Ramachandran, R. P., and Ritchie, S. G. (2004). Vehicle reidentification using multidetector fusion. *IEEE Transactions on Intelligent Transportation Systems*, 5:155–164.

[133] Sun, Z., Bebis, G., and Miller, R. (2006a). Monocular precrash vehicle detection: Features and classifiers. *IEEE Transactions on Image Processing*, 15(7):2019–2034.

[134] Sun, Z., Bebis, G., and Miller, R. (2006b). On-road vehicle detection: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):694–711.

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

[136] Tatarchenko, M., Dosovitskiy, A., and Brox, T. (2016). Multi-view 3d models from single images with a convolutional network. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 322–337. Springer.

[137] Totten, V. F. (2008). Application of vehicle detector waveforms in vehicle reidentification and evaluating detector installation performance. *ECE Masters Theses*, page 14.

[138] Uijlings, J. R., van de Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171.

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

[140] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164.

[141] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE.

[142] Wan, J., Wang, D., Hoi, S. C. H., Wu, P., Zhu, J., Zhang, Y., and Li, J. (2014). Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 157–166. ACM.

[143] Wang, F., Zuo, W., Lin, L., Zhang, D., and Zhang, L. (2016). Joint learning of single-image and cross-image representations for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[144] Wang, X., Doretto, G., Sebastian, T., Rittscher, J., and Tu, P. (2007). Shape and appearance context modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1–8. IEEE.

[145] Wang, Z., Tang, L., Liu, X., Yao, Z., Yi, S., Shao, J., Yan, J., Wang, S., Li, H., and Wang, X. (2017). Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE.

[146] Wei, S., Jian, W., Bai-gen, C., and Qin, Y. (2009). A novel vehicle detection method based on wireless magneto-resistive sensor. In *2009 Third International Symposium on Intelligent Information Technology Application*, pages 484–487. IEEE.

[147] Weinberger, K. Q. and Saul, L. K. (2008). Fast solvers and efficient implementations for distance metric learning. In *Proceedings of The International Conference on Machine Learning*, pages 1160–1167.

[148] Weinberger, K. Q. and Saul, L. K. (2009). Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244.

[149] Xiao, T., Li, H., Ouyang, W., and Wang, X. (2016). Learning deep feature representations with domain guided dropout for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1249–1258.

[150] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057.

[151] Yan, X., Yang, J., Sohn, K., and Lee, H. (2016). Attribute2image: Conditional image generation from visual attributes. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 776–791. Springer.

[152] Yang, L., Liu, J., and Tang, X. (2014). Object detection and viewpoint estimation with auto-masking neural network. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 441–455.

[153] Yang, L., Luo, P., Change Loy, C., and Tang, X. (2015). A large-scale car dataset for fine-grained categorization and verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3973–3981.

[154] Yang, M., Han, G., Li, X., Zhu, X., and Li, L. (2011). Vehicle color recognition using monocular camera. In *Wireless Communications and Signal Processing (WCSP), 2011 International Conference on*, pages 1–5. IEEE.

[155] Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. (2016). Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–29.

[156] Yi, D., Lei, Z., Liao, S., and Li, S. Z. (2014). Deep metric learning for person re-identification. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 34–39. IEEE.

[157] Yu, H.-X., Wu, A., and Zheng, W.-S. (2017). Cross-view asymmetric metric learning for unsupervised person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*.

[158] Zapletal, D. and Herout, A. (2016). Vehicle re-identification for automatic video traffic surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–31.

[159] Zhang, L., Xiang, T., and Gong, S. (2016a). Learning a discriminative null space for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1239–1248.

[160] Zhang, Y., Li, B., Lu, H., Irie, A., and Ruan, X. (2016b). Sample-specific svm learning for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1278–1287.

[161] Zhao, L., Li, X., Zhuang, Y., and Wang, J. (2017). Deeply-learned part-aligned representations for person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*.

[162] Zhao, R., Ouyang, W., and Wang, X. (2013a). Person re-identification by salience matching. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2528–2535.

[163] Zhao, R., Ouyang, W., and Wang, X. (2013b). Unsupervised salience learning for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3586–3593.

[164] Zhao, R., Ouyang, W., and Wang, X. (2014). Learning mid-level filters for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 144–151.

[165] Zheng, H., Fu, J., Mei, T., and Luo, J. (2017). Learning multi-attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE International Conference on Computer Vision*.

[166] Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C., Wang, S., and Tian, Q. (2016a). Mars: A video benchmark for large-scale person re-identification. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 868–884. Springer.

[167] Zheng, L., Zhang, H., Sun, S., Chandraker, M., and Tian, Q. (2016b). Person re-identification in the wild. *arXiv preprint arXiv:1604.02531*.

[168] Zheng, W.-S., Li, X., Xiang, T., Liao, S., Lai, J., and Gong, S. (2015). Partial person re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4678–4686.

[169] Zhong, Z., Zheng, L., Cao, D., and Li, S. (2017). Re-ranking person re-identification with k-reciprocal encoding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[170] Zhou, T., Tulsiani, S., Sun, W., Malik, J., and Efros, A. A. (2016a). View synthesis by appearance flow. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 286–301. Springer.

[171] Zhou, Y., Liu, L., Shao, L., and Mellor, M. (2016b). Dave: a unified framework for fast vehicle detection and annotation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 278–293. Springer.

[172] Zhou, Y. and Shao, L. (2017). Cross-view gan based vehicle generation for re-identification. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press.

[173] Zhu, C., Zhao, Y., Huang, S., Tu, K., and Ma, Y. (2017a). Structured attentions for visual question answering. *arXiv preprint arXiv:1708.02071*.

[174] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017b). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE.

[175] Zhu, Y., Comaniciu, D., Pellkofer, M., and Koehler, T. (2006). Reliable detection of overtaking vehicles using robust information fusion. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):401–414.

[176] Zitnick, C. L. and Dollár, P. (2014). Edge boxes: Locating object proposals from edges. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 391–405.

Appendix A

Classic Deep Convolutional Neural Network Models

AlexNet

AlexNet [76] was the one of the first deep neural networks to push ImageNet Classification accuracy by a significant increase in comparison to traditional methodologies. It is composed of 5 convolutional layers followed by 3 fully connected layers, as depicted in Figure A.1.

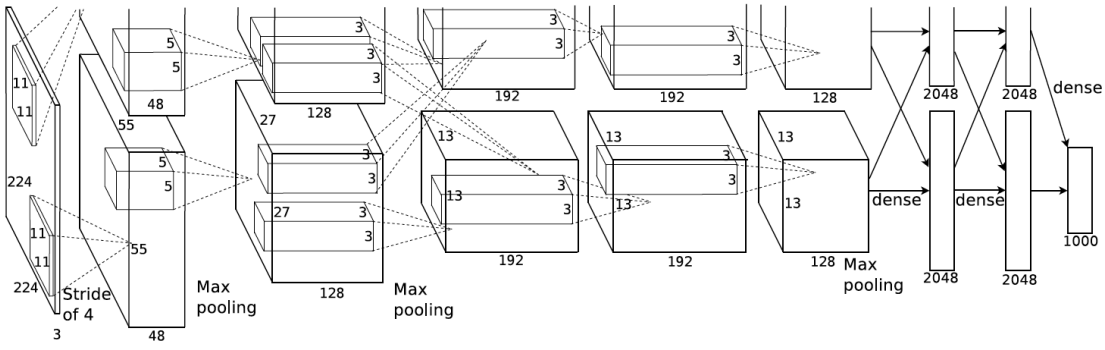


Fig. A.1 The architecture of AlexNet [76].

AlexNet, proposed by Alex Krizhevsky, uses ReLU (Rectified Linear Unit) for the non-linear part, instead of a tanh or sigmoid function which was the earlier standard for traditional neural networks. The advantage of the ReLU over sigmoid is that it trains much faster than the latter because the derivative of sigmoid becomes very small in the saturating region and therefore the updates to the weights almost vanish. This is called the vanishing gradient problem.

Another problem that this architecture solved was reducing the over-fitting by using a Dropout layer after every FC layer. Dropout layer has a probability p, associated with it and is applied at every neuron of the response map separately. It randomly switches off the activation with the probability p.

VGG-16

VGG-16 Net [128] is from the VGG group, Oxford. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. With a given receptive field (the effective area size of input image on which output depends), multiple stacked smaller size kernel is better than the one with a larger size kernel because multiple non-linear layers increases the depth of the network which enables it to learn more complex features, and that too at a lower cost. The network architecture is listed in Figure A.2.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Fig. A.2 The architecture of VGGNet [128].

GoogLeNet

While VGG achieves a phenomenal accuracy on ImageNet dataset, its deployment on even the most modest sized GPUs is a problem because of huge computational requirements, both in terms of memory and time. It becomes inefficient due to large width of convolutional layers. In a convolutional operation at one location, every output channel is connected to every input channel, and so we call it a dense connection architecture. The GoogLeNet [135] builds on the idea that most of the activations in deep network are either unnecessary (value of zero) or redundant because of correlations between them. There are techniques to prune out such connections which would result in a sparse weight connection.

GoogLeNet devised a module called inception module that approximates a sparse CNN with a normal dense construction. Since only a small number of neurons are effective as mentioned earlier, width/number of the convolutional filters of a particular kernel size is kept small. Also, it uses convolutions of different sizes to capture details at varied scales (5×5, 3×3, 1×1). Another salient point about the module is that it has a so-called bottleneck layer (1X1 convolutions). It helps in massive reduction of the computation requirement. The first version of GoogLeNet is illustrated in Figure A.3.

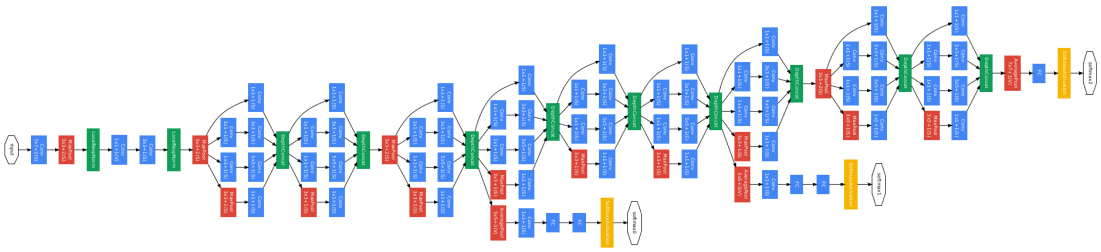


Fig. A.3 The architecture of the first version GoogLeNet [135].

ResNet

So far, we have seen that increasing the depth should increase the accuracy of the network, as long as over-fitting is taken care of. But the problem with increased depth is that the signal required to change the weights, which arises from the end of the network by comparing ground-truth and prediction becomes very small at the earlier layers, because of increased depth. It essentially means that earlier layers are almost negligible learned. This is called vanishing gradient. The second problem with training the deeper networks is, performing

the optimization on huge parameter space and therefore naively adding the layers leading to higher training error. Residual networks [58] allow training of such deep networks by constructing the network through modules called residual models as shown in Figure A.4.

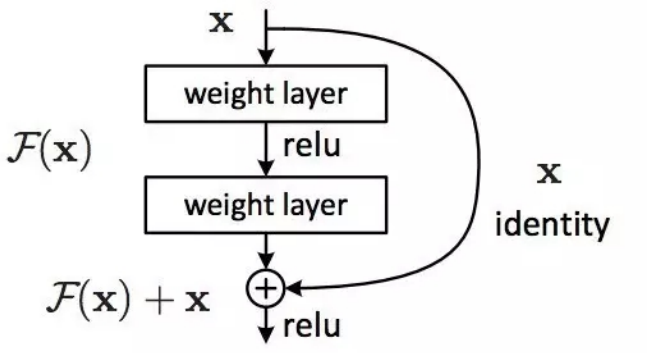


Fig. A.4 Residual learning: a building block. [58].

ResNets were learned with network depth of as large as 152. It achieves better accuracy than VGGNet and GoogLeNet while being computationally more efficient than VGGNet. ResNet-152 achieves 95.51% top-5 accuracy on the ImageNet Classification. An overall network architecture of a small version of ResNet is shown in Figure A.5.

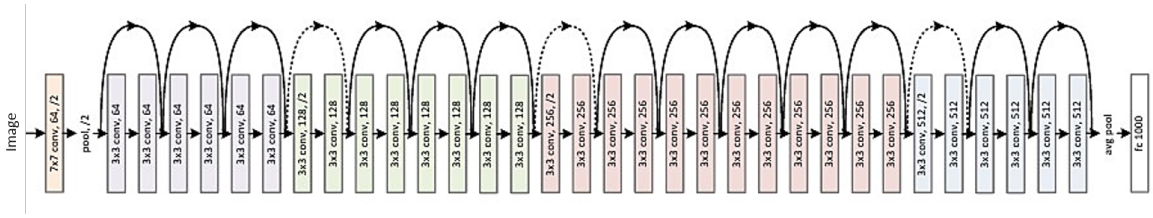


Fig. A.5 The architecture of ResNet [58].

Appendix B

Deep Learning Toolbox

B.1 Caffe

Caffe [67] is developed by Yangqing Jia who is now the lead engineering for Facebook AI platform. Caffe is the first mainstream industry-grade deep learning toolkit, started in late 2013. Due to its excellent convolutional model, it is one of the most popular toolkits within the computer vision community and won an ImageNet Challenge in 2014. Caffe is released under the BSD 2-Clause license.

Speed makes Caffe perfect for research experiments and commercial deployment. Caffe can process over 60M images per day with a single Nvidia K40 GPU. That's 1 ms/image for inference and 4 ms/image for learning and more recent library versions are faster still.

Caffe is C++ based, which can be compiled on a variety of devices. It is cross-platform and includes a port to windows. Caffe supports C++, Matlab and Python programming interfaces. Caffe has a large user community that contributes to their own deep net repository known as the "Model Zoo." AlexNet and GoogleNet are two popular user-made nets available to the community. More details can be found on http://caffe.berkeleyvision.org/.

B.2 TensorFlow

TensorFlow is currently the most successful deep learning toolbox sourced by Google. It supports a broad set of capabilities such as image, handwriting and speech recognition, forecasting and natural language processing. TensorFlow is released under the Apache 2.0 open source license in late 2015.

TensorFlow programming interfaces includes Python and C++. With the version 1.0 announcement, alpha releases of Java, GO, R, and Haskell API will be supported. Additionally, TensorFlow is supported in Google and Amazon Cloud Environment.

TensorFlow supports fine grain network layers that allows users to build new complex layer types without implementing them in a low-level language. Subgraph execution allows you to introduce and retrieve the results of discretionary data on any edge of the graph. This is extremely helpful for debugging complicated computational graphs. More details can be found on https://www.tensorflow.org/.