2016-01-28

Going Deeper with Convolutional Neural Network for Intelligent Transportation

Tairui Chen
Worcester Polytechnic Institute

Going Deeper with Convolutional Neural Network for Intelligent Transportation

by

Tairui Chen

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Electrical and Computer Engineering

by

____

Nov 2015

APPROVED:

____

Professor Xinming Huang, Major Thesis Advisor

____

Professor Yehia Massoud, Head of Department

Abstract

Over last several decades, computer vision researchers have been devoted to find good feature to solve different tasks, such as object recognition, object detection, object segmentation, activity recognition and so forth. Ideal features transform raw pixel intensity values to a representation in which these computer vision problems are easier to solve. Recently, deep features from covolutional neural network(CNN) have attracted many researchers in computer vision. In the supervised setting, these hierarchies are trained to solve specific problems by minimizing an objective function. More recently, the feature learned from large scale image dataset have been proved to be very effective and generic for many computer vision task. The feature learned from recognition task can be used in the object detection task.

This work uncover the principles that lead to these generic feature representations in the transfer learning, which does not need to train the dataset again but transfer the rich feature from CNN learned from ImageNet dataset.

We begin by summarize some related prior works, particularly the paper in object recognition, object detection and segmentation. We introduce the deep feature to computer vision task in intelligent transportation system. We apply deep feature in object detection task, especially in vehicle detection task. To make fully use of objectness proposals, we apply proposal generator on road marking detection and recognition task. Third, to fully understand the transportation situation, we introduce the deep feature into scene understanding. We experiment each task for different public datasets, and prove our framework is robust.

Acknowledgements

First of all, I would like to thank my supervisor, Professor Xinming Huang for helping me go through the entire process of the dissertation development, answering thousands of e-mails, giving me incredibly useful insights and introducing me to the huge field of deep learning.

I would also like to express my deep appreciation to all the people who assisted me with this complex project during my graduated years: My family, who has never constrained me and has always been present in times of need. My frients, who filled me with enthusiasm, motivation and love even in the hardest moments. My lab PhD students, for helping me to develop my ideas on brain, learning and artificial intelligence.

The many thinkers and friends, whose ideas and efforts have significantly contributed to shape my professional and human personality.

Contents

List of Figures

List of Tables

Chapter 1

Background

This chapter will provide a brief history and background about machine learning and computer vision, especially neural networks.

1.1 Machine Learning

Machine learning is usually divided to two types, supervised learning and unsupervised learning.

1.2 Types of machine learning

supervised learning For the supervised learning method, it learns a mapping from input X to output Y, given a labeled set of datasets (also called training sets). We talk about classification first. Assume we have a goal to learning a mapping from input X to output Y, where Y belong to 1, 2, . . . , C. If C = 2, this is called binary classification, if C ¿ 2, we call it multi-class classification. There are two major steps in supervised learning: Training: The learning phase that examines the provided data (called the training dataset) and constructs a classification model;

Testing: the model that has been built in the training phase is used to classify new unseen instances. A toy example of classification, you would like to use software to examine individual customer accounts, and for each account we need to decide if it has been hacked or compromised. We have two classes of objects which correspond to hacked or compromised. The input are individual customer accounts. These have been described by a set of D features or attributes, which are stored in an N * M matrix X. Also we have a training vector Y (hacked = 1; compromised = 0). Thus we are required to generalize beyond the training set and find which attributes belong to hacked and which attributes belong to compromised. There exist a number of supervised learning classification algorithms, for example, Decision Tree and Naive Bayes classification algorithms.

Unsupervised learning In unsupervised learning, we are just given output data, without any inputs. The goal is to discover internal connection in these data. Unlike supervised learning, these data cannot told what the desired outputs for each input. Unsupervised learning is more typical of human learning. It is more widely used than supervised learning, since it does not require a human experience (no need to labeled data). La belled data is not only expensive, but also cannot provide us with enough information. The example of unsupervised learning is clustering data into groups. X1 and X1 denotes the attributes Of input data, but they has not given outputs. It seems that there might be two clusters, or subgroups. Our goal is to estimate which cluster each point belongs to. There are three basic clustering methods: the classic K-means algorithm, incremental clustering, and the probability based clustering method. The classic k-means algorithm forms clusters in numeric domains, partitioning instances into disjoint clusters, while incremental clustering generates a hierarchical grouping of instances.

Reinforcement learning Reinforcement learning (RL) is learning by interacting with an environment. An RL agent learns from the consequences of its actions, rather than from being explicitly taught and it selects its actions on basis of its past experiences (exploitation) and also by new choices (exploration), which is essentially trial and error learning. The reinforcement signal that the RL-agent receives is a numerical reward, which encodes the success of an action's outcome, and the agent seeks to learn to select actions that maximize the accumulated reward over time.

1.3 Deep Learning

Deep learning (DL) is a branch of machine learning based on a set of algorithms that attempt to model high-level abstractions in data by using model architectures, with complex structures or otherwise, composed of multiple non-linear transformations.

Deep learning is part of a broader family of machine learning methods based on learning representations of data. An observation (for example, an image) can be represented in many ways such as a vector of numerical values, or in a more abstract way as a set of edges, regions of particular shape, etc. Some representations make it easier to learn tasks from a set of images, audios and documents. One of the great progressions of deep learning is replacing handcrafted features with efficient algorithms for unsupervised or semi-supervised feature learning and hierarchical feature extraction.

Deep learning are based on the supervised or unsupervised learning of multiple levels of features or representations of the data. Higher level features are derived from lower level features to form a hierarchical representation.

1.4 Deep Learning in Computer Vision

In 1998, Lecun proposed convolution neural network on MNIST dataset, a popular dataset for image classification. MNIST is composed of handwritten digits and includes 60000 training examples and 10000 test examples. The current best result on MNIST is an error rate of 0.23%, achieved by Ciresan et al. in 2012, the real impact of deep learning in image or object recognition, one major branch of computer vision, was felt in the fall of 2012 after the team of Geoff Hinton and his students won the large-scale ImageNet competition by a significant margin over the then-state-of-the-art shallow machine learning methods. The technology is based on 20-year-old deep convolutional nets, but with much larger scale on a much larger task, since it had been learned that deep learning works quite well on large-scale speech recognition. In 2013 and 2014, the error rate on the ImageNet task using deep learning was further reduced at a rapid pace.

Chapter 2

Introduction

In recent years, deep Convolutional Networks(ConvNets) have become the most popular architecture for large-scale image recognition tasks. The field of computer vision has been pushed to a fast, scalable and end-to-end learning framework, which can provide outstanding performance results on object recognition, object detection, scene recognition, semantic segmentation, action recognition, object tracking and many other tasks. With the explosion of computer vision research, Advanced Driver Assistance System (ADAS) has also become a main stream technology in the automotive industry. Autonomous vehicles, such as Google's self-driving cars, are evolving and becoming reality. A key component is vision-based machine intelligence that can provide information to the control system or the driver to maneuver a vehicle properly based on the surrounding and road conditions. There have been many research works reported in traffic sign recognition, lane departure warning, pedestrian detection, and etc. Car detection is a type of object detection, which is related to rich applications in car safety, surveillance, and robotics. This paper is to evaluate if the success of ConvNets is applicable to real-time vehicle detection.

Vehicle detection is challenging due to the changes of road conditions, lighting, positions and viewpoints. Many classical object detectors have been developed for car detection. They use feature extraction from image, such as HOG, combined with a classifier, such as Support Vector Machine(SVM) or Boosting, to train a model to detect the object. Deformable Part-based Model (DPM) have also been proposed to handle complex object variations.

Among recent works using ConvNets, Region Convolutional Neural Networks(R-CNN) attract great attentions in the field of computer vision. It combines selective search, a method of proposal generator, CNN feature extractor, SVM classifier, and bounding box regressors to provide an end-to-end trainable framework for object detection. R-CNN utilizes ConvNets, which is pre-trained by a large-scale image dataset such as ImageNet or PLACE, to extract feature from region proposals and achieves an outstanding performance of objection detection evaluated on PASCAL VOC dataset.

Inspired by the R-CNN framework, we formulate a technique specifically for car detection by making training pipeline more unified and reducing the complexity of R-CNN framework. Firstly, we focus on two classes only - car or non-car, so we remove the SVM classifier for each class in R-CNN and substitute it with a softmax classifier by CNN, which produce an even better end-to-end trainable network than R-CNN. To compensate the possible performance degradation caused by the removal of SVMs, we carefully fine-tune the CNN using the KITTI car detection dataset. Secondly, to reduce the complexity of running time for R-CNN model, we reduce the net structure of fully connected (FC) layers since many weights in FC layers are redundant.

The rest of this paper is organized as follows. In Section 2, we review some related works. In Section 3, we present the details of our work followed by the experimental result in Section 4. We conclude our work and future work in Section

5.

2.1 Prior work on Object Detection

The following will present a brief review about some related works in object detection and vehicles detection.

In this section, we will present a brief review of some related works in object detection and vehicles detection. In recent years, object detection has achieved many successes in computer vision field, from raw images (e.g. image pixels) to hand-crafted features, such as HOG or SIFT (e.g. [51], [42], [55]), for the improvement of detection accuracy. These features are often combined with SVM or Boosting [44] algorithms that are widely used in applications such as pedestrian detection.

There are also several works about deformable part-based model (DPM) by Felzenszwalb et al. [13]) and their derivatives [14] for object detection. The approach was the winner of PASCAL Object Detection Challenge in 2012 and the object detector achieved excellent results on the PASCAL and INRIA people detection datasets. DPM extends HOG feature and combines it with a discriminative model together. Their approach uses ensemble SVM [35] as well as latent SVM based on HOG features from a limited dataset. But their approach needs an exhaustive search for all possible locations and different scales to detect objects within an image. Another limitation is of using weak features usually HOG [9]).

With the appearance of large scale labeled datasets, e.g. ImageNet [10], and the surge of powerful computing machines, Krizhevsky et al. [26] won the Large-Scale Visual Recognition Challenge (ILSVRC-2012) and their CNN-based approach has been widely used in object classification. Most recently, OverFeat [45] uses a CNN as a regressor to localize objects in a coarse sliding-window detection framework.

In ILSVRC 2013, Ross proposed a R-CNN framework for ImageNet detection and obtained 31.4% mAP , which is an increment of 7.1% comparing to OverFeat [45] at 24.3% mAP. Their method utilizes transferrable feature from pre-trained CNN to a unknown dataset. When combined with a fast object proposals selection method, it achieves state-of-art results on PASCAL VOC dataset. Our work is inspired by R-CNN but applies specifically for vehicle detection using KITTI dataset. For vehicle detection, Ohn-Bar [40] proposed a method using AdaBoost, which employed pixel lookup features for fast detection and showed some promising results.

Chapter 3

Convolutional Neural Network for Vehicles Detection

In this section, we will talk about detailed method that employed in our vehicle detection framework. We propose a method for object detection using deep convolutional networks, especially on car detection. Convolutional neural networks (CNN) have demonstrated great success in various computer vision tasks. Our approach is simple, by transferring the rich feature hierarchies of CNN learned by large scale image dataset to specific task - car detection. We also evaluate the performance of the car detection algorithm using KITTI dataset. We improve runtime performance through algorithmic and implementation optimizations and are able to achieve near-realtime frame rates of over 10 FPS for high resolution images. Our proposed method can achieve the detection mean average precision of 66% on the KITTI car detection dataset.

3.0.1 Proposals generation

Some recent works on object proposals attempt to improve the detection accuracy by reducing the number of candidates and possible regions, which may increase the recall of detection result. These works includes objectness [1], selective search [49], BING [6], Edge Boxes [57], CPMC, Geodesic object proposals(GOP [25], MCG [2] and most recently LPO [24]. In our framework, we choose Edge Boxes as the proposal generator. The main idea behind the Edge Boxes is based on the edge map of an image. Compare to selective search used in original R-CNN pipeline, Edge Boxes generate 2k object proposals within 0.2 seconds, while selective search need more than one second. We also compare the results which proposals generated by MCG shows in figure 2.

3.0.2 Early rejection of proposals

To reduce the computational time on CNN forward passing, which is only needed in predicting process, we design a new model for objectness also based on CNN. In order to decide whether the proposals is an object or not, the objectness CNN is a simple binary classifier. We use the edge box results as training data: proposals with 70% overlap on ground truth are set as positive sample and the others are set as negative samples. We finetune a four convolutional layer model, initialized with Alexnet pretrain model. During test, we simply pass all the proposals generated by Edgebox and re-rank them using softmax score from objectness CNN model. Our experiments show that using only the top 50 proposals from our objectness CNN model can achieve comparable results from using all 2000 proposals from Edgebox. In addition, we find this early rejection module can speed up our detector considerably by passing at most 50 samples to VGGNet or GooogleNet, resulted in much less computational time than R-CNN approach.

3.0.3 Network architectures

Starting with LeNet-5 [27], CNN has form a standard structure with stacked convolutional layers, each optionally followed by local response normalization and max pooling layer, and also several fully-connected layers at the output. The deep ConvNets design attracts many researchers to work on it. Variants of the LeNet-5 model become prevalent in image classification and they have achieved the unprecedented results to some classical datesets, such as MNIST, CIFAR-10, CIFAR-100, and most notably the ImageNet. In the past several years, many famous network structures have been proposed for image classification, such as AlexNet [26], GoogLeNet [48], VGGNet [46], and so on. Some trends can be observed from the evolution of AlexNet to VGGNet: smaller convolutional kernel size, smaller convolutional strides, and deeper network architectures. These are effective techniques that can improve the performance for object recognition.

GoogLeNet. It is essentially a deep convolutional network architecture named Inception, whose basic idea is Hebbian principle and the intuition of multi-scale processing. An important component of the Inception network is the Inception module. Inception module is composed of multiple convolutional filters with different sizes alongside each other. In order to speed up the computational efficiency, 1×1 convolutional operation is chosen for dimension reduction. GoogLeNet is a 22-layer network consisting of Inception modules stacked upon each other, with occasional max-pooling layers with stride 2 to halve the resolution of grid. More details can be found in its original paper[48].

VGGNet. It is a new convolutional architecture with smaller convolutional size (3×3), smaller convolutional stride (1×1) , smaller pooling window (2×2), and deeper structure (up to 19 layers). The VGGNet systematically investigates the influence of network depth on the recognition performance, by building and pre-training deeper architectures based on the shallower ones. Two successful network structures are proposed for the ImageNet challenge: VGG-16 (13 convolutional layers and 3 fully connected layers) and VGG-19 (16 convolutionallayers and 3 fully-connected layers). More details can be found in its original paper [46].

DeepCar. We name our proposed model as DeepCar for vehicle detection using deep ConvNets. Due to the high accuracy and rich feature on VGGNet, we choose VGGNet with 16 layers as the basis. Because an average forward pass of VGGNet need 2298.68 ms and each image has 2k proposals generated by EdgeBox, we have to modify the VGGNet to reduce the forward time. Because more depth layers often results higher accuracy of a CNN model, we opt not to reduce the convolution layers in VGGNet. The computational time of forward and back propagation is largely depended on the fully connected layers, so we decided to prune 3/4 of weights in the fully connected layers (FC-6, FC-7).

3.0.4 Data augmentation

In the R-CNN model, they choose the bounding box with IoU (intersection over union) larger than 0.5 as positive data and others as negative (or background) data. For SVMs training process, they use the ground truth as positive data to improve localization precision and IoU less than 0.3 as negative data. In our work, all the training data are extracted from the raw images. Our data augmentation schemes can be described as follows. For the detection problem, the number of background proposals is typically much larger than that of the positive proposals during the test time. We generate 2k proposals for each images and calculate the IoU with ground truth. We set two different choices during the model training process. First, if the IoU with ground truth less than 0.5, we set these proposals as negative data and the rest as positive data. Secondly, the IoU with ground truth higher than 0.7 is set as positive data and less than 0.7 are set as negative data. The last step is to shuffle all data. By designing the training data in this way, we can achieve a precise localization without class specific SVMs.

3.0.5 Feature learning

For feature learning, we can finetune the pre-trained CNN models using the generated training data from KITTI dataset. We finetune the four models described above as our pre-trained models. These four models are all stacked with several convolutional layers, also pre-trained by the ILSVRC2012 ImageNet dataset. The last layer (FC-8) has two output: 0 for vehicle class and 1 for background. We will describe the detailed experiments and results in the next section.

3.0.6 Object detection

Next step is to detect objects using trained CNN model. Firstly, we generate some region proposals by Edge Boxes and then resize each proposal to the input blob size of each CNN model (for AlexNet is 227 and for VGGNet and GoogleNet is 224).

Then, we forward each proposal to CNN model and obtain the softmax output for each proposals. Each element of softmax output represents the probability of corresponding class (background or vehicle).

Subsequently, we employ the non-maximum suppression (NMS) algorithm to ignore the redundant proposals. The idea is to sort the proposals by their confidence scores and then ignore the proposals overlapping with a higher-scored proposal. The overlapping threshold is typically defined as the IoU between two proposals. Note that the IoU threshold will affect the performance of our detector, which should be tuned carefully to achieve the best performance.

3.0.7 Drop the SVM

As discussed in Section 2.1, R-CNN involves training an SVM classfier for each target object class as well as finetuning the CNN features for all classes. An obvious question is whether SVM training is redundant and can be eliminated. The finetuning process learned the last fully connected layers for the softmax predictor on top of CNN, whereas SVM training learns a linear predictor SVM the same features. In the first case, The softmax score Ps is an estimate of the class posterior for each proposals, in the second case Pc is a score that discriminates class c from any other class. In our case background is treated as one of the classes. As verified by our experiments in Section 4, Ps works poorly as a score for an object detector.However, and somewhat surprisingly, using Pn = Ps/P0 will lead to better performance nearly as good as using an SVM as score, where the P0 is the probability of the background class.

3.0.8 Bounding box refinement

We apply non-maximal suppression (NMS), which is a popular post-processing method for eliminating redundant object detection windows, for selected bounding boxes passed through by CNN before being evaluated. Non-maximum suppression can eliminate duplicated bounding box regions with higher softmax score. Starting from the highest ranked region in an image, other regions are iteratively removed if they overlap more than 0.3 with any of the currently retained regions so far.

3.1 From CNN to LSTM

Recurrent Neural Networks(RNN) have been used for many vision tasks for decades. Recently, RNN are explosive to be used in natural language processing(NLP), speech recognition and machine translation. A significant

To produce intermediate representations, we use expressive image features from GoogLeNet that are further fine-tuned as part of our system. Our architecture can thus be seen as a "decoding" process that converts an intermediate representation of an image into a set of predicted objects. The LSTM can be seen as a "controller" that propagates information between decoding steps and controls the location of the next output . Importantly, our trainable end-to-end system allows joint tuning of all components via back-propagation.

3.1.1 RNN background

Recurrent Neural Networks can be used for modeling complex temporal dynamics information by mapping input sequences to a sequence of hidden states. Although

RNN has been successfully used in speech recognition and natural language processing, but not many works try to resolve the non-sequence problems, such as object detection and object recognition. Due in part to the vanishing and exploding gradients problem that can result from propagating the gradients down through the many layers of the recurrent network, each corresponding to a particular timestep. [?] propose a CNN+LSTM framework for people detection in crowded scene to form a end-to-end trainable system. They use CNNs to get the rich feature then use RNN with LSTM units to decode image content into a coherent real-valued output of variable length. Inspired by their work, we implement LSTM for car detection with CNN feature.

3.2 Experiments

Dataset description. The KITTI object detection benchmark consists of 7481 training images and 7518 test images, comprising a total of 80256 labeled objects and 9 classes: ”Car”, ”Van”, ”Truck”, ”Pedestrian”,”Person sitting”, ”Cyclist”, ”Tram”, ”Misc” or ”DontCare”. All images are color. We split training dataset for training and validation, which split as 5500 and 1981 images. Also we select images with object ”Car”, ”Van”, and ”Trunk” for our experiment. All the result we describe below use the training image to train and most result are based on validation set. The detailed results will discuss in next section.

We use the Caffe [23] and cuDNN [7] libraries for our convolutional network implementation. All experiments were performed on one workstation equipped with a hex-core Intel 4790K CPU with 32 GB of memory. A single NVIDIA Tesla K40 with 12 GiB of memory was used for GPU computations.

Ensemble of Multiple CNNs: Several successful deep CNN architectures have been designed for the task of object recognition at the ImageNet Large Scale Visual Recognition Challenge. These architectures can be roughly classified into two categories: (i) deep CNN including AlexNet and Clarifai, (ii) very-deep CNN including GoogLeNet and VGGNet. We exploit these very-deep networks in our proposed Object-Scene CNN architecture and aim to verify the superior performance of deeper structure.

3.3 From CNN to LSTM

Recurrent Neural Networks(RNN) have been used for many vision tasks for decades. Recently, RNN are explosive to be used in natural language processing(NLP), speech recognition and machine translation. A significant limitation of simple RNN models which strictly integrate state information over time is known as the "vanishing gradient" effect: the ability to back-propogate an error signal through a long-range temporal interval becomes increasingly impossible in practice. To resolve "vanishing gradient" issue, recently many works proposed a LSTM unit and show improvement for performance.

To produce intermediate representations, we use expressive image features from GoogLeNet that are further fine-tuned as part of our system. Our architecture can thus be seen as a "decoding" process that converts an intermediate representation of an image into a set of predicted objects. The LSTM can be seen as a "controller" that propagates information between decoding steps and controls the location of the next output . Importantly, our trainable end-to-end system allows joint tuning of all components via back-propagation.

3.3.1 RNN background

Recurrent Neural Networks can be used for modeling complex temporal dynamics information by mapping input sequences to a sequence of hidden states, and hidden states to outputs via the following recurrence equations:

Although RNN has been successfully used in speech recognition and natural language processing, but not many works try to resolve the non-sequence problems, such as object detection and object recognition. Due in part to the vanishing and exploding gradients prob- lem [12] that can result from propagating the gradients down through the many layers of the recurrent network, each corresponding to a particular timestep. [**?**] propose a CNN+LSTM framework for people detection in crowded scene to form a end-to-end trainable system. They use CNNs to get the rich feature then use RNN with LSTM units to decode image content into a coherent real-valued output of variable length. Inspired by their work, we implement LSTM for car detection with CNN feature.

3.3.2 Best practice of finetune

In our work, we have finetune full VGGNet, reduced VGGNet, GoogleNet and AlexNet on KITTI car detection dataset. First, we try to reproduce R-CNN result, but apply our model achitecture on on PASCAL 2007 dataset [11] which has 20 object class and one background class.

We use the almost the same protocal with R-CNN provided solver, but we reduce the iteration for just 40k of finetune the AlexNet. Also we used ”xaviar” initialization for new layers(FC8 layer). Furthermore, we use ”poly” learning rate policy instead of ”step” policy as it is proved to converge faster than ”step”. The result presented in section 5.

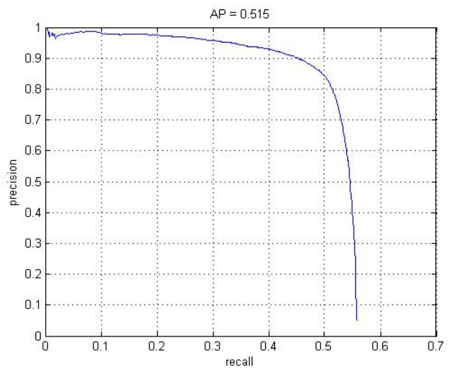

Figure 3.1: mAP on the KITTI car detection data as proposals generted by EdgeBox and model by GoogleNet

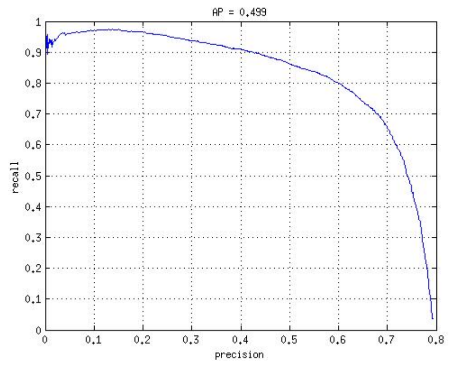

Figure 3.2: mAP on the KITTI car detection data as proposals generted by MCG and model by GoogleNet

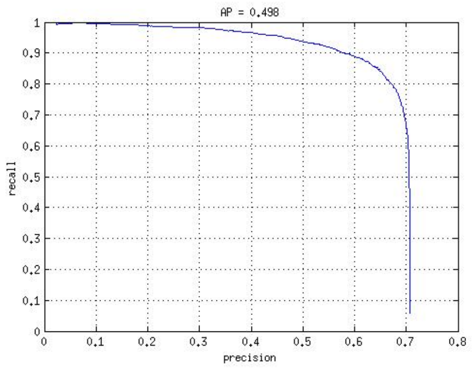

Figure 3.3: mAP on the KITTI car detection data as proposals generted by EdgeBox and model by Reduce VGGNet

VOC2007	IoU	mAP
	0.1	0.3589
	0.2	0.3665
	0.3	0.3712
	0.4	**0.3722**
	0.5	0.3680
	ensemble	**0.3738**

Table 3.1: Finetune AlexNet on PASCAL VOC2007.

KITTI on VGGNet. We then finetune VGGNet on KITTI dataset. To explore the finetuning process, we try several methods for updating weights. We use the very deep VGG16 model to validate the convolutional layers are important for detection. We first make first 13 layers remain fixed at their initialization, and only the 3-layer fully connected layers will be updated, and achieve the mAP as 48.52%. Then we release the all conv5_1 to conv5_3 convolutional layers, and set 10 times learning rate than previous layers, which make mAP increase to 49.80%. We continue to explore the effect on convolutional layers for detection, to release conv4_1 to conv4_3 layers, we get 50.32% mAP.

Faster detection on VGGNet. For VGGNet-16 model, one forward pass time average need 2000 ms or more, for detection task the large amount proposals need to forward several thousands time for fully connected layers. We use offers a simple way to compress fully connected layers. By sub-sampling the weight of fc6, fc7 and fc8 weights, we compression the 4096 to 1024 by average sampling. On the other size, the target of our task is specifically for two class (car and non-car), the original fully connected layer weights is heavily redundant for our scenarios. This simple compression method gives good speedups for detection without the need for additional fine-tuning.

IoU	mAP
from fc	48.52%
from conv5	49.80%
from conv4	**50.32%**

Table 3.2: Finetune VGGNet16 on KITTI.

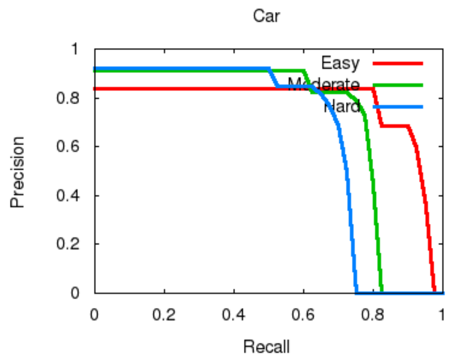

Figure 3.4: mAP on the KITTI Benchmark by CNN + LSTM

KITTI Benchmark	**Easy**	**Moderate**	**Hard**
Car (Detection)	83.70 %	82.36 %	68.35 %

KITTI on GoogleNet. Finally, we apply the very deep GoogleNet model on the KITTI dataset, in order to balance the performance and speed, we just finutune the the 3-layer classifier, which means the learning rates for other layers remain fixed at their initialization. Compared to VGGNet, we can obtain a better mAP as 51.50%. Also the speed for GoogleNet improved a lot than VGGNet.

Compare of proposals. R-CNN starts by running an algorithm such as Selective Search(SS) to extracts from an image x a shortlist of of image regions R that are likely to contain objects. To reduce the complexity of the R-CNN, we adopt edge box instead of selective search used in R-CNN. The figure shows that even though the mean average precision (mAP) between edge boxes and selective search are almost the same, edge boxes runs much faster than selective search.

EdgeBox[57] is a scoring function is evaluated in a sliding window fashion. This method uses object boundaries estimates (obtained via structured decision forests) as feature for the scoring. Interestingly, the authors propose tuning parameters to optimize recall at a desired overlap threshold.

MCGMC[2] is one of the most recent methods combining gPbUCM and CPMC. The authors propose an improved multi-scale hierarchical segmentation, a new strategy to generate proposals by merging up to 4 segments, and a new ranking procedure to select the final detection proposals.

For these proposals, in the order of a few thousands per image, may have arbitrary shapes, but in the following are assumed to be converted to rectangles. For fast detection, we select EgdeBox as proposals generator as the header of the detection pipeline. Also we measure the proposal generator MCG to evaluate the performance. We find that the increase number of proposals would not increase the performance for detection result.

We can draw several interesting conclusions. First, for the same low number of candidate boxes, EdgeBox is much better than any fixed proposal set; less expected is that performance does not increase even with 3 times more candidates, indicating that the CNN is unable to tell which bounding boxes wrap objects better even when tight boxes are contained in the shortlist of proposals. This can be explained by the high degree of geometric invariance in the CNN.

speed of detection time. In the testing procedure, we first generate proposals for each test image by edge boxes. The average run time for each image(1242×375)is around 0.15 seconds. However, for each image the number of proposals ranges from 2000 up to 6000. We use the Caffe framework, the forward pass for each batch(batch size is 128) takes roughly 500 ms per batch and around 5 to 15 seconds per image. Based on the analysis above, we try to explore the real time application, and try to diminish the number of proposals to 200 per image heuristically. and found directly performs really poorly, with a drop of about 10% mAP point.

3.3.3 Discussion

Comparison with state-of-the-art: To evaluate the performance of the detection, we use the mean average precision (mAP). The mAP equals to the integral over the precision-recall curve.To determine the precision-recall curve, we need to compute the true positive and false positive value of our prediction first. We use the IoU (intersection of union) to determine the successful detection.Then we designate a threshold for IoU, for example 0.5, if the IoU exceeds the threshold, the detection marked as correct detection. Multiple detections of the same object are considered as one correct detection and with others as false detections. After we get the true positive and false positive values, we can compute the precise-recall curve, and then evaluate the mAP.

3.3.4 Conclusion

In this work we presented CarNet, a simple end to end trainable convolutional neural network architecture that works good in object detection. As part of developing and analyzing this approach we provided analysis of many architectural choices for the network, discussing best practices for training, and demonstrated the importance of finetuning, proposal generation and how deep the model effect the detection performance.

Our most significant finding is that current CNNs do contain sufficient spatial information for accurate object detection, although in the convolutional rather than fully connected layers. This finding opens the possibility of building state-of-the-art object detectors that rely exclusively on CNNs, removing region proposal generation schemes such as EdgaBox, and resulting in integrated, simpler, and faster detectors.

Our current implementation of a proposal-free detector is already much faster than R-CNN, and very close, but not quite as good, in term of mAP. However, we have only begun exploring the design possibilities and we believe that it is a matter of time before the gap closes entirely. In particular, our current scheme is likely to miss small objects in the image. Although theoretically, features from higher level layers of a network have very large receptive fields, in practice the size of receptive fields at higher levels is much smaller.

Given the simplicity and easy of training, we find these results very encouraging. In our ongoing work, we are exploring combining our technique with proposal-free framework, such as done in [17].

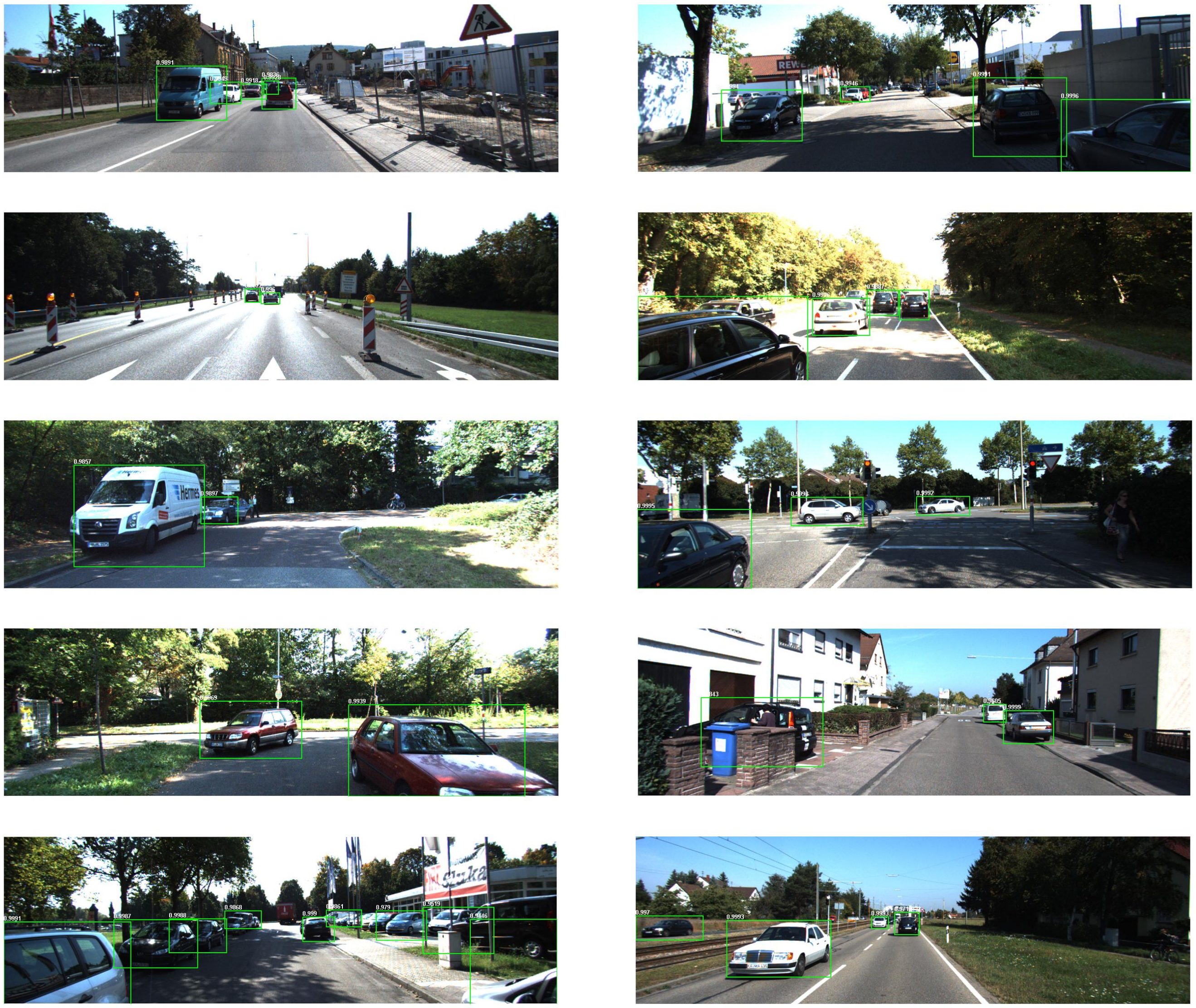

Figure 3.5: Result using DeepCar model

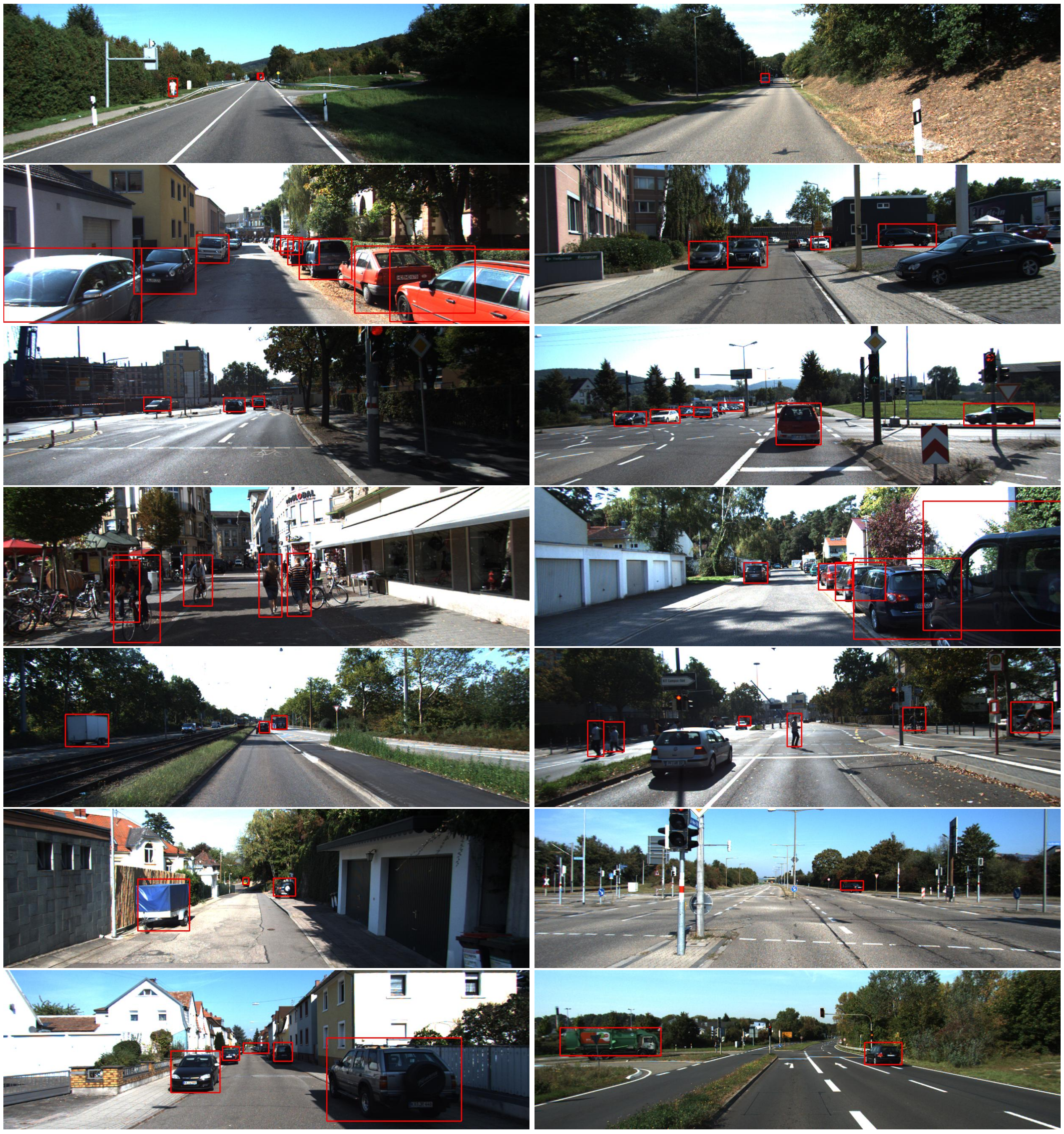

Figure 3.6: Result using LSTM model

Chapter 4

Road Marking Detection and Classification

This chapter presents a novel approach for road marking detection and classification based on machine learning algorithms. Road marking recognition is an important feature of an intelligent transportation system (ITS). Previous works are mostly developed using image processing and decisions are often made using empirical functions, which makes it difficult to be generalized. Hereby, we propose a general framework for object detection and classification, aimed at video-based intelligent transportation applications. It is a two-step approach. The detection is carried out using binarized normed gradient (BING) method. PCA network (PCANet) is employed for object classification. Both BING and PCANet are among the latest algorithms in the area of machine learning. Practically the proposed method is applied to a road marking dataset with 1,443 road images. We randomly choose 60% images for training and use the remaining 40% images for testing. Upon training, the system can detect 9 classes of road markings with an accuracy better than 96.8%. The proposed approach is readily applicable to other ITS applications.

4.1 Introduction

Object detection and classification have attracted considerable interests from researchers in recent decades. Various databases are built to evaluate the latest object detection and classification algorithms, such as the Caltech101 [12] and Caltech256 [20], Pascal visual object classes dataset [VOC], ETHZ shape classes [15], face detection dataset, and etc. These datasets have been broadly used as benchmarks for new algorithm development and performance comparison.

In recent year, the approach of machine learning has become increasingly popular to explore the structures or algorithms that a system can be programmed to learn from data or experience. It has been widely used in computer vision, search engines, gaming, computational finance, robotics and many other fields. Since Hinton et. al proposed an effective method to train the deep belief networks[21] in 2006, deep learning networks have gained lots of attentions in the research community. Deep learning networks are able to discover multiple levels of representations of a target object. Therefore, they are particularly powerful for the tasks of pattern recognition. For instance, the convolution neural network (CNN) has demonstrated superior performance on many benchmarks [CNN1, CNN2], although CNN requires significant computations. PCA network (PCANet) [3] is a type of deep learning networks that has been introduced recently. When compared to CNN, the structure of PCANet is much simpler, but it has been demonstrated as an effective method for image classification [3]. The PCANet architecture mainly consists of the following components: patch-mean removal, PCA filter convolutions, binary quantization and mapping, block-wise histograms, and an output classifier. More details about the PCANet algorithm will be discussed in Section [sec:Proposed-Method].

Advanced Driver Assistance System (ADAS) has become a main stream technology in the auto-industry. Autonomous vehicles, such as Google's self-driving cars, are evolving and becoming reality. A key component is video-based machine intelligent that can provide information to the system or the driver to maneuver a vehicle properly based on the surrounding and road conditions. There have been lots of research works reported in traffic sign recognition [36], [56], lane departure warning [lane], pedestrian detection [8], and etc. Most of these video-based object detection methods are developed using the classic image processing and feature extraction algorithms. For different types of objects, certain features usually works better than others as reported in the literature. Often, object detection is followed by a classification algorithm in these intelligent transportation applications. Typical classifiers, such as Support Vector Machine (SVM), artificial neural network, and boosting, are applied to identify one or multiple classes of the detected objects.

4.2 Related work

Road marking detection is an important topic in Intelligent Transportation System (ITS) and has been researched extensively. As described in [37], many previous works were developed based on various image processing techniques such as edge detection, color segmentation and template matching. Road marking detection can also be integrated as part of a lane estimation and tracking system [50]. The lane borders and arrow markings were detected using scan-lines and template matching methods. The information of the lane types, i.e. forward, left-turn, and right-turn, were sent to the console or the driver. In [31], it presented a method of lane detection. Lines were extracted from the original image through edge detection, following by some rule-based filtering to obtain the candidates of lanes. Additional properties such as brightness and length of the lines were examed to detect the lanes. [16] was able to detect and recognize lanes, crosswalks, arrows and many other markings on the road. The road marking on an image were extracted first using a modified median local threshold method. The road displayed on the image was a trapezoidal area due to the effect of camera angle and 3D space projection. Thus, road markings on the image also had distortions and variations in shape and size. Then perspective transform was applied to convert the trapezoidal road area into a rectangular area, which reduced the distortions and variations of the road marking, making it easier for detection. Similarly, perspective transformation was also applied in [32]. The lanes were detected using Augmented Transition Network (ATN). Subsequently, the detected lanes were used to locate the Region of Interests (ROIs) on an image for detecting other road marking such as arrows. In [53], the Maximally Stable Extremal Regions (MSERs) was employed as an effective way of detecting region of interest. Both Histograms of Oriented Gradients (HOG) [8] features and template matching methods were used for classification.

4.3 proposed method

We propose a system that is capable of detecting and recognizing different road markings. We use BING feature to find and locate the potential objects on a road image, i.e. road markings. The potential objects are then classified by a PCANet [chan2014pcanet] classifier to obtain the final results. Unlike the traditional approach of tuning image processing techniques geared specifically for road marking detection, our system is an extendable framework that can be adopted to other detection and classification tasks.

4.4 BING feature for detection

The BING feature is employed to find the potential objects in an image. It is the binary approximation of the 64D norm of the gradients (NG) feature. Each image window is resized to 8×8 pixels for computational convenience, and its norm of the gradients forms the 64D NG feature. It represents the contour of the target object in a very abstracted view with little variation. Thus, the BING features can be used to find objects in an image. It is very efficient in computations compared to some existing featureextract algorithms. The BING feature is suitable for finding road markings, because the road markings have closed boundaries and high gradients around the edges.

In order to locate the target objects in an image using the BING feature, we need to train it with training samples. The positive samples are true objects manually labeled in images and the negative samples are the background in images. The machine learning method inside BING is actually linear SVM. It is observed that some window sizes (e.g. 100×100 pixels) are more likely to contain objects than other sizes (e.g. 10×500 pixels). Therefore an optional fine-tune step trains another SVM, taking window size into consideration. These two SVMs form a cascaded predictor with better accurate.

Although BING is an efficient way of finding objects in an image, it has certain limitations. First, because the 64D NG feature or BING feature represents the object in a very abstracted view, the trained detector does not filter some background very well. In other words, some background may have similar BING feature as the true objects, and they may still be selected as potential objects. Secondly, as a bounding box based detection algorithm, it has the common problem that a bounding box may not accurately locate the true object. Such inaccuracy may cause failure in the subsequent recognition stage. However, these limitations can be alleviated or overcome. As a fast object detection method, we manually assign an arbitrary number that represents the number of potential objects to be selected by the detector, according the their confidence values. This number is often much large than the number of true objects in an image. For example, BING may provide 30 potential objects from an image, while there are only one or two true objects in it. Therefore, the true objects are unlikely to be missed, but adversely many false objects might also be included in the candidates pool. We deal with the false candidates in the classification step using PCANet. For the problem of inaccurate bounding box locations, we have collected a large number of true objects at various bouding box locations by multiple runs of BING detection. Therefore, the true objects can still be recognized even if the bounding box locations are not precise. Fig. 2 shows an example that BING produces 30 candidates through object detection.

4.5 PCANet for Detection

Taking the detection results from the BING stage, we build a PCANet classifier to filter out the false candidates and to recognize the true road markings. The PCANet classifier consists of a PCANet and a multi-class SVM. The structure of PCANet is simple, which includes a number of PCA stages followed by an output stage. The number of PCA stages can be varied. A typical PCANet has two stages. According to [chan2014pcanet], the two-stage PCANet outperforms the single stage PCANet in most cases, but increasing the number of stages does not always improve the classification performance significantly, depending on the applications. In this work, we choose two-stage PCANet.

To a certain extend, the structure of PCANet is to emulate a traditional convolutional neural network [Hinton2012]. The convolution filter bank is chosen to be PCA filters. The non-linear layer is the binary hashing (quantization). The pooling layer is the block-wise histogram of the decimal values of the binary vectors. There are two parts in the PCA stage: patch mean removal and PCA filters for convolution. For each pixel of the input image, we have a patch of pixels whose size is the same as the filter size. We then remove the mean from each patch, followed by convolutions with PCA filters. The PCA filters are obtained by unsupervised learning during the pre-training process. The number of PCA filters can be variant. The impact of the number of PCA filters is discussed in [chan2014pcanet]. Generally speaking, more PCA filters would result better performance. In this paper, we choose the number of filters equals to 8 for both PCA stages. We find that it is sufficient to deliver desirable performance. The PCA stages can be repeated multiple times as mentioned above, and here we choose to repeat it only once.

The output stage consists of binary hashing and block-wise histogram. The output of PCA stages are converted to binary values by a step function, which converts positive values to 1 and else to 0. Thus, we obtain a binary vector for each patch and the length of this vector is fixed. We then convert this binary vector to decimal value through binary hashing. The block-wise histogram of these decimal values forms the final output features. We then feed the SVM with the features from PCANet. Fig 3. shows the structure of a two-stage PCANet. The number of filters in stage 1 is m and in stage 2 is n. The input images are object candidates from BING.

4.6 Result

In our experiments, we evaluate the proposed system using the road marking dataset provided by [53]. The dataset contains 1,443 road images, each with size of 800×600 pixels. There are 11 classes road marking in these images. In this paper, we evaluate 9 of them because the data of the other 2 classes are insufficient for machine learning. We train the object detection model by manually labeling the true road markings in the images. The PCANet model is trained iteratively to ensure its accuracy. The initial training samples are manually labeled from a small portion of the dataset, and the trained model along with the object detection model is applied to the whole dataset to detect road markings. The results are examined and corrected by human interference in order to ensure the correctness of the data during the next training iteration. Through the iterative procedure, one road marking on an image can be detected multiple times and generates multiple training samples. Because of the utilization of the BING feature and its object detection model, the true samples may be extracted using various bounding boxes, making the PCANet classifier more robust.

We measure the performance of our PCANet classifier by using 60% images for training and 40% images for test. The 1,443 images are re-ordered randomly and thus the training and test images are selected randomly without overlap. The window-sized training samples and test samples are from the training images and test images respectively. We perform data augmentation over the collected samples by transforming the original images with parameters such as roll, pitch, yaw, blur and noise. Table [Tab:result] shows the evaluation results of the PCANet classifier, which is referred as the confusion matrix. The test samples for each class is 250. The cell at the ith row and the jth column gives the percentage that the ith samples are recognized as the jth samples. The “OTHERS” class represents negative samples without road marking. Comparing to the previous results in [Wu2012], our classification accuracy is more consistent and significantly better especially for the “FORWARD” sign.

4.7 Conclusion

In this chapter, we present a framework for object detection and classification using the latest machine learning algorithms including BING and PCANet. BING can quickly identify the target classes of objects after the system is trained with a set of images with target objects. Subsequently, these detected objects are classified by the PCANet classifier. Similarly, the classifier is also pre-trained using the dataset and is capable of identifying many types of objects simultaneously. As an example, we demonstrate this approach by building a system that can detect and identify 9 classes of road marking at very high accuracy. More importantly, the proposed approach can be employed for many other video-based ITS applications provided that sufficient training datasets are available.

Chapter 5

End-to-end Convolutional Network for Weather Recognition with Deep Supervision

We propose a novel weather recognition algorithm based on pixel-wise semantic information. The proposed end-to-end convolutional network model combines a segmentation model with a classification model. The segmentation model is inspired by the fully convolutional net-works (FCN) [33] and is able to produce intermediate pixel-wise semantic segmentation maps. Next, an ensemble of color image and semantic segmentation maps feed to the next classification model to designate the weather category. Since the proposed model is complex, it makes training more difficult and computationally expensive. In order to train deeper networks, we transfer the early supervision idea from deeply-supervised nets [28] into our segmentation task by adding auxiliary supervision branches in certain intermediate layers during training. The experiments demonstrate that the proposed novel segmentation model makes the training much easier and also produces competitive result with the

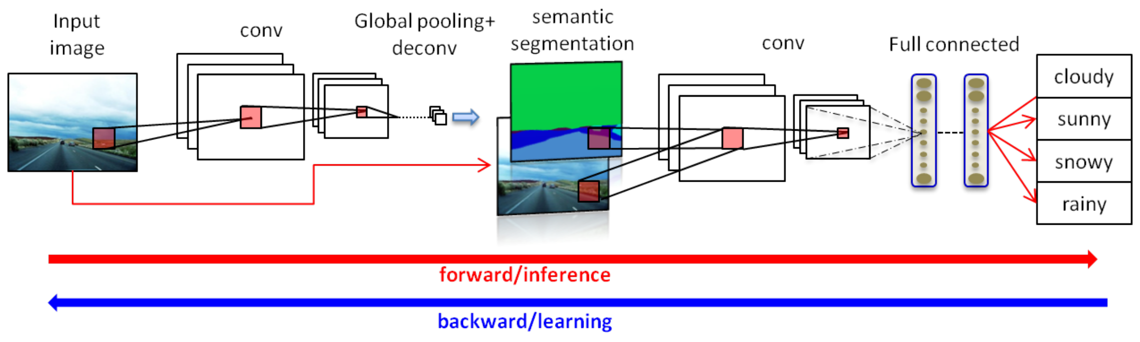

Figure 5.1: The proposed end-to-end convolutional network model combines a semantic segmentation model with a weather classification model.

current state-of-the-art FCN on the PASCAL Context dataset. By employing our segmentation network for weather recognition on a end-to-end training classification framework with additional semantic information, we gain a significant improvement (i.e., from the state-of-the-art 91.1% to 95.2%) for the public weather dataset.

5.1 Introduction

Understanding weather conditions is crucial to our daily life. Weather conditions strongly influence many aspects of our daily lives from solar technologies, outdoor sporting events, to many machine application including the driver assistance systems (DAS), surveillance and real time graphic interaction. While most current existing weather recognition technologies rely on human observation or expensive sensors, they limit scalability of analyzing local weather conditions for multiple locations. Thanks to the cost of decreasing cameras, cameras have spread extensively everywhere in the world. Image-based weather recognition derived from computer vision techniques is a promising and low cost solution to automatically obtain weather condition information anywhere in the world.

Semantic information can successfully help provide effective cues for scene classification. Li et al. [29] proposed an object bank representation for scene recognition. The word "object" mentioned is a very general form where any number of data points can be classified as such, from cars and dogs, to sky and water. This representation carries high-level semantic information rather than low-level image feature information, making its result superior than other methods of high-level visual recognition processes. However, this approach also highly relies on the performance of object detection and the cost of scaling is very high to expand the object categories.

In this paper, we propose an end-to-end convolutional network to predicts the class of the weather on a given image (e.g., cloudy, sunny, snowy, and rainy). This model combines a segmentation model with a classification model shown in Figure 5.1. The former model conveys high-level semantic information to the latter model which gets better accuracy. During the training, the end-to-end learning framework automatically decides the most re-liable features of the specific category, e.g., the dusky sky corresponding to cloudy, but the non-uniform dusky color on roads might be the shadow corresponding to sunny.

The main contributions of this work can be summarized as follows.

1. To the best of our knowledge, this is the first paper to propose an end-to-end convolutional network model which combines a segmentation model with a classification model, allowing for high-level visual recognition tasks.

2. The proposed model effectively conveys semantic information to the enhance classification performance. Our results have a significant improvement over current state-of-the-art weather classification methods. Our approach achieves an accuracy of 94.2% instead of 91.1% from current practices [38].

3. The modified segmentation model with early supervision and global/feature fusion can show improvement over current state-of-the art methods that involve fully convolutional networks (FCN) [33].

The rest of this paper is organized as follows. In Section 2, we review related works. In Section 3, we present the details of our work followed by the experimental result in Section 4. We conclude our work and future works in Section 5.

5.2 Related work

Only a few methods have investigated image-based weather recognition using low-level features. These methods [43, 54, 5, 34] usually extract a set of hand crafted low-level features from Regions of Interest (ROIs) and then train the classifiers, e.g., Support Vector Machine (SVM)[43, 5], Adaboost[54] and k-nearest neighbor[47]. [43] extracts features using hue, saturation, sharpness, contrast and brightness histograms from the predefined global and sub region of interest (ROI). Based the extracted features, support vector machine is applied to classify the data into three classes, clear, light rain, and heavy rain. [54] focus the image captured in vehicle. Both histograms of gradient amplitude, HSV and gray value on the road area are extracted and classify the image into three classes (sunny, cloudy, and rainy). In addition to the static features, the dynamic motion features also applied in [5], extracts the color(HSV), shape, texture(LBP and gradient) and dynamic motion features from the sky region and classify it by way of the SVM classifier. These approaches may work well for some images with specific layouts but they fail for weather classification of images taken in the wild, i.e., it can not be expected to extract the features from the specific semantic regions, e.g., sky or road.

In order to better address these challenges, Cewu [34] et al. recently proposed a complex collaborative learning framework using multiple weather cues. Specifically, this method proposed a 621 dimensional feature vector formed by concatenating five mid-level components, namely: sky, shadow, reflection, contrast and haze which correspond to key weather cues. To extract these cues, this process involves many pre-processing techniques such as sky detection, shadow detection, haze detection and boundary detection. This makes this model highly relies on the performance of the aforementioned techniques.

Recently, deep convolutional neural network (CNN) have shown great potential to learn the discriminative features and the decision boundary of classification simultaneously. Some pre-trained convolutional neural networks [26, 18, 4] have shown the ability to possess rich and diverse features that have been learned from the large scale dataset, e.g., LSVRC-2012 ImageNet challenge dataset [10]. Elhoseiny et al [38] apply the finetuning procedure on the Krizhevskys CNN [26], which follows the same structure in the first seven layers while the output layer (8th layer) is replaced with two nodes, one for cloudy and one for sunny. This approach uses an extract holistic feature without any semantic information e.g., objects category and spatial location. However, semantic information can lead to good feature cues that contribute high-level visual recognition tasks [29]. As a result, we proposed a method to take advantage of the power of the CNN while also leveraging the classification result based on the semantic information.

5.3 Our Method

Our proposed convolutional network model combines a segmentation model with a classification model. To implement this, we first introduce our segmentation model and then the classification model after.

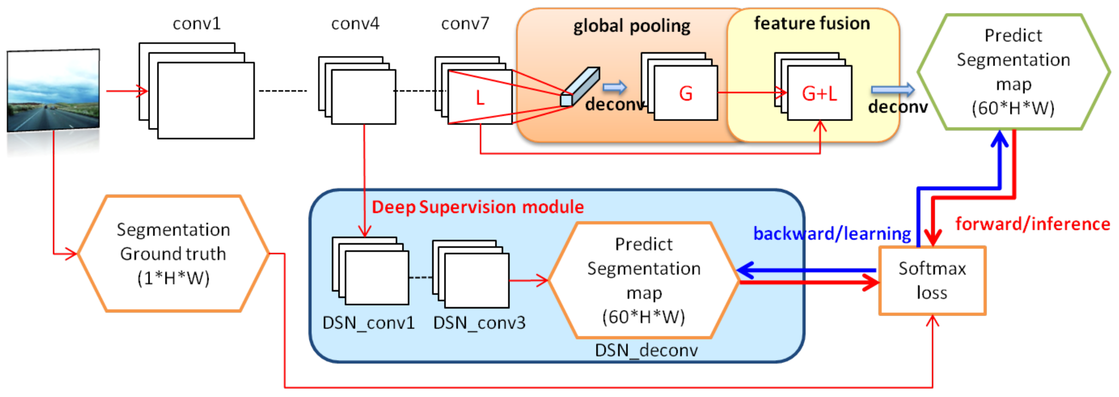

Figure 5.2: Illustration of our early supervision full convolutional network (ES-FCN) models. The network consists of a main branch and one additional early supervision loss branch, which allows the deeper network to be trained easily. Meanwhile, this network integrates the global pooling and multiple features fusion to generate reliable semantic segmentation results.

5.3.1 Segmentation Model: Early Supervision Full Convolutional Network (ES-FCN)

Pixel-wise semantic segmentation map can be considered the most informative semantic information which provides not only the category information but also the spatial layout of each category. Recently, many CNN segmentation methods [33, 30] have shown a promising result to extract the pixel-wise semantic segmentation map. Inspired by the novel architecture fully convolutional neural network (FCN)[33] which modifies the contemporary classification networks(AlexNet [26], the VGGNet [4], and GoogLeNet [48]) to allow the network to produce a segmentation result with correspondingly-sized input image. The format of this semantic segmentation map is suitable for being intermediate cues for advanced scene classification. As shown in Figure 5.1, we use the fully convolutional neural network to fulfill the segmentation task.

We perform the network surgery of the object classification contemporary classification networks(deeply-supervised nets(DSN) [28]) to maintain feature map as the image format via converting original full connected layers to the convolution layers.

Since the proposed model is complex, it makes training more difficult and computationally expensive. In order to train deeper networks, we transfer the early supervision idea from deeply-supervised nets(DSN) [28] into our segmentation task by adding auxiliary supervision branches in certain intermediate layers during training. Meanwhile, we adopt two additional procedures, "global pooling" and "feature fusion". "Global pooling", as shown the proposed network in Figure 5.2 smooths out some discontinuous segmentation results and the "feature fusion" enhances the discriminative power of feature by combining global pooling results with coarse feature maps from previous layers. These modification can produce more accurate and detailed segmentations shown in the experiment section.

5.3.2 Early Supervision Module

Since very deep neural network [46, 48, 28], has made great progress on large scale image dataet - ILSVRC ImageNet Contest [10], it is incredibly hard to train the model efficiently and effectively. VGG group suggests a 19 layer CNNs [48]. To train the model, they finetune the larger network based on the small initialized CNN till 19 layers. While they achieve very good performace on the ImageNet competition, the training process is very slow and time-consuming. Their approach also relies on experience for finetuning very deep models. Deepy-supervised nets (DSN [28]) integrated deep supervision in intermediate hidden layer. The optimized loss function combines intermediate hidden layer loss and final classification loss together to prevent gradient from vanishing.

We follow the rule in DSN [28] to add the supervision module in intermediate hidden layers. To decide where to put the deep supervision branch, we follow the rule from [52]. In their eight layers model, the gradient start vanish at fourth convolutional layers, so we decided to put deep auxiliary supervision module after the third convolutional layers with a max-pooling operation.

Contrary to DSN with simple fully connected layers in auxiliary supervision, our target is not the classification task, but the segmentation task. So we convert classifiers to dense fully convolutional layer for auxiliary supervision module and final output module. Firstly, we convert the fully connected layers into 1×1 kernel convolutional layers. For Pascal context segmentation task, there are 60 classes (59 classes + 1 background), so the last convolutional layer should have 60 output feature maps, also the output feature map size will be the same as the size of the ground truth label.

5.3.3 Global Feature Extraction: Global Pooling

For semantic segmentation, due to the per-pixel classifier or per-patch classifier in the top layer of CNN, the local information can lead to a final segmentation result. However, ignoring the global information of the image would easily generate segmentation results with small noise fragments. This problem has been solved with many different methods. ParseNet[30] uses global pooling to get global information and fuse with local information. FCN [33] fuses together different layers feature map to contribute to the final output segmentation result.

Considering the FCN model [33], the features from higher level layers have very large receptive fields (e.g. FC7 in FCN has a 404×404 pixels receptive field). However if the size of a receptive field at higher levels is much smaller, it will prevent the model from making global decisions. Thus, adding features from the global information of the whole image is needed and is rather straightforward for our ES-FCN framework.

To simplify our model structure, we apply a method similar with ParseNet.

Specifically, we use global average pooling after convolution seven layer and combine the context features from the last layer or any previous desired layer. The quality of semantic segmentation is greatly improved by adding the global feature to local feature map. Experiment results on PASCAL-Context [39] dataset also verifies our assumption. Compared with FCN, the improvement is similar to using CRF to post-process the output of FCN.

5.3.4 Feature Fusion

We get the extract the global information via global pooling to get M feature maps of size 1×1 then unpooling to the same size as high level feature. The global unpooling map concatenates with high level feature (previous layer in our setting) to M new fusion layers using element product, as shown in Table **??** where $M = 1024$. Because the features in different layers are in different scales, simple fusion of top layer feature with low level features will lead to poor performance. Thus, ParseNet apply L2-norm and learn the scale parameter for each channel before using the feature for classification, which leads to a more stable training. For our model structure, we replaced the L2-norm layers by a batch normalization layer [22] which shows a more reliable result.

5.3.5 Ensemble Semantic Segmentation Map for Classification

To fully utilize the segmentation result from our ES-FCN model, we proposed four types of fusion methods for segmentation results and raw images, which transfer the segmentation task to classification and make the whole network trainable end to end. The fusion methods are as follows: 1. raw RGB images concatenate with 60 channel segmentation results (63 channels in total); 2. Based on the segmentation map, we employ a convolutional layer with 1×1 kernel size, used for feature selection and use element-wise product with raw image. 3. generate a full segmentation map within one channel and concatenate with a 3 channel raw image.

5.4 Experiments

We use the Caffe [23] and cuDNN [7] libraries for our convolutional network implementation. All experiments are performed on one workstation equipped with an Intel E5-1620 CPU with 32 GB of memory and an NVIDIA TiTan X with 12 GB of memory for GPU computations.

Dataset We evaluated our algorithm on PASCAL Context dataset, which is extend PASCAL VOC 2010. This dataset is a set of additional annotations for PASCAL VOC 2010. It goes beyond the original PASCAL semantic segmentation task by providing annotations for the whole scene. The segmentation results for the 59 categories(and background class) Following the same training and validation split by FCN, we employed 4998 images for training, and 5105 images for validation. All the results are employed by the validation set. We also use Caffe and finetune our ES-FCN model. Without supervision model on ImageNet dataset public available now, we start a new training process for ImageNet (ILSVRC) dataset with 1.2 million images and 1k classes.

Evaluation metrics For segmentation task, all previous works used mean Intersection over Union(mIoU) to evaluate performance. We not only evaluate our model employed on mIoU and compare it with well-known results. We also use per pixel accuracy, per label accuracy, and weighted IoU accuracy to evaluate and compare models.

Train a Deep Supervision model DSN [28] reports results on the ILSVRC subset of ImageNet [10], which includes 1000 categories and is split into 1.2M training, 50K validation, and 100K testing images (the latter has held-out class labels). The classification performance is evaluated using top-1 and top-5 classification error. Top-5 error is the proportion of images such that the ground-truth class is not within the top five predicted categories.

In our work, we pretrain an ImageNet-DSN model first, which contains 8 convolutional layers and 3 fully connected layers, using the strategy: we use stochastic gradient descent with polynomial decay policy to train a network with five convolutional layers, and then we initialize the first five convolutional layers and the last three fully connected layers of the deeper network with the layers from the shallower network. The other intermediate layers are initialized by Xavier [19] initialization method, which works well in practice. Including the time for training the shallower network, ImageNet-DSN takes around 6 days with 80 epochs on two NVIDIA TiTan X GPUs with batch size 128. Then, we add deep supervision branch on ImageNet-DSN model using our method in section 5.3.1. This model is trained with auxiliary supervision that's added after the third convolutional layer as shown in Table **??**. This model takes around 3 days to train with 35 epochs on two TiTan X GPUs with batch size 128. The learning rate starts with 0.05 and weight decay as 1e-5 in all our ImageNet-DSN training.

Fully Convolutional Layer To fully exploit the rich feature in ImageNet-DSN pretrain model, we do net surgery for all the fully connected layers for supervision module and final classifiers, simply replace fully connected to convolutional layers with 1×1 kernel size. Also we remove the last classifiers(1000 outputs), then replace with a convolutional layers with 1×1 kernel size, but the output we set as 60(59 classes + 1 background). Following PaserNet, we remove the 100 padding in the first convolutional layer, which employed in FCN model. To carefully design the kernel size, we use 12×12 kernel size in fc-6 (convolutional layer).

5.4.1 Global Feature Fusion

For simplicity, we use features from pooling layer as the global context feature. We then apply the same model on PASCAL-Context by concatenating features from different layers of the network. By adding global context pool6, it instantly improves mean IoU by about 1.5%. Context becomes more important proportionally to the image size. In contrast from Parsenet [30], we do batch normalization for pool-6 feature, which will increase mean IoU by about 1.0%.

5.4.2 Supervision for Segmentation

To verify our supervision model on segmentation task, we train two models, one with supervision branch and another without supervision. To accelerate the training process, for the supervision branch, we remove the global fusion and simply add deconvolution layer in order to get the feature map same with the size of label. But for the final output prediction, we add batch normalization layer for both then concatenate features from pool6 layer and fc7 layer. To get the same size feature map with fc7 layer, we need to do unpooling for pool6 feature back to the size with fc7 layer, which is a one dimensional feature vector. We also use ”poly” learning rate policy to train the network with 1e-8 as base learning rate, 0.99 as momentum and power set to 0.9. We train the network with 150k iteration to achieve the 38.87 mean IoU shown in Table 5.1. Our method outperforms the well-know approach FCN [33] and show the effectiveness of the early supervision for the semantic segmentation problem.

model	FCN-32s[33]	ES-FCN
per_pixel_acc	61.75	68.47
per_label_acc	44.04	50.23
weighted_iou_acc	46.52	49.12
mean_IoU	35.10	38.87

Table 5.1: Pixel-wise semantic segmentation comparison on PASCAL Context dataset [39].

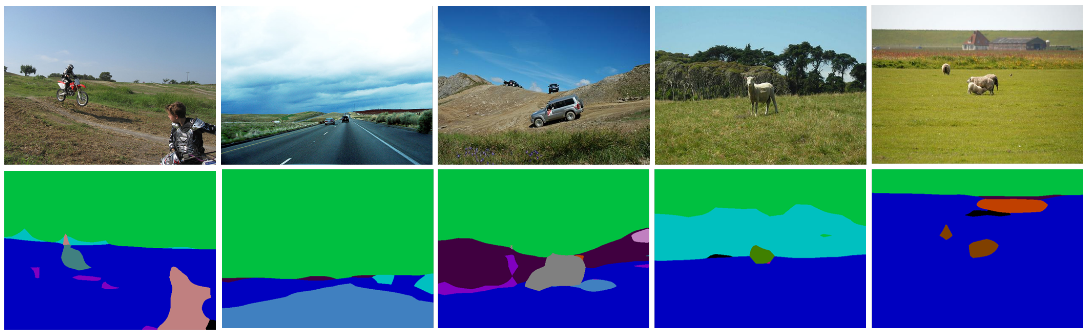

Figure 5.3: Some Semantic segmentation results using Early Supervision Full Convolutional Network (ES-FCN), where blue represents the grass, green represents the sky, light blue represents the ground and other colors represent other specific objects (referencing the object color corresponding to the list in PASCAL-Context Dataset[39])

5.4.3 Two Class Weather Classification

To validate our model for a new dataset, we evaluate our method using the most recent and largest a public weather image dataset available [34]. This two-class dataset consists of 10K sunny and cloudy images. For comparison, we adopt the same evaluation metric in [34] which is the normalized accuracy as $\max \{(a - 0.5)/(1 - 0.5),\ 0\}$, where a is the general accuracy. We following the same experimental setting in [38] which randomly selects 80% of the image from each class for training and the remain 20% of images are used for testing.

In order to distinguish three different semantic segmentation ensemble methods mention in Section 5.3.5, we name the first method: raw RGB images concatenate with 60 channel segmentation results (63 channel input for classification model) as Directly Ensemble; the second one: employing a convolutional layer with 1×1 kernel size to a pre-defined number of output(setting 3 in our experiment), used for feature selection and use element-wise product with raw image as Mixed Ensemble (3 channel input for classification model), and the third mode: generating a full segmentation map within one channel and concatenate with a 3 channel raw image as Unify Ensemble (4 channel input for classification model). The comparison of three different ensemble methods is shown in Table 5.3. The result shows that the Unify Ensemble provides the most compact semantic information and is the most accurate.

Table 5.2 shows the comparison with current state-of-the-art methods. We select two well known low level hand-crafted features, HOG [9], GIST [41] (top 3 rows in Table 5.2) and the delicate features which is specifically designed for the weather recognition. Our method achieves 95.2%, a new state-of-the-art performance standard on a two-class weather classification dataset. Although the CNN is a powerful neural network model especially in classification tasks [38], the additional semantic

Methods	NormAcc	Acc
GIST +SVM [41]	11.3%	89.3%
HOG + SVM [9]	38.5%	93.7%
Combined Feature [34] + SVM	41.2%	70.6%
Yen et al. [54]	24.6%	62.3%
Roser et al. [43]	26.2%	63.2%
Lu et al. [34]	53.1%	76.6%
Weather CNN [38]	82.2%	91.1%
Ours	90.4%	95.2 %

Table 5.2: Weather recognition comparison with current state-of-the- art methods.

type	ensemble method	Acc
1	Directly Ensemble	89.3%
2	Mixed Ensemble	93.7%
3	Unify Ensemble	95.2%

Table 5.3: Weather recognition results using different semantic ensemble methods.

information cues can leverage the CNN to obtain even more precise results.

To make our model more scalable, we extend our model for two more class of weather- rainy and snowy. We use Fliker crawler to grab 3000 images for each class. We also finetune from our original 2 class model and change the output layer for 4 outputs, see the result in Table 5.4.

5.5 Conclusion

We believe this is the first paper to propose an end-to-end convolutional network model which combines a segmentation model with a classification model, allowing for high-level visual recognition tasks. Our segmentation algorithm learns the pixel

Class	sunny	cloudy	rainy	snowy
Ours	95.1%	94.2%	88.91%	90.6%

Table 5.4: Weather recognition results on our extended weather data set.

level information for the representation of an image through the pretrain model, and captures the semantic information for whole images. The semantic segmentation information provided by the segmentation model gives leverage to the image classification in order to obtain better accuracy. This approach can then generally be deployed in the recognition task. We achieve outstanding performance on both public semantic segmentation datasets as well as weather classification datasets, compared to current state-of-the-art weather classification algorithms in use today (i.e., from the state-of-the-art 91.1% to 95.2%).

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2189–2202, 2012.

[2] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Computer Vision and Pattern Recognition*, 2014.

[3] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *arXiv preprint arXiv:1404.3606*, 2014.

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014.

[5] Zichong Chen, Feng Yang, Andreas Lindner, Guillermo Barrenetxea, and Martin Vetterli. How is the weather: Automatic inference from images. In *ICIP*, 2012.

[6] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 3286–3293. IEEE, 2014.

[7] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. cudnn: Efficient primitives for deep learning. *arXiv*, 2014.

[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, June 2005.

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[11] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2014.

[12] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[14] Pedro F Felzenszwalb, Ross B Girshick, and David McAllester. Cascade object detection with deformable part models. 2010.

[15] Vittorio Ferrari, Frederic Jurie, and Cordelia Schmid. From images to shape models for object detection. *International Journal of Computer Vision*, 87(3):284–303, 2010.

[16] P. Foucher, Y. Sebsadji, J.-P. Tarel, P. Charbonnier, and P. Nicolle. Detection and recognition of urban road markings using images. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1747–1752, Oct 2011.

[17] Ross Girshick. Fast r-cnn. *arXiv preprint arXiv:1504.08083*, 2015.

[18] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[19] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, 2010.

[20] G Holub Griffin and AD Perona. P. the caltech-256. Technical report, Caltech Technical Report, 2012.

[21] G Hinton, S Osindero, and Y Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, July 2006.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv*, 2015.

[23] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014.

[24] Philipp Krähenbühl and Vladlen Koltun. Learning to propose objects.

[25] Philipp Krähenbühl and Vladlen Koltun. Geodesic object proposals. In *Computer Vision–ECCV 2014*, pages 725–739. Springer, 2014.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[27] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989.

[28] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. *arXiv*, 2014.

[29] Li-Jia Li, Hao Su, Yongwhan Lim, and Li Fei-Fei. Objects as attributes for scene classification. In *Trends and Topics in Computer Vision*. Springer, 2012.

[30] Wei Liu, Andrew Rabinovich, and Alexander C Berg. Parsenet: Looking wider to see better. *arXiv*, 2015.

[31] Wei Liu, Hongliang Zhang, Bobo Duan, Huai Yuan, and Hong Zhao. Vision-based real-time lane marking detection and tracking. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 49–54, Oct 2008.

[32] Ziqiong Liu, Shengjin Wang, and Xiaoqing Ding. Roi perspective transform based road marking detection and recognition. In *Audio, Language and Image Processing (ICALIP), 2012 International Conference on*, pages 841–846, July 2012.

[33] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *arXiv*, 2014.

[34] Cewu Lu, Di Lin, Jiaya Jia, and Chi-Keung Tang. Two-class weather classification. In *CVPR*, 2014.

[35] Tomasz Malisiewicz, Abhinav Gupta, Alexei Efros, et al. Ensemble of exemplar-svms for object detection and beyond. 2011.

[36] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition - how far are we from the solution? In *Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN 2013)*, August 2013.

[37] J.C. McCall and M.M. Trivedi. Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *Intelligent Transportation Systems, IEEE Transactions on*, 7(1):20–37, March 2006.

[38] Sheng Huang Mohamed Elhoseiny and Ahmed Elgammal. Weather classification with deep convolutional neural networks. In *ICIP*, 2015.

[39] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[40] Eshed Ohn-Bar and Mohan Manubhai Trivedi. Learning to detect vehicles by clustering appearance patterns. 2015.

[41] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 2001.

[42] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[43] Martin Roser and Frank Moosmann. Classification of weather situations on single color images. In *IVS*, 2008.

[44] Robert E Schapire. A brief introduction to boosting. 1999.

[45] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[46] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014.

[47] Hongjun Song, Yangzhou Chen, and Yuanyuan Gao. Weather condition recognition based on feature extraction and k-nn. In *Foundations and Practical Applications of Cognitive Systems and Information Processing*. Springer, 2014.

[48] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv*, 2014.

[49] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[50] Stefan Vacek, Constantin Schimmel, and Rüdiger Dillmann. Road-marking analysis for autonomous vehicle guidance. In *EMCR*, 2007.

[51] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367. IEEE, 2010.

[52] Liwei Wang, Chen-Yu Lee, Zhuowen Tu, and Svetlana Lazebnik. Training deeper convolutional networks with deep supervision. *arXiv*, 2015.

[53] Tao Wu and A. Ranganathan. A practical system for road marking detection and recognition. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 25–30, June 2012.

[54] Xunshi Yan, Yupin Luo, and Xiaoming Zheng. Weather recognition based on images captured by vision system in vehicle. In *Advances in Neural Networks*. Springer, 2009.

[55] Shulin Yang, Liefeng Bo, Jue Wang, and Linda G Shapiro. Unsupervised template learning for fine-grained object recognition. In *Advances in Neural Information Processing Systems*, pages 3122–3130, 2012.

[56] Fatin Zaklouta and Bogdan Stanciulescu. Real-time traffic sign recognition in three stages. *Robotics and Autonomous Systems*, 62(1):16 – 24, 2014. New Boundaries of Robotics.

[57] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *Computer Vision–ECCV 2014*, pages 391–405. Springer, 2014.

Appendices