UNIVERSITY OF CALIFORNIA, MERCED

Learning to Recognise Objects and Actions for Intelligent Agents

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Electrical Engineering and Computer Science

by

Nakul Agarwal

Committee in charge:

Professor Ming-Hsuan Yang, Chair
Professor Shawn Newsam
Professor Sungjin Im

2019

The thesis of Nakul Agarwal is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Professor Shawn Newsam

Professor Sungjin Im

Professor Ming-Hsuan Yang Chair

University of California, Merced

2019

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2016	B. E. in Instrumentation and Control Engineering, Netaji Subhas Institute of Technology, University of Delhi, Delhi, India
2019	M. S. in Electrical Engineering and Computer Science, University of California, Merced

PUBLICATIONS

Nakul Agarwal, Vineeth Balasubramaniam, and C.V. Jawahar, *Improving Multiclass Classification by Deep Networks using DAGSVM and Triplet Loss*, In Pattern Recognition Letters (PRL), 2018.

Nakul Agarwal*, A.H. Abdul Hafez*, and C.V. Jawahar, *Connecting Visual Experiences using Max-flow Network with Application to Visual Localization*, In arXiv, 2018. (* indicates equal contribution)

ABSTRACT OF THE THESIS

Learning to Recognise Objects and Actions for Intelligent Agents

by

Nakul Agarwal
Master of Science in Electrical Engineering and Computer Science
University of California Merced, 2019
Professor Ming-Hsuan Yang, Chair

Computer vision involves a host of tasks, such as boundary detection, semantic segmentation, surface estimation, object detection, image classification, action localization, to name a few. For a holistic understanding of a scene, which is required by a lot of real-world applications, many of these tasks need to be combined together. For instance, an autonomous car should not only be able to detect other cars (object) but also if a pedestrian is walking (action). The former requires localizing the object, which can either be at the pixel level or bounding box level. The latter requires localizing the action, and by extension the actor, in both space and time. These problems are best dealt with approaches involving supervised learning models which rely on large annotated datasets, and so the problem becomes even more challenging when there is lack of labeled data.

In this thesis, we first tackle the problem of spatio-temporal action localization in an unsupervised setting. As the name suggests, it requires modeling of both spatial and temporal features. So, we propose an end-to-end learning framework for an adaptation method which aligns both spatial and temporal features and conduct experiments on the action localization task. To highlight the potential benefits for autonomous cars, we also construct and benchmark a new dataset which contains pedestrian actions collected in driving scenes. Then, for a holistic understanding of the scene, we shift our attention from localizing actions to recognising objects especially in a city street scenario. We do this by jointly dealing with the tasks of object detection and semantic segmentation. While the former localizes the individual instances of objects at the bounding box level,

the latter provides pixel level distinction but at the category level. We explore a novel observation that connects the two tasks and provide an end-to-end learning framework to exploit this connection.

Chapter 1

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, a lot of real-world applications are reaping its benefits. For example, Advanced Driver Assistance System (ADAS) has 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 manoeuvre a vehicle properly based on the surrounding and road conditions. On the other hand, surveillance systems have also started becoming automated. Surveillance is a repetitive and mundane task. This may cause performance dips for us human beings. By letting technology do the surveillance, we could focus on taking action if something goes amiss and the whole process could be much more efficient. Moreover, the same technology can be used for a variety of other applications which are, but not limited to, automated product delivery, traffic monitoring, transport networks, traffic flow analysis, understanding of human activity, home nursing, monitoring of endangered species, and observation of people and vehicles within a busy environment along many others to prevent theft and robbery.

All the above applications rely on vision based intelligent agents for automated func-

tioning. In this thesis, we focus our attention on recognising objects and actions, which are the building blocks of these vision based systems. We first build an adaptive model for localizing human actions in both space and time to solve the generalization issue in supervised learning methods. Then, we move on to the task jointly detecting and segmenting objects for a better understanding of a scene.

In Chapter 2, we present our approach for spatio-temporal action localization. Spatio-temporal human action localization is an important problem in computer vision that involves localization of actions in both space and time, and therefore requires modeling of both spatial and temporal features. This problem is typically studied in the context of supervised learning, where the learned classifiers operate under the assumption that both training and test data are sampled from the same underlying distribution. However, this assumption does not hold in situations where there is a significant domain shift, leading to poor generalization performance on the test data. In order to address this problem, we propose an end-to-end learning approach for unsupervised domain adaptation for spatio-temporal action localization. We show that for adaptation to work effectively, both spatial and temporal features must be adapted. We empirically verify the effectiveness of the proposed method using the UCF-Sports and UCF-101 benchmark datasets. Additionally, to highlight the potential benefits of the proposed algorithm in realistic applications, we used the AVA dataset to evaluate the generalization performance on a challenging Pedestrian Action Dataset (PAD) that we created in driving scenes for development of advanced driver assistance technologies.

In Chapter 3, we focus on the task of object detection and semantic segmentation. Object detection and semantic segmentation are two of the most fundamental problems for scene understanding in computer vision. When addressed jointly, it can be applied to many fields such as autonomous driving. While most of the existing works mainly rely on sharing the deep convolutional features for jointly dealing with the two tasks, we exploit a much deeper connection between object detection and semantic segmentation. We observe that the feature maps used by RPN (i.e., Region Proposal Networks), a popular module used in two stage object detectors, resemble a dense class-agnostic (i.e., foreground/background) segmentation map. We explore this observation to improve the functioning of RPN as well as improve performance for semantic segmentation. We

propose a framework called SegRPN where we endow Faster-RCNN with a semantic segmentation branch using a shared Feature Pyramid Network (FPN) backbone. The semantic segmentation branch is facilitated by a class agnostic segmentation module, which serves two purposes: (i) it provides a scale specific objectness prior for semantic segmentation and (ii) it supports the RPN in the segmentation branch which improves the functioning of the RPN in the detection branch. Experimental results on Cityscapes dataset demonstrate that the proposed SegRPN is able to improve both object detection and semantic segmentation results.

At last, we conclude the thesis in Chapter 4 and discuss possible future directions.

Chapter 2

Unsupervised Domain Adaptation for Spatio-Temporal Action Localization

2.1 Introduction

In recent years, there has been a growing interest for development of algorithms to address the problem of spatio-temporal human action localization due to its importance and applications. With the recent availability of benchmark datasets [76, 24] and progress in development of temporal neural network architectures [71, 6], numerous algorithms for spatio-temoral action localization have been proposed. However, as compared with conventional action recognition [17] and action segmentation [14], annotation of spatio-temporal action localization is more complex and costly. In particular, as observed in many crowded environments, the presence of multiple actors performing multiple concurrent actions in a single frame leads to an exhaustive labeling task. Transferring models pre-trained on label-rich domains would appear to be an attractive solution in such cases. However, existing supervised learning based approaches cannot yet generalize to unseen data. To address the problem of domain shift, various methods for unsupervised domain adaptation (UDA) have been proposed. Nevertheless, a majority of existing methods focus on images and not videos, catering to areas such as image classification [61, 73, 46, 19], semantic segmentation [64, 84] and object detection [11, 63]. One reason is attributed to the fact that a well-organized setting to develop

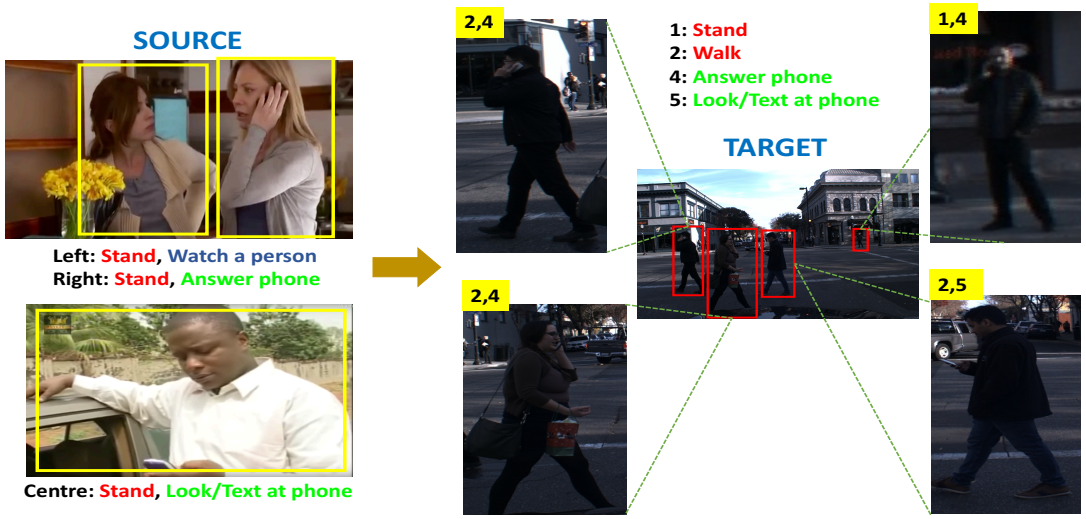


Figure 2.1: An illustration of an unsupervised domain adaptation task for spatio-temporal action localization. The source and target images are from the AVA and proposed PAD dataset, respectively, where each box is associated with multiple labels. Although the datasets belong to two completely different domains, the same atomic actions are present in both datasets, but in different contexts. The goal is to transfer these atomic actions between the two domains.

and benchmark the performance of domain adaptation algorithms for videos does not exist.

Recent UDA methods for video activity understanding are developed for: i) whole-clip action recognition, ii) action localization in the temporal domain, and iii) spatio-temporal action localization. Significant progress has been made but only at the video-clip level [32, 10]. This is primarily due to lack of data when it comes to UDA for spatial and/or temporal action localization.

Existing datasets, e.g., CMU [36], MSR Actions [81], UCF Sports [59], and JHMDB [33] provide spatio-temporal annotations but only for a small number of short video clips. Additionally, the whole video clip contains only a single person performing a single action. The DALY [76] and UCF-101 [69] datasets have a similar characteristic. Specifically, the majority of these videos contain a single action and only a small set have annotations with multiple persons. Furthermore, these datasets only have very few overlapping categories, making it difficult to evaluate domain adaptation algorithms. The

Table 2.1: Annotation setup of PAD vs other datasets with spatio-temporal annotations.

Dataset	Multi-Person	Multi-Label	Action Type
JHMDB	No	No	Everyday
UCF-101	Mostly No	No	Sports
UCF-Sports	No	No	Sports
DALY	Mostly No	No	Everyday
AVA	Yes	Yes	Everyday
PAD	**Yes**	**Yes**	**Everyday**

recently proposed AVA [24] dataset addresses some of the aforementioned limitations by providing a large-scale dataset for spatio-temporal action localization. This dataset contains annotations of multiple persons performing multiple concurrent actions in realistic settings. However, a dataset with similar size, annotation setting, and overlapping person activities has not been developed.

Motivated by these observations, we propose an end-to-end trainable unsupervised domain adaptation method for spatio-temporal action localization. We tackle the problem from both the approach as well as the data perspective. To bridge the domain shift gap for spatio-temporal action localization models, we align both spatial and temporal features. To evaluate the hypothesis for spatial-temporal action localization, we conduct different ablation studies to adapt what is learned from the UCF-Sports [60] dataset to scenes in the UCF-101 [69] database. Note that we refer to this setting as single label adaption (SLA), in which the actor in the frame is only performing a single action at any given instance. Our experimental studies show that it is necessary to adapt both spatial and temporal features for the task.

To demonstrate the effectiveness of the proposed algorithm in a real-world setting, we devised experiments on multi-label adaptation (MLA) settings, in which the actor in the frame performs multiple concurrent atomic actions at any given instance. However, as previously noted, AVA is the only dataset containing annotations that meet the desired experimental setting. Thus, we created a challenging pedestrian action detection (PAD) dataset with annotations, which consists of atomic actions performed by pedestrians at road intersections under different lighting conditions and visibility. A significant amount of frames in the dataset contain multiple pedestrians where each pedestrian per-

forms multiple concurrent atomic actions at any given instance. We show a comparison between the annotation setup of PAD and other datasets in Table 2.1. An illustration of the overall task with this annotation setting is shown in Figure 2.1. Experimental results on PAD highlight the challenges of the dataset as well as the problem of transfering models from two completely different domains.

The contributions of this work are summarized as follows. First, we present an end-to-end learning framework for unsupervised domain adaptation for spatio-temporal action localization. Second, we demonstrate that it is necessary to adapt both spatial and temporal features for spatio-temporal action localization. Third, to show the effectiveness of the proposed unsupervised domain adaptation approach for spatio-temporal action localization in real-world problems, we introduce and benchmark the proposed algorithm on a challenging dataset for pedestrian action detection in driving scenes.

2.2 Related Work

2.2.1 Spatio-temporal action localization

Spatio-temporal action localization has been an active research topic, where the goal is to localize the action in both space and time. Existing approaches can be categorized as either single frame or multi-frame. Most of the recent methods [24, 23, 53, 62, 68, 75] fall in the former category. These schemes extend object detection frameworks [21, 58] to first generate region proposals and then classify them into actions at the frame level using a two-stream variant which processes both RGB and flow data separately. The backbone of these networks is generally a 3D CNN (e.g., C3D [71] or I3D [6]). The resulting per-frame detections are then linked using dynamic programming [23, 68] or tracking [75]. Some recent approaches, however, aim to jointly estimate localization and classification over several frames [34] or use 3D convolutions to predict short tubes [29]. In this chapter, the spatial-temporal action localization model is inspired by [24], which uses an I3D backbone with an RPN.

2.2.2 Domain Adaptation

Domain adaptation aims to bridge the gap between the source and target data collected from different domains. Recent domain adaptation techniques under both supervised and unsupervised settings have been introduced for image applications [13]. Most efforts have been dedicated to applications involving image classification [61, 73, 46, 19, 65, 49, 25], object detection [11, 63], and semantic segmentation [64, 84]. Since the emergence of DANN [20], a common approach for domain adaptation is to utilize adversarial learning on the intermediate feature representations in order to align the feature distribution between the two domains [2, 72, 11]. In contrast, much less attention has been paid to adapt videos between domains, and especially for activity understanding. Existing domain adaptation efforts for activity understanding are designed for: i) whole-clip action recognition, ii) action localization in the temporal domain, and iii) spatio-temporal action localization. Some progress has been made in this area, but only for whole-clip action recognition [32, 10]. To the best of our knowledge, this work is one of the first to adapt spatio-temporal action localization under the unsupervised setting.

2.3 PAD Dataset

Although there exist several datasets [39, 55] for pedestrian behavior understanding, the focus and setting are different, making them less applicable to the problem studied in this work. As both the datasets contain task specific actions (pedestrian/driver behavior and pedestrian cross streets) which do not occur commonly across scenes, these images cannot be easily used for spatio-temporal action localization with the unsupervised domain adaption setting. On the other hand, the proposed PAD dataset consists of pedestrians performing commonly observed atomic actions around intersections in driving scenarios as shown in Table 2.2. These atomic actions are typically found in everyday activities other than driving scenarios, which can be used for the tasks of interest. The data collection strategy is discussed in the following section.

Table 2.2: The taxonomy of pedestrian action at intersection.

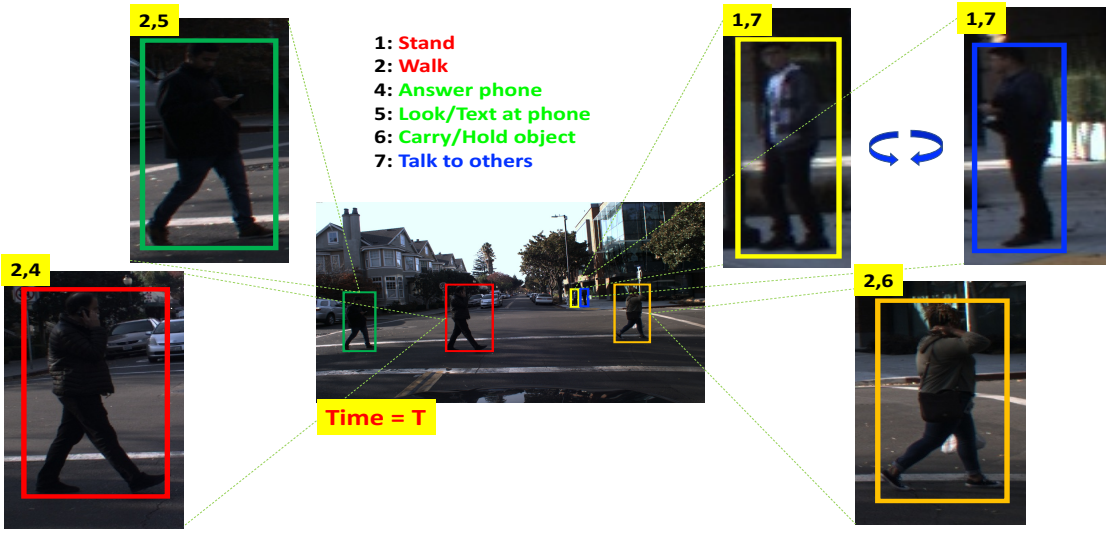
Primary	Secondary	Tertiary
pedestrian/group	pose	stand
		walk
		run
	person-object	look/text at cell phone
		answer phone
		carry/hold object
	person-person	talk to others

2.3.1 Data Collection and Annotation Methodology

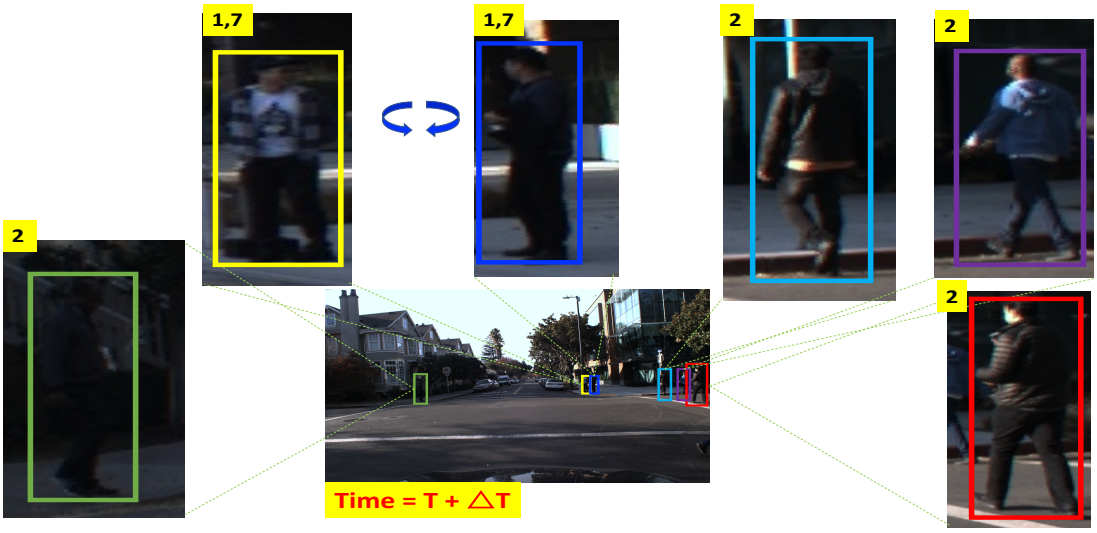
Understanding pedestrian behavior is perhaps the most important factor in realization of safe, robust, and ubiquitous deployment of self-driving and driver assistance technologies on urban roads. The PAD dataset was created in the San Francisco Bay Area using an instrumented vehicle equipped with a Point Grey Grasshopper video camera with a resolution of 1920 x 1200 pixels. Video was recorded at a frame rate of 30Hz with a field of view of 80 degrees. The atomic actions were labeled using an annotation method similar to that proposed by [24].

Three categories are defined for the pedestrian action annotations: i) *primary*, ii) *secondary* and iii) *tertiary*. Under the *primary* category, we check whether the actor is an individual pedestrian or part of a group of pedestrians. If it is part of a group, we do not provide action annotations since analysis of group activity is beyond the scope of this work. The *secondary* category consists of three subcategories: a) *pose*, b) *person-object* interaction and c) *person-person* interaction. The *tertiary* category comprises of the action itself. For *pose*, we consider three atomic actions, i.e., *stand*, *walk* and *run*. Under *person-object* interaction, *text/look at cell phone*, *answer phone* and *carry/hold object* are defined. Finally, *person-person* interaction consists of *talk to others*. The overall annotation structure is shown in Table 2.2.

Each action is labeled at the frame level, i.e., for each frame in the clip. Only actions of pedestrians with greater than 70 pixels in height were annotated. This threshold was empirically determined by the requirement that the average confidence of joint detection be higher than 0.4 in our pose estimation algorithm [5]. Note that the track ID of each



(a)



(b)

Figure 2.2: Examples from the PAD dataset. We represent the changes at a fixed intersection scene over time (few seconds) by showing before and after images in (a) and (b) respectively. The track id of a person is associated with the color of the bounding box. Each actor in the scene is cropped and enlarged for better visibility.

actor is annotated. We show this visually in Figure 2.2, where we represent the changes at an intersection scene over time (few seconds) by showing before and after images in Figure2.2(a) and Figure 2.2(b) respectively. Each person has a unique color bounding

box that represents the track id. For example, the pedestrian (red box) in Figure 2.2(a) is tracked even after he goes out of the scene and reappears at a different location after a few seconds.

The presence of multiple pedestrians engaged in concurrent actions that are typically atomic in nature makes the PAD dataset extremely challenging for not only labeling, but also for action recognition. An example of concurrent actions would involve, for instance, a pedestrian *talking on the phone* while *walking* as shown in Figure 2.2(a). Another challenge with the dataset is attributed to the large variations of pedestrian size (i.e. size of bounding box) across the frames. Figure 2.2 shows few such examples from the dataset.

The actions in these scenes require discriminating fine-grained differences, such as *looking at cell phone* where leveraging temporal context might also help. Discriminating these fine-grained differences becomes even more challenging when the visibility of the scene or pedestrian is low either due to lighting conditions or because the pedestrian is too small to even detect, as shown in Figure 2.2. In addition, the appearance also varies within an action class: local context for the action may change. For example, a pedestrian may be walking on the sidewalk vs on the road.

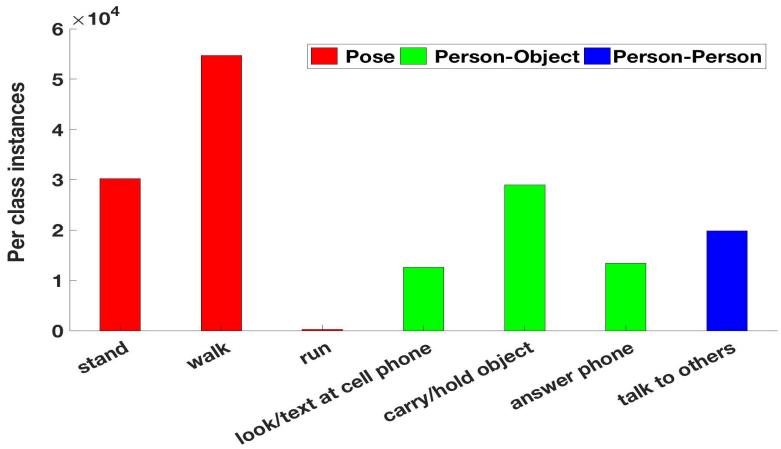


Figure 2.3: Size of each action class in the PAD dataset, with colors indicating the action category.

2.3.2 Dataset Statistics

The PAD daataet contains 182 video clips (without labels) for unsupervised training and 24 video clips (with labels) for evaluation, There are 15997 frames in the evaluation set. Figure 2.3 shows the distribution of action annotations in the PAD evaluation set. While the common classes such as *stand* and *walk* have reasonably high number of instances, classes such as *look/text at cellphone* and *answer phone* have relatively fewer instances, with *run* having the least amount of instances as expected.

We also show the variability in the bounding box size in Figure 2.4. As shown, the pedestrian size is typically much smaller than the full frame height, unlike those in the AVA dataset. The aspect ratio of these bounding boxes also hovers around 0.4, suggesting less variation in the broad pose of the pedestrian in driving scenes. Note that the classes defined in the PAD dataset do not incorporate such variations. This makes discriminating between these classes even more challenging. Similar to the AVA dataset, there are multiple labels for most of the pedestrian bounding boxes. All bounding boxes have a pose label, 65% of bounding boxes have at least 1 person-object interaction and 23% of them at least 1 person-person interaction label.

2.3.3 Training and test sets

PAD contains 24 annotated videos for evaluation and 182 videos (without labels) for unsupervised training. This setting is actually more in sync with the real-world

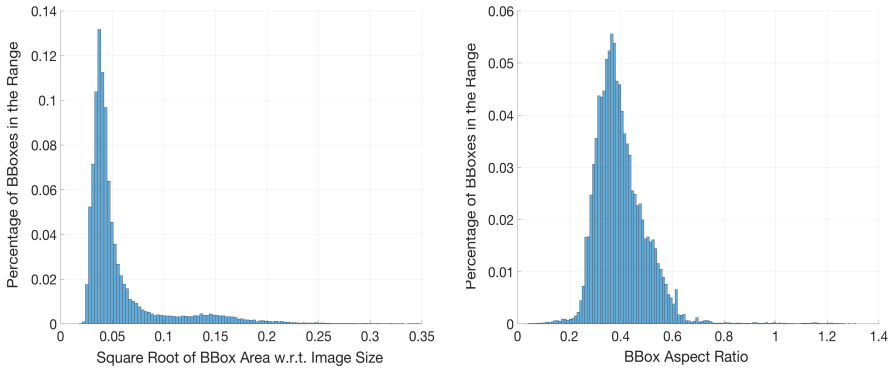


Figure 2.4: Size and aspect ratio variations of annotated bounding boxes in the PAD dataset.

problem of collecting large scale labeled data. Existing datasets [33, 69] that provide annotations at the frame level typically have a faster evolution of activity over frames, because of which every frame is used in both training and evaluation [24]. Although the evolution of activity in the PAD is slower over consecutive frames due to the nature of pedestrian activities in driving scenes, we use all the frames present in both the training and evaluation set. Similar to [24], each video-clip is divided into multiple overlapping segments where the middle frame (key frame) contains the annotation. This results in a total of 52k segments for training and 16k segments for evaluation.

2.4 Proposed Algorithm

Most recent approaches for spatio-temporal action localization rely on an object detection framework for generating actor proposals and a 3D CNN based backbone for explicitly modeling the temporal context for action recognition. While the object detection framework focuses on learning the spatial features of an input image, the 3D CNN models the temporal features using the input sequence of frames. To effectively adapt the model for spatio-temporal action localization, we show that both types of features are important and need to be adapted. The architecture of the proposed framework is shown in Figure 2.5.

2.4.1 Action Localization Model

Similar to [24], our model is based on the Faster-RCNN [58] for end-to-end localization and classification of actions [53]. To explicitly model the temporal context, I3D architecture [6] is incorporated. Our model is different from the base architecture [24] in several aspects. For temporal context modeling, the I3D model takes a video V of length T frames and generates the corresponding temporal feature representation using feature extractor TF.In this work, TF can be further decomposed into TF_1 and TF_2. Here, TF_1 temporally flattens the features from the fused mixed_4f layer of I3D, which has a spatial and temporal stride of 16 pixels and 4 frames respectively. This results in a compact representation of the entire input sequence.

For the actor proposal generation, we use a 2D ResNet-50 model as the spatial en-

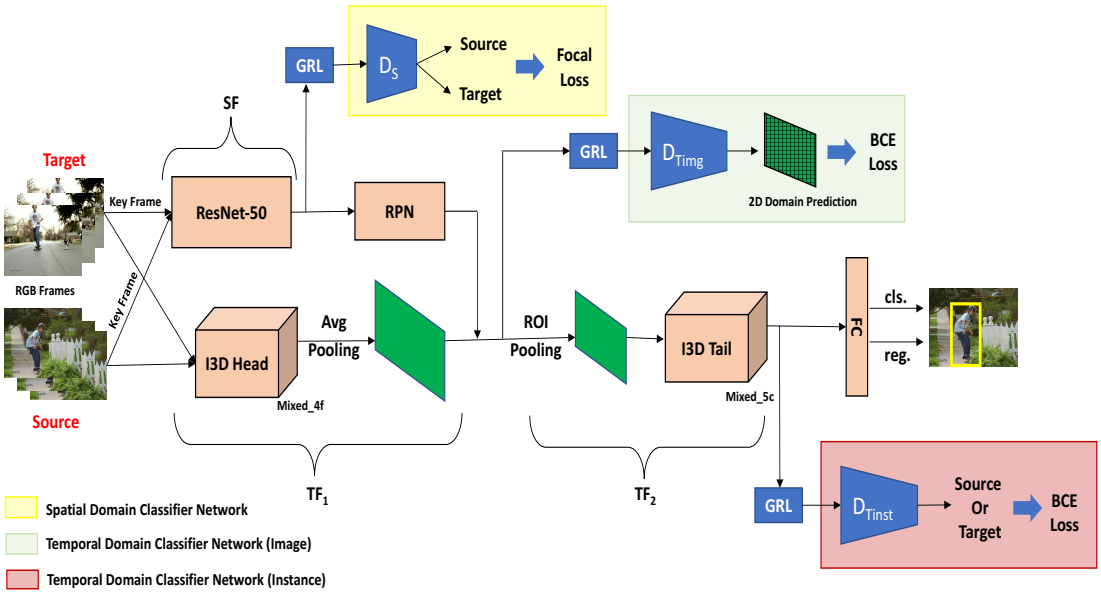


Figure 2.5: Proposed Network Architecture. The proposed algorithm aligns the distribution of both the spatial and temporal features of source and target domains for adapting actor proposals and action classification respectively. We use a spatial domain classifier network D_s to align the spatial features generated by SF. The temporal features are adapted on the image and instance level using their respective temporal domain classifier networks, i.e., D_{Timg} and D_{Tinst}. Image level features are extracted by TF_1 and instance level features are obtained from TF_2.

coder SF on the keyframe K as the input for the region proposal network (RPN). We note K is also the middle frame of an input clip to I3D. The proposals are generated using the conv4 block of Resnet [28]. As the spatial stride of the conv4 block is also 16 pixels, we directly use the actor RPN proposals on $TF_1(V)$ and perform ROI Pooling to obtain a fixed length representation of size 7x7x832. This feature representation is then passed through TF_2, which is an action classifier comprising of the remaining I3D layers up to mixed_5c and an average pooling layer that outputs a feature vector with size 1x1x1024. This feature is then used to learn a linear classifier for actions and a regressor for bounding box offsets.

The loss function of the action localization model is formulated:

$$\mathcal{L}_{act} = \mathcal{L}_{rpn} + \mathcal{L}_{cls} + \mathcal{L}_{reg} \tag{2.1}$$

where $\mathcal{L}_{rpn}$, $\mathcal{L}_{cls}$, $\mathcal{L}_{reg}$ are the loss functions for the RPN, final classifier and box regressor. The details regarding these individual loss functions can be found in the original paper [58].

2.4.2 Adaption in Space and Time

The adaptation process is comprised of two components: i) Actor Proposal Adaptation and ii) Action Classification Adaptation.

Actor Proposal Adaptation

We draw inspiration from recent works [11, 63] on adversarial learning for object detection to align the distribution of source and target features for the actor proposal network. Specifically, the spatial domain discriminator D_S is designed to discriminate whether the feature $SF(K)$ is from the source or the target domain. Motivated by [63], we train the domain classifier to ignore easy-to-classify examples while focusing on hard-to-classify examples with respect to the classification of the domain with the help of the Focal Loss [43]. This helps in preventing strong alignment between global features, which is both difficult and not desirable when there is a considerable domain shift. The loss is based on domain label d of the input image, where $d = 0$ refers to K from the source domain and $d = 1$ refers to K from the target domain. The estimated probability by D_S for the class with label $d = 1$ is denoted by $P \in [0, 1]$, where P is defined as:

$$P = \begin{cases} D_S(SF(K)), & \text{if } d = 1 \\ 1 - D_S(SF(K)), & \text{otherwise} \end{cases} \tag{2.2}$$

We formulate the spatial discriminator loss function as:

$$\mathcal{L}_{\mathcal{D}_\mathcal{S}} = -(\frac{1}{n_s}\sum_{i=1}^{n_s}(1 - P_i^s)^\gamma \log(P_i^s) + \frac{1}{n_t}\sum_{j=1}^{n_t}(P_j^t)^\gamma \log(1 - P_j^t)) \tag{2.3}$$

where n_s and n_t denote number of source and target samples in a minibatch respectively, and γ controls the weight on hard to classify examples.

The Gradient Reversal Layer (GRL) [20] is placed between the spatial domain discriminator D_S and spatial feature extractor SF. It helps SF produce domain invariant features $SF(K)$ that fool the discriminator while D_S tries to distinguish the domain.

Action Classification Adaptation

We adapt the temporal features at both the image and instance level. Specifically, we use TF_1 as a feature extractor for adaptation at the image level and TF_2 for adaptation at the instance level. The temporal feature extractor TF_1 takes a video clip V of length T frames and generates a compact feature representation $TF_1(V)$.The temporal domain discriminator D_{Timg} takes $TF_1(V)$ as input and outputs a 2D domain classification map $Q = D_{Timg}(TF_1(V)) \in \mathbb{R}^{H \times W}$. The parameters H and W are determined based on the resolution of V as the spatial strides of TF_1 and D_{Timg} are fixed. We then apply binary cross-entropy (BCE) loss on Q based on the domain label d of the input video V, where $d = 0$ if V belongs to the source distribution and $d = 1$ if V belongs to the target distribution. The loss function for D_{Timg} is formulated as:

$$\mathcal{L}_{D_{Timg}} = -\sum_{h,w} d \log Q^{(h,w)} + (1-d)\log(1 - Q^{(h,w)}) \tag{2.4}$$

where h and w correspond to the spatial index of an activation in Q.

The instance level representation generated by TF_2 refers to the ROI-based feature vectors before they are fed to the final category classifiers (i.e., the "FC" layer in Figure 2.5). The instance level temporal domain classifier D_{Tinst} takes the feature vector $TF_2(TF_1(V))$ as input and outputs a domain classification output for the j-th region proposal in the i-th image as $R_{i,j}$. A standard cross entropy loss is applied to generate the final output. The corresponding loss function is formulated as:

$$\mathcal{L}_{D_{Tinst}} = -\sum_{i,j} d_i \log R_{i,j} + (1-d_i)\log(1 - R_{i,j}) \tag{2.5}$$

where $d = 0$ if V belongs to the source distribution and $d = 1$ if V belongs to the target distribution.

2.4.3 Overall Objective

The overall objective combines losses from the action localization model and the domain adaptation components. We denote the overall adversarial loss from domain adaptation components as:

$$\mathcal{L}_{adv}(SF, TF, D) = \mathcal{L}_{\mathcal{D}_S} + \mathcal{L}_{D_{Timg}} + \mathcal{L}_{D_{Tinst}} \tag{2.6}$$

In Section 2.5, we conduct ablation studies to demonstrate the impact of each component.

For the adaptation task $s \rightarrow t$, given the source video V^s and target video V^t, and by extension their corresponding key frames K^s and K^t respectively, the overall min-max loss function of the adaptive spatio-temporal action localization model is defined as the following:

$$\min_{SF,TF} \max_{D} \mathcal{L}(V^s, K^s, V^t, K^t) = \mathcal{L}_{act}(V^s, K^s) + \lambda \mathcal{L}_{adv}(SF, TF, D) \tag{2.7}$$

where λ controls the trade-off between the action localization loss and adversarial training loss.

2.5 Experiments and Analysis

In this section, we evaluate the proposed approach on two settings: i) Single Label Adaptation (SLA) and ii) Multi-Label Adaptation (MLA). Under the SLA setting, we focus on the scenario of adapting from a smaller annotated domain to a much larger and diverse dataset. For the MLA setting, we choose to adapt everyday actions from a large dataset with annotations to the relatively smaller dataset depicting the real-world problem of autonomous navigation. In the following, we first describe the datasets and metric used for evaluation and then move on to discuss experimental results of our adaptation tasks.

2.5.1 Datasets and Metric

In addition to the proposed PAD dataset, we use the following datasets for our experiments.

UCF Sports

UCF Sports [60] is one of the earliest action recognition datasets consisting of various sports actions collected from broadcast television channels including ESPN and BBC. The dataset includes 10 actions, out of which we only use 4 for our experiments: *Diving*, *Golf-Swing*, *Horse-Riding*, *Skate-Boarding*.

UCF-101

UCF-101 [69] is also dedicated to action classification with more than 13000 videos and 101 classes. However, we only use 4 classes which are common with UCF-Sports, out of a 24-class subset with spatio-temporal annotations provided by [68]. We conduct experiments on the official split 1 as is standard.

AVA

AVA [24] is a recently proposed large-scale dataset for human activity understanding which consists of 80 atomic everyday actions. It contains spatio-temporal annotations collected under a realistic setting: a majority of its frames consists of multiple actors performing multiple concurrent actions. Due to its annotation setting, we use the dataset under the MLA setting.

Metric

We use the standard evaluation protocol for our experiments. We report the frame-level mean average precision (frame-mAP) using an IoU threshold of 0.5. For each class, we compute the average precision and report the average over all classes. We also use the video-level mean average precision (video-mAP) following the methodology provided in [53].

2.5.2 Implementation Details

The ResNet-50 and I3D networks are initialized with ImageNet and Kinetics [35] pre-trained models respectively. During training, we fix the input resolution to be 320×400 pixels. One key difference between the SLA and MLA setting is that the action labels are not mutually exclusive in the latter. Therefore, different loss functions are required for the two settings [24]. For applying the proposed adaptation method, we first pre-train the action localization network using the source domain images, and then fine-tune for adaptation. All feature layers are jointly updated during training using Stochastic Gradient Descent (SGD). We use different training schedules for the SLA

Table 2.3: Frame and Video mAP results for adaptation from UCF-Sports to UCF-101. Average Precision (%) is evaluated on target images.

UCF-Sports → UCF-101						
Method	Diving	Golf-Swing	Horse-Riding	Skate-Boarding	Frame-mAP	Video-mAP
I3D+RPN (Source only)	7.7	57.2	31.5	40.7	34.3	57.1
Temporal (Image level)	12.8	65.4	40.6	42.8	40.4	61.0
Temporal (Instance level)	12.5	65.5	41.2	42.8	40.5	61.6
Spatial	14.5	**65.8**	52.8	52.5	46.4	68.9
Spatial + Temporal (Image + Instance)	**18.4**	64.1	**64.5**	**54.7**	**50.4**	**73.6**
Oracle	92.3	98.1	94.2	91.8	94.1	99.0

and MLA experiment settings. The details of the two settings can be found in the corresponding experimental section. We implement the proposed method with Pytorch [51].

2.5.3 Single Label Adaptation

The SLA setting focuses on the case where the annotated person is only performing a single action at any given instance. Most of the existing datasets fall under this setting, which is also accompanied by a single person being annotated in the frame. We make use of the of the UCF-Sports and UCF-101 datasets as source and target domain for this task. Both of these datasets have 4 classes in common with spatio-temporal annotations: *Diving*, *Golf-Swing* and *Horse-Riding*, *Skate-Boarding*. We train the base model on source data for 60k iterations with a batch size of 2 per GPU (2 GPUs in total), mini-batch size of 256 for actor RPN and 64 for action classifier following [24]. We warm-start the learning rate from 0.00001 to 0.001 in 3K steps using linear annealing for stabilizing training and then use cosine learning rate decay [47]. Batch-norm updates are disabled for Resnet-50 but not for I3D. The adaptation network is fine-tuned for another 5k iterations using a learning rate of .0005 for the discriminator and .0001 for rest of the network.

We choose the pair of UCF-Sports and UCF-101 because in addition to being single-labeled datasets, UCF-Sports is also much smaller in size and less diverse compared to UCF-101. This helps us in examining the adaptation from a relatively smaller dataset to a large unlabeled domain, which has its own significance in the current world. In the modern world that we live in, with the advancement of digital cameras, collecting large

amount of images or videos has become relatively easy. However, annotating this large amount of data still remains a challenge, especially for video activity understanding. With our experimental setting, we also want to show that we can harvest more from the existing resources, or even collect data at smaller scale which is easier to annotate, and adapt them to a larger, diverse and challenging environment.

The results are shown in Table 2.3. The baseline and oracle results show the the difficulty and huge performance gap on the target dataset UCF-101. We're able to show considerable improvement over the baseline by adapting both spatial and temporal features individually. For aligning the temporal features, both image level as well as instance level adaptation yields a similar frame-mAP improvement of 6.1% and 6.2% respectively. A similar behaviour is also reflected in the video-mAP results, with an improvement of 3.9% and 4.5% for image and instance level respectively. However, aligning the spatial features, which is responsible for adapting the actor proposals, yields 12.1% (frame-mAP) and 11.8% (video-mAP) improvement over the baseline. This emphasizes the importance of localizing the action in space, which is also intuitive, because you need to localize the action first before classifying it. We also show that the combination of aligning both spatial and temporal features gives the best results, with an improvement of 16.1% (frame-mAP) and 16.5% (video-mAP) over the baseline.

2.5.4 Multi Label Adaptation

The MLA setting on the other hand, focuses on the case where the annotated person is performing multiple concurrent actions at a given instance. This setting is more aligned with the real-world scenarios, because as humans, our actions at any instance are built up of multiple atomic actions. Apart from the AVA [24] dataset, there does not exist another dataset with this annotation setting. Therefore, we collect and annotate our own small but challenging PAD dataset, for the purpose of evaluating our proposed algorithm on the task of autonomous navigation. In addition to being multi-label, the majority of frames in AVA and PAD also have multiple people annotated in a single frame, which makes the task of spatio-temporal action localization even more challenging.

Since we only have labels for the evaluation set in PAD, we use AVA and PAD as source and target domain respectively. In total these datasets have 6 classes in common:

Method	I3D+RPN (Source Only)
Stand	28.8
Walk	22.1
Answer Phone	0.5
Carry/Hold an object	7.9
Frame-mAP	14.8

Table 2.4: Frame-mAP results for generalization from AVA to PAD. Average Precision (%) is evaluated on target images.

Stand, *Walk*, *Answer Phone*, *Hold/Carry an Object*, *Talking to others* and *Run*. However, we only use the first 4 classes for our experiments. This is because *Talking to others* class in PAD only refers to the case when pedestrians are talking to each other, whereas in AVA it also includes a lot of instances when a person is only talking to themselves or someone else who is not present in the frame. Additionally, PAD does not have enough instances of the *Run* class for a reliable evaluation. We train the base model for 125k iterations using a batch size of 4 per GPU (8 GPUS in total), and warm-start the learning rate from 0.00001 to 0.001 in 5K steps using linear annealing for stabilising training. The rest of the training settings are the same as the ones under SLA setting.

The results are shown in Table 2.4. The baseline results are quite poor on the PAD dataset, especially for subtle classes such as *Answer Phone* and *Hold/Carry an Object*. There are multiple reasons for this behaviour: i) The primary reason for the poor performance is that AVA and PAD belong to two completely different domains, which is why the classifier trained on AVA cannot generalise well to PAD, ii) We train on AVA with very low resolution of 320×400 pixels, while we evaluate on the full resolution of PAD of 1200×1920 pixels. We do this keeping in mind the resources required to train a large-scale video dataset like AVA, iii) As shown in Figure 2.4, PAD mostly contains very small size annotated pedestrians which are hard to even detect, let alone classify their actions. This is the primary reason why performance is poor on subtle classes such as *Answer Phone* and *Hold/Carry an Object* perform poorly, as these classes not only require the person to be visible but also the object in question.

2.6 Summary

In this chapter, we propose a novel approach for unsupervised domain adaptation for spatio-temporal action localization. We show that in order to adaptation to work effectively, both spatial and temporal features need to be adapted. We show promising results using two benchmark dataset UCF-Sports and UCF-101.
In order to evaluate the proposed approach on a real-world problem of autonomous navigation, we also introduce a new and very challenging dataset called Pedestrian Action Detection (PAD) dataset. We use the recently introduced AVA dataset to evaluate the performance on PAD, and use the results to highlight the difficulty of the PAD dataset as well as the problem of transferring models from two completely different domains.

Chapter 3

SegRPN: Scale Aware Joint Object Detection and Semantic Segmentation

3.1 Introduction

The task of object detection is to identify in an image all objects of predefined categories and localize them via bounding boxes. Semantic segmentation operates at a finer scale where the goal is to assign a class label to each pixel. While there has been significant progress in the individual tasks of object detection [58, 44, 43, 42, 56, 57] and semantic segmentation [45, 7, 83, 9, 80], only few works have tackled them jointly. Existing approaches could benefit from solving these tasks jointly [16, 4, 70, 38]. For example, object detection should be easier if we know the semantics of the scene at the class-agnostic level, i.e. which pixels belong to the objects (e.g., car, person) and which belong to the background (e.g., sky, building). Conversely, semantic segmentation should be easier if we know where the object of interest is. In fact, it has been shown in the past that semantic segmentation usually used as a multi-task supervision can help object detection [48, 40, 26], and object detection used as a prior knowledge can improve semantic segmentation [54, 26]. Therefore, these two tasks are highly related.

Joint object detection and semantic segmentation has attracted a lot of attention in the past few years, leading to some interesting works. These works can broadly be summarised into four categories: (i) a common encoder with parallel branches for object

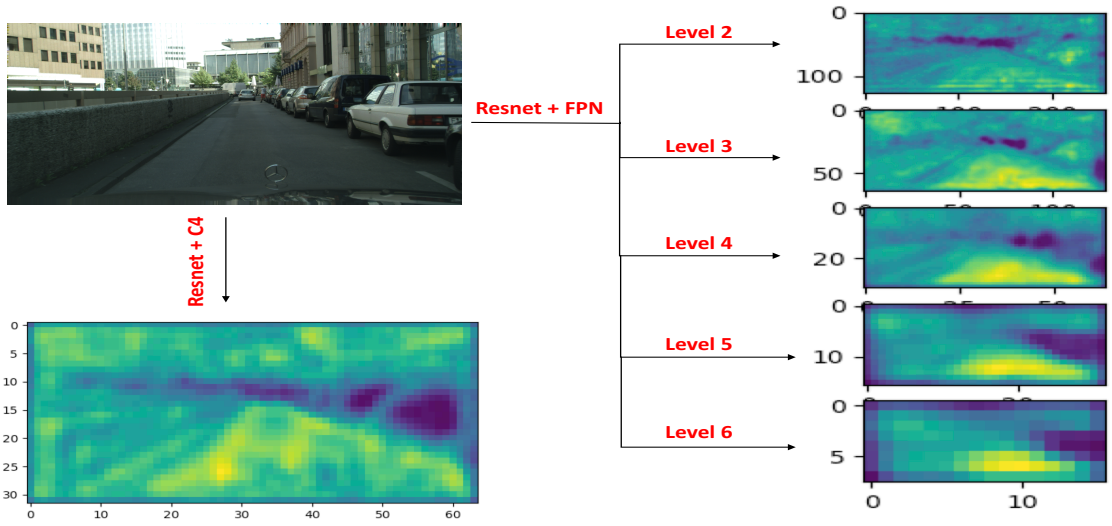


Figure 3.1: RPN feature maps as class-agnostic segmentation maps. Feature maps are shown for two different backbone networks: Resnet+C4 and Resnet+FPN. In case of FPN, maps of each level (2-6) have been visualized. The darker regions have lower activation values while the brighter have higher activation values.

detection and semantic segmentation attached to the last layers of the encoder [3, 70], (ii) same as (i) but with features from semantic segmentation branch refining object detection [48, 82], (iii) encoder-decoder network where the concatenated feature map from each layer of the decoder is used for semantic segmentation, whereas each layer of the decoder focuses on multi-scale object detection [16] and (iv) encoder-decoder network where each layer of the decoder is simultaneously used for object detection and semantic segmentation [4]. Although the above methods have shown to be effective by jointly training the two tasks, the improvement in the performance of object detection and semantic segmentation has not been much. One possible reason is that most of the above methods still mainly rely on simply sharing convolutional features for the two tasks, without focusing on a particular aspect of the pipeline responsible for semantic segmentation and object detection.

In this chapter, in addition to sharing convolutional features for the two tasks through the common encoder-decoder architecture, we focus our attention on a much deeper connection between object detection and semantic segmentation. We observe that the feature maps used by RPN resemble a dense class-agnostic (i.e., foreground/background)

segmentation map. We observe this across different backbone networks, two of which have been shown in Figure 3.1. We use this observation to improve the functioning of the RPN. Now the goal of the RPN is to produce candidate regions (i.e., proposals) which have the potential of containing an object, which forms the first part of a two stage object detector. These proposals are filtered out from all possible proposals based on probability score maps generated by RPN, as well as non-max suppression. These proposals then go on for final classification and regression, which forms the second part. Therefore, the RPN, and consequently the RPN input feature map as well as the RPN probability map play a very crucial role in object detection. If the first stage (i.e., RPN) itself misses out on good proposals, the second stage cannot recover them. Therefore, we propose a framework called SegRPN where we endow Faster-RCNN with a semantic segmentation branch using a shared Feature Pyramid Network (FPN) backbone. The semantic segmentation branch is facilitated by a multi-scale class agnostic segmentation module, which has another RPN (i.e. RPN_{mask}) attached to it. We improve the RPN (i.e. RPN_{det}) in the detection branch using RPN_{mask} in the segmentation branch in two ways: (i) we fuse the input feature maps of the RPN_{mask} score classifiers and bounding box regressors at different scale levels with the corresponding feature maps from RPN_{det}, (ii) we fuse the probability maps generated by RPN_{mask} score classifier at different scale levels with the corresponding ones in RPN_{det}. This helps us improve the quality of proposals generated by RPN_{det} both in terms of precision and recall.

Since RPN_{mask} is attached to a mutli-scale class-agnostic segmentation module, it helps us learn scale specific features for class-agnostic segmentation. This means the higher resolution feature maps in the class-agnostic segmentation module focus on segmenting the lower scale objects, whereas the lower resolution feature maps focus on segmenting the higher scale objects. We then use these scale specific features to improve the semantic segmentation performance, which becomes possible since the class-agnostic segmentation module is attached to the class-specific module.

The contributions of this work are summarised as follows: First, we explore a novel observation that the feature maps used by RPN represent a dense class agnostic segmentation map, and treat it as a much deeper connection between object detection and semantic segmentation. Second, we propose a framework called SegRPN for joint ob-

ject detection and semantic segmentation that effectively exploits this said connection. Third, we show why leveraging the scale of objects is an integral part of SegRPN.

3.2 Related Work

Before we introduce our approach, we review in this section techniques for object detection, semantic segmentation and past attempts at jointly dealing with the two tasks.

3.2.1 Object Detection

The goal of object detection is to classify all objects and localize them via bounding boxes. The methods of object detection can be broadly summarized into two categories: one stage methods and two stage methods. Two stage method, as popularized in the R-CNN framework [22] and its variants [21, 58], is a proposal driven mechanism where the first stage generates a *sparse* set of candidate object locations and the second stage classifies each candidate location as one of the foreground classes or as background using a convolutional neural network. These methods have dominated the field of object detection and are the most representative frameworks among the two stage methods. Over the years, there have been some notable extensions to this framework, especially the ones based on multi-scale features with strong semantics [42, 26].

One-stage methods, on the other hand, are applied over a regular, *dense* sampling of object locations, scales and aspect ratios. Overfeat [66] was one of the first modern one-stage object detector based on deep networks. It was more recently followed by SSD [44, 18], YOLO [56, 57] and RetinaNet [43]. These detectors have been tuned for speed and so are much faster, but their accuracy still trails that of two-stage methods. In this chapter, we adapt a two-stage method for the object detection pipeline since our focus is not on improving speed.

3.2.2 Semantic Segmentation

Semantic segmentation aims to predict the semantic label of each pixel in an image. It has achieved significant progress in the past few years [45, 7, 83, 9, 80]. The methods

of semantic segmentation can also be broadly categorised into two categories: encoder-decoder methods and spatial pyramid methods. The encoder-decoder methods contain two subnetworks: an encoder subnetwork and a decoder subnetwork. The encoder subnetwork is usually based on the standard CNN models (e.g., VGG [67], ResNet [28], DenseNet [30]) pre-trained on ImageNet [15]. It extracts strong semantic features and reduces the spatial resolution of feature maps. The decoder subnetwork on the other hand, gradually upsamples these feature maps with reduced spatial resolution. While some methods [1, 50] directly upsample the feature maps using max-pooling indices of the encoder subnetwork, others [52, 41, 80] extract context information by adopting skip-layer connection between the feature maps from the encoder and decoder subnetworks.

Spatial pyramid methods rely on exploiting multi-scale information, which is extracted from the last output feature maps using the idea adopted from spatial pyramid pooling [27]. Specifically, this multi-scale information can also be utilised in different ways. PSPnet [83] propose pyramid pooling module, which downsamples and upsamples the feature maps in parallel. Some other prominent works [7, 8, 74, 78] propose to use multiple convolutional layers of different atrous rates in parallel (called ASPP) to extract multi-scale features. For our work, we opt a simple design for our semantic segmentation branch, appending a set of conolvutional layers on top of the FPN backbone in a multi-scale fashion.

3.2.3 Joint Object Detection and Semantic Segmentation

The task of jointly dealing with object detection and semantic segmentation has attracted a lot of attention in the past few years, where the goal is to simultaneously detect objects and predict pixel semantic labels by a single network. Recently, researchers have done some attempts. A graphical model to holistic scene understanding is proposed in [79]. [70] propose a joint object detection and semantic segmentation framework by sharing the encoder subnetwork. A real-time approach to jointly solve this task was also introduced in [16]. More recently, [4] propose an encoder-decoder network where each layer of the decoder is simultaneously used for object detection and semantic segmentation.

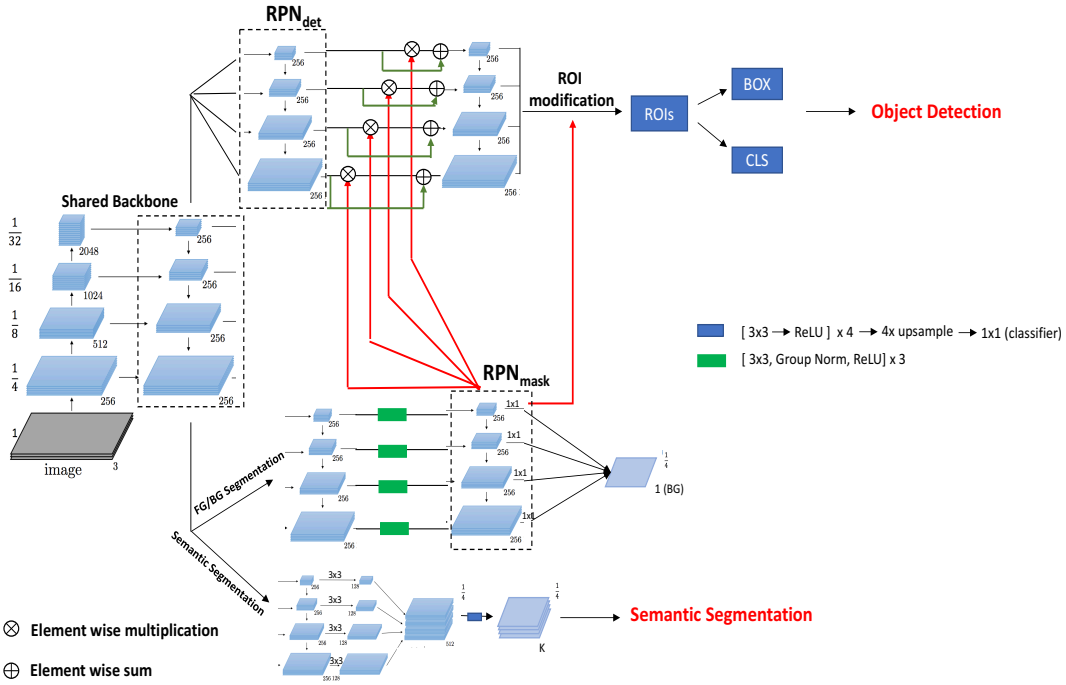


Figure 3.2: Proposed Network Architecture. We use a Resnet+FPN backbone, which is shared by the object detection and semantic segmentation branches.

Almost all of the above works rely on sharing convolutional features in some or the other way, without giving a concrete explanation of why these features are connected. Although a lot of progress has been made in this field, we argue that there is still room for further improvement. To support this claim, in this chapter, we explore a novel connection between the two tasks of object detection and semantic segmentation. We subsequently provide a framework for exploiting this said connection and show some positive experimental results.

3.3 SegRPN

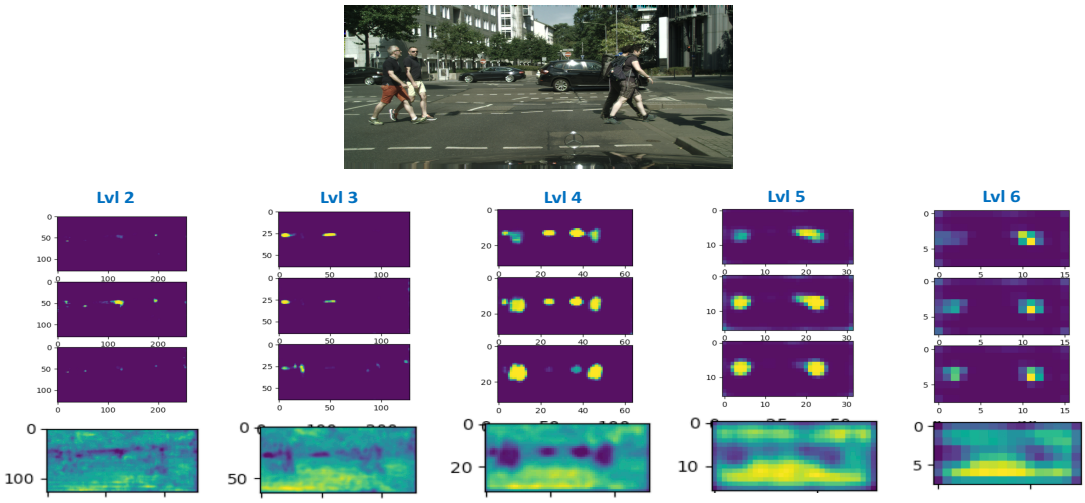
Different from the existing works dealing with joint object detection and semantic segmentation, we explore a much deeper connection between object detection and semantic segmentation and build our model to exploit said connection. We observe that the feature maps used by RPN resemble a dense class-agnostic (i.e., foreground/background)

segmentation map. The architecture of the proposed framework is shown in Figure 3.2. We endow the Faster-RCNN framework with a semantic segmentation branch using a shared Feature Pyramid Network (FPN) backbone. We first discuss this observation in detail in Section 3.3.1, and then move on to explaining the different components of the model.

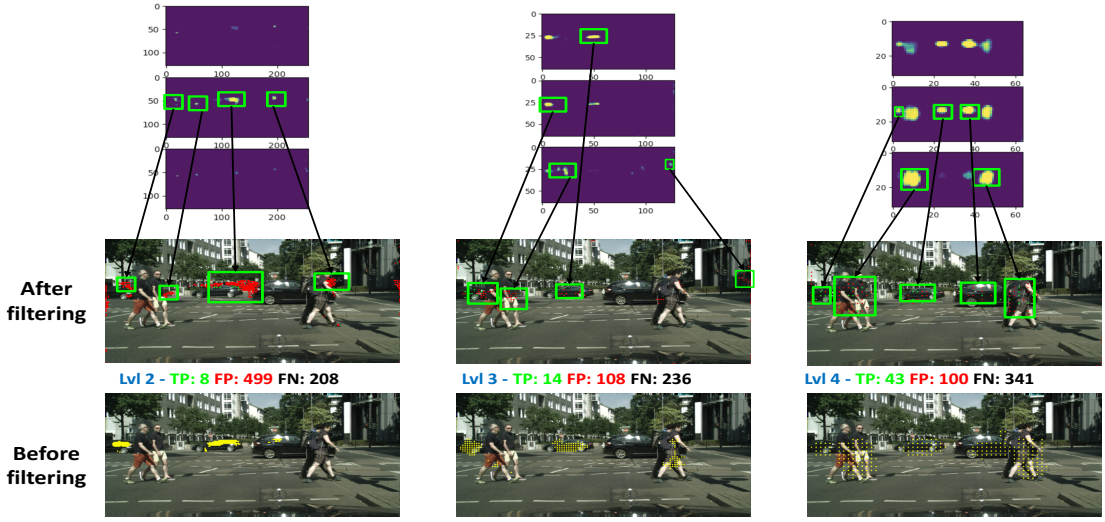
3.3.1 RPN Feature Map as Class-Agnostic Segmentation Map

The RPN is the first stage of Faster R-CNN, and it's job is to generate regions of interest (i.e., ROIs) which potentially contain the object. When used on top of FPN, each level of the pyramid generates ROIs corresponding to a fixed anchor size. The lower the level of the FPN, the higher the resolution and the smaller the objects it focuses on. The RPN generates these ROIs using its own binary classifier (i.e. foreground vs background) and box regressor, which take as input a set of feature maps. Using these features maps, the RPN generates bounding box offsets and probability maps representing pixels whose anchors have a possibility of containing the objects. We visualize both the input RPN feature maps as well as the generated ROI probability map for an image from the Cityscapes dataset in Figure 3.3.

For our work, we stick with the standard FPN setting of 5 scales and 3 aspect ratios of $\{32, 64, 128, 256, 512\}$ and $\{.5, 1, 2\}$ respectively. This means that there are 5 pyramid levels (2-6), each generating a single RPN feature map and three ROI probability maps corresponding to each of the three aspect ratios. The RPN feature map along with its corresponding ROI probability maps are shown for each of the five pyramid levels in Figure 3.3 (a). There are two interesting observations here; First, as you can see, the RPN feature maps have a lower activation region around the foreground objects in the image at all levels, where each level focuses on a certain scale of object. These maps resemble a class-agnostic segmentation map where the foreground objects have a lower activation, trying to almost mimic the contour of the object. This is especially visible in the feature map of level 3, where the human and car contours are clearly visible. Second, the generated ROI probability map behaves in a complete opposite manner to the RPN feature map, having higher activation for the same region with lower activation in the RPN feature map. This suggests that the RPN classifier acts as a low-pass filter, trying



(a)



(b)

Figure 3.3: In (a), visualization of the RPN feature maps and ROI probability maps for an image in the Cityscapes dataset is shown. Filtering of proposals by RPN based on ROI probability maps is shown in (b), where yellow points denote pixels with proposals having an overlap of greater than 0.5 with the ground truth bounding box before filtering. Red and green points are pixels denoting false positives (FPs) and true positives (TPs) respectively, after filtering.

to invert the activations of the RPN feature map. We also analyse the reason behind these two interesting observations in detail.

The RPN is designed in such a way that each pixel in the RPN feature map generates 3 proposals corresponding to the three aspect ratios. This results in a lot of proposals when combined over all the FPN levels. The RPN filters these proposals using the ROI probability score and non-maximal suppression (NMS), to form a batch of positive and negative proposals with a ratio of 1:3. We show that this filtering done by the RPN based on the ROI probability score misses out on a lot of candidate regions, and we argue that this happens because these candidate regions were not segmented in the RPN feature map. In Figure 3.3 (b), we show the effect of the filtering done by the RPN as well as the importance of the ROI probability map. The pixels responsible for generating the proposals having an overlap of greater than 0.5 with the ground truth bounding box before filtering are shown in yellow at the bottom. After filtering, we show the true positives (TPs) and false positives (FPs) in green and red color respectively. These TPs and FPs are directly related to regions in the ROI probability map, as shown in Figure 3.3 (b). As you can see, a lot of yellow pixels covering the object before filtering are missed once the filtering is done, which results in very few TPs and a lot of FNs and FPs at every level of the FPN. Note that we stick with the standard setting of using 2000 proposals after filtering by the RPN, and we show the exact number of proposals belonging to TPs, FPs and FNs for the three levels (2-4) individually in Figure 3.3 (b). We argue that this happens because these yellow pixels do not have a higher activation value in the corresponding ROI probability map, which in turn is caused by these regions not being segmented in the RPN feature map.

3.3.2 Class-Agnostic Segmentation Branch

As mentioned in Section 3.3.1, the RPN feature maps rely on its segmentation characteristic for subsequent filtering of the generated proposals. However, these maps are not entirely accurate. So, we append a class agnostic module in parallel to our segmentation branch, which produces a binary (i.e., foreground/background) segmentation map. The estimated probability for the binary cross entropy (BCE) loss function of this branch for background class is denoted by $p \in [0, 1]$. This branch serves two purposes: (i) it

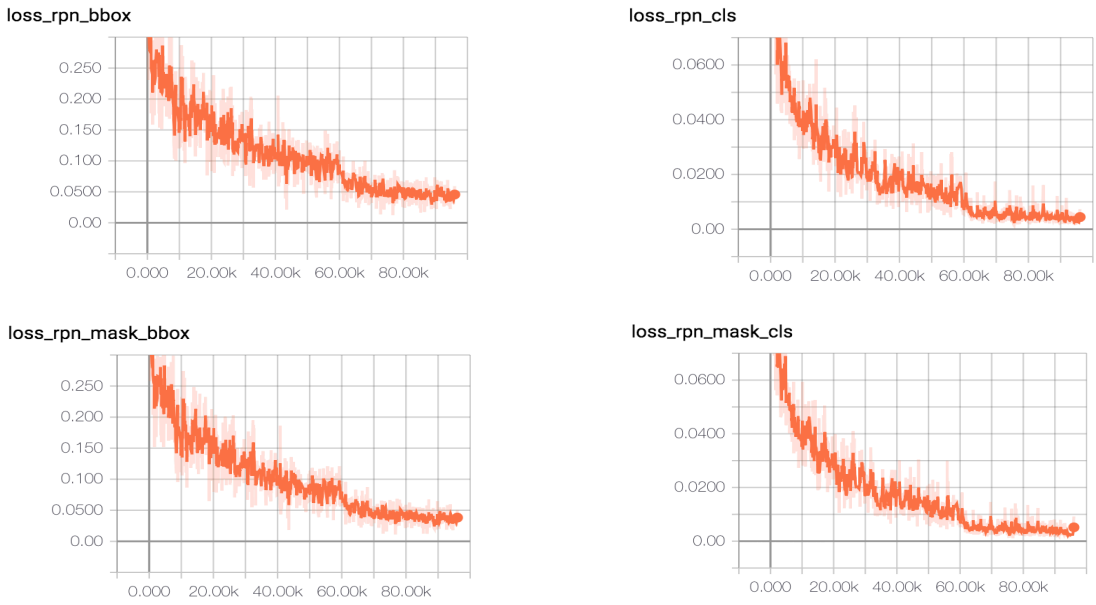


Figure 3.4: Loss curves of box regressor and class classifier corresponding to RPN_{det} (**top**) and RPN_{mask} (**bottom**).

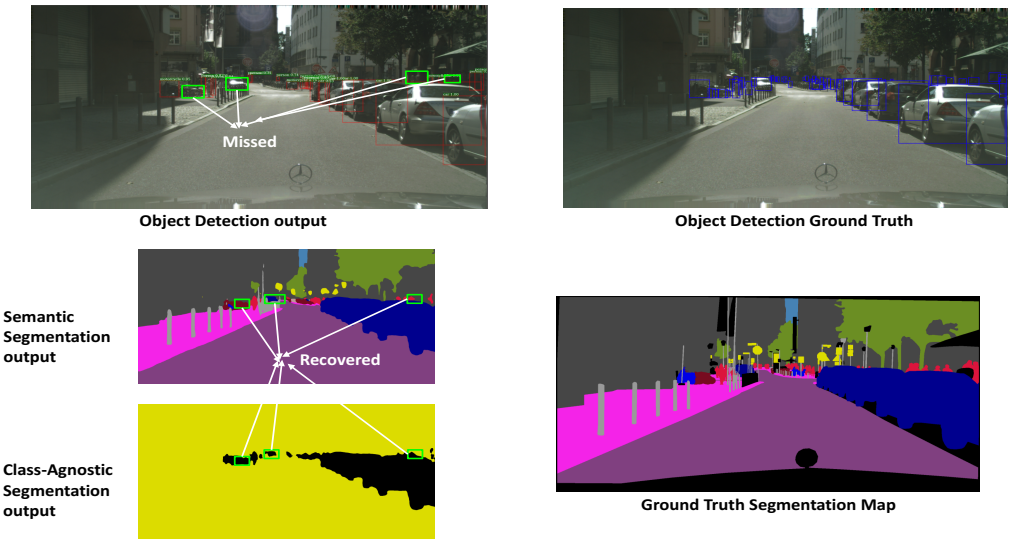
provides a scale specific objectness prior for semantic segmentation and (ii) it supports RPN_{mask} which improves the functioning of RPN_{det} in the detection branch.

Semantic segmentation is class-aware, trying to discriminate between all the semantic classes at once. However, it does not focus on the specific distinction between objects and the background. Inspired by [4], we add a class agnostic module in parallel to our semantic segmentation branch, both sharing the Resnet+FPN backbone. This module helps the semantic segmentation by giving an objectness prior. In addition, it also serves as a base for adding a RPN in the segmentation branch.

3.3.3 RPN in Class-Agnostic Segmentation Branch

Since the RPN feature maps resemble a class-agnostic segmentation map, a natural question to ask is can we train RPN on top of the segmentation feature maps? We explore this question by adding a RPN called RPN_{mask} on top of the feature maps in the class-agnostic segmentation branch. We show the training loss curves of the box regressor and binary classifier corresponding to both the RPN_{det} and RPN_{mask} in Figure 3.4. We observe that we're able to train the RPN_{mask} as well as RPN_{det}, further solidifying our

(a)



(b)

Figure 3.5: Motivation behind RPN_{mask}. A challenging image from the Cityscapes validation set is shown in (a). In (b), we visualize the outputs from different parts of our proposed network when RPN_{mask} is not attached. Blue and red boxes represent ground truth and predicted bounding boxes. Green boxes represent large objects which are missed by the object detection branch but recovered by the semantic and class-agnostic segmentation branches.

claim that the RPN feature maps resemble a class-agnostic segmentation map.

The motivation behind adding another RPN in the class-agnostic segmentation branch and exploiting the segmentation modality to improve object detection is shown in Figure 3.5. We show an example of a challenging image from the Cityscapes validation set in Figure 3.5(a) and its corresponding predictions from our network in Figure 3.5(b)

when RPN_{mask} is not attached. As can be gleaned from the object detection ground truth, the image contains a lot of objects, many of them being small in size. The small objects will naturally be tough to predict, but the network is still able to predict some of the small objects. However, it misses out on some objects which are reasonably large in size. As shown in Figure 3.3 and discussed in Section 3.3.1, the RPN feature maps indirectly control the proposals which are picked after filtering by the RPN, and these filtered proposals eventually get picked as predictions. We argue here that the reason why objects are being missed in Figure 3.5(b) is because the RPN feature maps do not contain segmented regions corresponding to these objects. On the other hand, we show that the class-agnostic segmentation output, which also resembles an RPN feature maps, does a better job at segmenting these missed objects. Interestingly, the semantic segmentation output also recovers these missed objects, further facilitating the evidence that the segmentation modality can help the task of object detection.

Since we're able to successfully train the RPN_{mask}, we use the RPN_{mask} feature maps to modify the RPN_{det} feature maps. This modification is based on the intuition that the RPN_{mask} feature maps learn some relevant information missed out by the RPN_{det} feature maps, due to RPN_{mask} exploiting the segmentation modality. This modification is basically an element-wise multiplication followed by an element-wise sum of the RPN feature maps corresponding to each FPN level, as shown in Figure 3.2. We also modify the ROI probability maps based on the same intuition, by element-wise addition of the ROI probability maps generated by RPN_{mask} and RPN_{det} for each of the three aspect ratios, right before the sigmoid layer.

3.4 Experiments

3.4.1 Datasets and Metric

Cityscapes

The Cityscapes dataset [12] is a collection of images with city street scenarios. It includes instance segmentation annotation which we transform into bounding boxes for our experiments. It contains 2,975 training images and 500 validation images.

Metric

For object detection, mean average precision (i.e., mAP) is used for performance evaluation. The mAP is calculated under the IoU threshold of 0.5. For semantic segmentation, mean intersection over union (i.e., mIoU) is used for performance evaluation.

3.4.2 Implementation Details

We use a standard FPN configuration with 256 output channels per scale. For the (pre-FPN) backbone, we use ResNet [28] models pre-trained on ImageNet [15] using batch norm (BN) [31]. All feature layers are jointly updated during training using Stochastic Gradient Descent (SGD). When used in fine-tuning, we replace BN with a fixed channel-wise affine transformation, as is typical [28]. The input images are rescaled to the size of 512×1024, and the size of mini-batch is 1. The total number of iteration in the training stage is 96k. We warm-start the learning rate from 0.0003 to 0.001 in 500 steps using linear annealing for stabilizing training, and then drop the learning rate by a factor of 10 at 60k and 80k iterations. We use a single Nvidia Titan XP, and implement the proposed method with Pytorch [51].

Table 3.1: Results of object detection (mAP) and semantic segmentation (mIoU) on Cityscapes. We also show class-agnostic segmentation results (mIoU*) as well as proposal average recall rate (Prop. AR) at 1000 proposals per image.

Model	mIoU	mIoU*	mAP	Prop. AR
Baseline	67.34	NA	50.38	79.58
Baseline + ROI	67.80	94.41	51.87	80.22
Baseline + ROI (w/o ROI during inference)	67.80	94.41	51.65	79.77
Baseline + RPN	67.95	94.40	51.45	80.40
Baseline + RPN (w/o RPN during inference)	67.95	94.40	50.85	79.40

3.4.3 Results on Cityscapes

We show the results of our proposed framework in Table 3.1. We first show the baseline results without our class-agnostic module. We note that the mIoU results are not comparable to the current state of the art [37, 77], primarily because of the training settings (batch size, no. of GPUs used, image resolution, batch normalization etc). But since our focus is to explore the connection between object detection and semantic segmentation, we keep our training settings simple and don't compare with these methods. Also, since Cityscapes is not primarily used for object detection, we cannot compare the object detection results with other methods.

By modifying the ROI probability maps, we're able to improve the mAP by 1.5%. We also see an increase in the mIoU by .5%. The increase in the semantic segmentation performance can be attributed to the class-agnostic branch, which gives an objectness prior for semantic segmentation. Note that the mIoU* performance is extremely high since it's a binary classification task (i.e., foreground vs background). We also see an increase in the proposal average recall rate, suggesting that more objects are being covered by the RPN generated proposals. We also show the results without modifying ROI probability maps during inference to show the effect of the modification. Similar to the ROI probability map modification, the RPN feature map modification also improves the mAP by 1.1%. It's accompanied by a similar improvement in the mIoU and equally good mIoU* performance, along with an increase in the proposal average recall rate.

3.4.4 Additional Experiments and Analysis

We also conduct ablation studies to further explore the proposed hypothesis of treating RPN feature maps as a dense class-agnostic segmentation map. Since our framework is built on top of FPN, the scale of objects is an important aspect of the overall approach. In the original FPN paper [42], the authors adapt a heuristic to assign an ROI of width w and height h to a certain level P_k of the pyramid, which is given by:

$$k = \lfloor k_0 + \log_2(\sqrt{wh}/224) \rfloor \tag{3.1}$$

Here 224 is the canonical ImageNet pre-training size and $k_0 = 4$, as mentioned in [42]. This heuristic essentially suggests that each level of the FPN is designed to handle a

certain scale of objects. Specifically,

$$k = \begin{cases} 5 & \sqrt{wh} \geq 448 \\ 4 & 448 > \sqrt{wh} \geq 224 \\ 3 & 224 > \sqrt{wh} \geq 112 \\ 2 & \text{otherwise} \end{cases} \tag{3.2}$$

In this work, since we rely on the class-agnostic segmentation modality to improve object detection, we must divide the segmentation information based on scale in order for it to be applied to the FPN. We specifically use the ground truth class-agnostic segmentation map for this ablation study in order to verify the idea. Also, note that the canonical pre-training size (i.e., 224) is not really applicable for different input image sizes. So, we experiment with the above heuristic in three ways which we refer to as: i) Case 1 ii) Case 2, and iii) Case 3. The core idea behind these heuristics is to find a scale appropriate matching between the ground truth segmentation map and ROI assignment. The Cityscapes dataset has instance segmentation ground truth, which we use to find the scale of individual objects. We visualize these hueristics in Figure 3.7 and describe them in detail below.

Case 1

For this case, we use the same heuristic for ROI assignment as used in [42]. For dividing the ground truth segmentation map, we follow a different strategy. Each level of the FPN is designed to handle a certain scale of object, which is specified by the fixed anchor size used in the RPN. So we use these anchor sizes as a heuristic for dividing the segmentation map based on object/instance sizes. Specifically, let the FPN level to

Table 3.2: Results of additional experiments on Cityscapes. We refer to the three cases by their corresponding subscript. We show object detection (mAP) and semantic segmentation (mIoU) results as well as proposal average recall rate (Prop. AR) at 1000 proposals per image.

Model	mIoU	mAP	Prop. AR
Baseline	67.34	50.38	79.58
Baseline + ROI_1	67.89	51.42	80.56
Baseline + ROI_1 (w/o ROI_1 during inference)	67.89	51.25	79.95
Baseline + RPN_1	68.52	51.57	82.59
Baseline + RPN_1 (w/o RPN_1 during inference)	67.95	50.84	78.86
Baseline + ROI_1 + RPN_1	68.18	51.40	83.34
Baseline + ROI_1 + RPN_1 (w/o ROI_1 and w/o RPN_1)	68.18	49.93	79.02
Baseline + ROI_1 + RPN_1 (w/o ROI_1)	68.18	51.20	82.72
Baseline + ROI_1 + RPN_1 (w/o RPN_1)	68.18	50.30	79.90
Baseline + ROI_2	68.33	51.50	80.70
Baseline + ROI_2 (w/o ROI_2 during inference)	68.33	51.28	79.84
Baseline + RPN_2	68.23	51.71	82.54
Baseline + RPN_2 (w/o RPN_2 during inference)	68.23	50.17	79.84
Baseline + ROI_3	68.24	51.21	80.55
Baseline + ROI_3 (w/o ROI_3 during inference)	68.24	51.13	80.29
Baseline + RPN_3	68.43	51.56	82.84
Baseline + RPN_3 (w/o RPN_3 during inference)	68.43	50.43	79.64

which the segmentation map will be divided and assigned to be represented by L. Then,

$$L = \begin{cases} 6 & \sqrt{a} \geq 256 \\ 5 & 256 > \sqrt{a} \geq 128 \\ 4 & 128 > \sqrt{a} \geq 64 \\ 3 & 64 > \sqrt{a} \geq 32 \\ 2 & \text{otherwise} \end{cases} \tag{3.3}$$

where a is the area of the bounding box of an object. Note that a is calculated keeping in mind the spatial stride of the network. The above heuristic essentially maps an object in the ground truth segmentation map to a level in the FPN. We visualize this heuristic in Figure 3.7a.

Case 2

In the previous case, there was a difference in the heuristic used for ROI assignment [42] and the one we used for dividing the ground truth segmentation map. In order for them to be in sync, we keep the original heuristic for ROI assignment, but change our heuristic used for dividing the segmentation map. Specifically,

$$L = \begin{cases} 6,5 & \sqrt{a} \geq 448 \\ 4 & 448 > \sqrt{a} \geq 224 \\ 3 & 224 > \sqrt{a} \geq 112 \\ 2 & \text{otherwise} \end{cases} \tag{3.4}$$

We visualize this heuristic in Figure 3.7b.

Case 3

As mentioned earlier, the canonical pre-training size (i.e., 224) used in [42] is not really applicable for different input image size. This is because the canonical pre-training size affects the ROI assignment to the FPN levels, and the ROI sizes will be different for different input image size. Since we use an input size of $512x1024$, we empirically change the canonical pre-training size to 128 so that each level of the FPN gets some objects to handle. This can be visually seen in Figure 3.7c. Specifically,

$$L = \begin{cases} 6 & \sqrt{a} \geq 384 \\ 5 & 384 > \sqrt{a} \geq 192 \\ 4 & 192 > \sqrt{a} \geq 96 \\ 3 & 96 > \sqrt{a} \geq 48 \\ 2 & \text{otherwise} \end{cases} \tag{3.5}$$

Results

We show the results of the additional experiments in Table 3.2. We also show effect of the modifications on the foreground ROIs generated by RPN after filtering by NMS

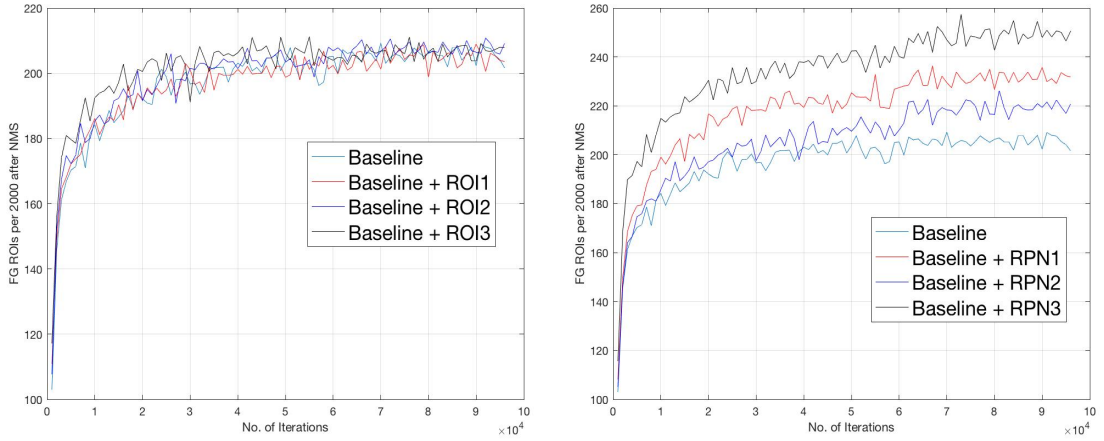


Figure 3.6: Plot of Foreground ROIs per 2000 after filtering by NMS for all three heuristic cases over no. of iterations

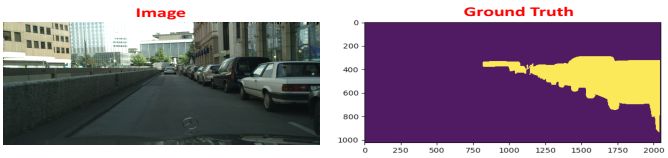
in Figure 3.6. For Case 1, we observe that by modifying the ROI probability maps, we're able to improve the mAP by 1.1%. We also see an improvement in the semantic segmentation performance by .5%, which can be attributed to the improvement in the common FPN backbone features shared by the object detection and semantic segmentation branch. The improvement is also reflected in the proposal average recall rate, which suggests more objects being covered by the RPN after modification. We see a similar trend when modifying the RPN feature maps, with an improvement of 1.2% for both mIoU and mAP. We also see an improvement in the proposal average recall rate by 3%. This improvement in the proposal average recall rate is also reflected in Figure 3.6 (**right**), where the FG ROIs generated are higher after modification when compared with baseline. This however, is not very clearly reflected for the case of ROI modifications. In addition, we also provide results without incorporating the modifications at test time, to show the effect of the proposed method.

For Case 2 and Case 3, we observe a similar trend as the previous case. By modifying the ROI probability maps, we're able to improve the mAP and and also the mIoU, which can again be attributed to the improvement in the common FPN backbone features shared by the object detection and semantic segmentation branch. The improvement is also reflected in the proposal average recall rate, which suggests more objects being covered by the RPN after modification. Figure 3.6 also shows the increasing trend in the

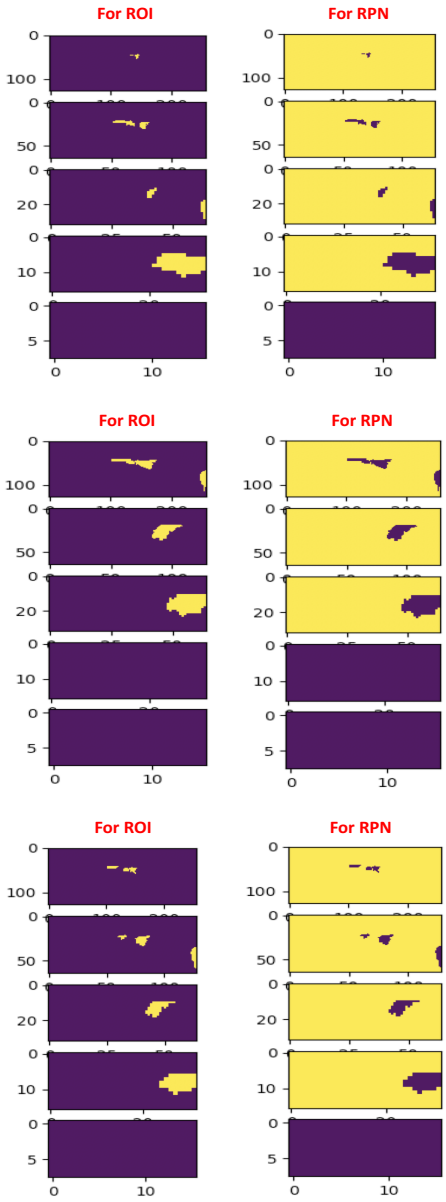
FG ROIs for subsequent cases, which is also reflected in the higher values of proposal average recall rate. We see a similar trend when modifying the RPN feature maps, with an improvement for both mIoU and mAP. In addition, we also provide results without incorporating Z at test time, to show the effect of the proposed method.

3.5 Summary

In this chapter, we propose an end-to-end learning framework called SegRPN for joint object detection and semantic segmentation. Our framework is based on the novel observation that the RPN feature maps in Faster-RCNN resemble a dense class-agnostic segmentation map. We treat this observation as a deeper connection between the tasks of object detection and semantic segmentation, and build a framework to exploit this observation. Experimental results on Cityscapes show the effectiveness of the proposed method. We also provide additional experiments and analysis to further explore this observation.



(a)



(b)

Figure 3.7: Visualization of the heuristic cases. In (a) we show an example image (**left**) from Cityscapes validation set its class-agnostic segmentation ground truth (**right**). We visualize the segmentation map division for all 5 FPN levels (2-6) in (b) corresponding to heuristic 1 (**top**), heuristic 2 (**middle**) and heuristic 3 (**bottom**) for both ROI and RPN modifications.

Chapter 4

Conclusion and Future Work

In this thesis, we focus on the recognition of objects and actions, which are the building blocks of any vision based machine intelligence. Firstly in Chapter 2, we propose an end-to-end learning adaptive framework for spatio-temporal action localization. We show that in order to effectively adapt the model, both spatial and temporal features need to be adapted. Our experimental results show the effectiveness of the proposed approach and the effect of adapting different components of the framework. In order to show potential benefits of the proposed approach for autonomous cars, we also introduce and benchmark a new dataset we collected of pedestrian actions in driving scenes.

Next in Chapter 3, we focus on the task of joint object detection and semantic segmentation. We explore a novel observation that the RPN feature maps in Faster-RCNN resemble a dense class-agnostic segmentation map. We treat this observation as a deeper connection between the two tasks. We propose an end-to-end learning framework which, in addition to sharing backbone convolutional features, also exploits this said connection. Our experimental results show that the proposed approach is able to improve both object detection and semantic segmentation performance. We also provide additional experiments and analysis to support our claimed observation.

In the end, we conclude by discussing the potential directions for future research. As pointed out earlier, objects and actions are the building blocks of any vision based intelligent system. In order for this intelligent system to be truly effective, it should be able to reason about both objects and actions simultaneously. For example, an autonomous car should be able to holistically reason about a given scene. This holistic understanding

should include detecting and segmenting other cars and pedestrians down to the individual pixel level, which would constitute as instance segmentation. This can be done by extending the framework in Chapter 3. The system should also be able to simultaneously classify the actions of these cars and pedestrians in order for effective automated navigation. One possible way to achieve this is to merge the frameworks proposed in Chapters 2 and 3, which can be done as both frameworks are based on the Faster R-CNN algorithm. In order to evaluate such a framework, an annotated dataset would also be required. This dataset would require pixel level as well as spatio-temporal action annotations. This can be achieved by extending the dataset we introduced in Chapter 2 to contain pixel level annotations as well as actions of other cars, which although helpful, would require additonal resources such as time and money.

Bibliography

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[2] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain Separation Networks. In *Advances in Neural Information Processing Systems*, 2016.

[3] G. Brazil, X. Yin, and X. Liu. Illuminating pedestrians via simultaneous detection & segmentation. In *IEEE International Conference on Computer Vision*, pages 4950–4959, 2017.

[4] J. Cao, Y. Pang, and X. Li. Triply supervised decoder networks for joint detection and segmentation. *arXiv preprint arXiv:1809.09299*, 2018.

[5] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[6] J. Carreira and A. Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[7] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[8] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[9] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision*, pages 801–818, 2018.

[10] M.-H. Chen, Z. Kira, and G. AlRegib. Temporal Attentive Alignment for Video Domain Adaptation. *arXiv preprint arXiv:1905.10861*, 2019.

[11] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool. Domain Adaptive Faster R-CNN for Object Detection in the Wild. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[12] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016.

[13] G. Csurka. Domain Adaptation for Visual Applications: A Comprehensive Survey. In *Domain Adaptation in Computer Vision Applications*, pages 1–35. Springer, 2017.

[14] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling Egocentric Vision: The EPIC-KITCHENS Dataset. In *European Conference on Computer Vision*, 2018.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[16] N. Dvornik, K. Shmelkov, J. Mairal, and C. Schmid. Blitznet: A real-time deep network for scene understanding. In *IEEE International Conference on Computer Vision*, pages 4154–4162, 2017.

[17] B. G. Fabian Caba Heilbron, Victor Escorcia and J. C. Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[18] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[19] Y. Ganin and V. Lempitsky. Unsupervised Domain Adaptation by Backpropagation. In *International Conference on Machine Learning*, 2015.

[20] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial Training of Neural Networks. *Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[21] R. Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision*, 2015.

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

[23] G. Gkioxari and J. Malik. Finding Action Tubes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[24] C. Gu, C. Sun, D. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, C. Schmid, and J. Malik. AVA: A Video Dataset of Spatio-temporally Localized Atomic Visual Actions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[25] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative Domain Adaptation. In *IEEE International Conference on Computer Vision*, 2017.

[26] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[29] R. Hou, C. Chen, and M. Shah. Tube Convolutional Neural Network (T-CNN) for Action Detection in Videos. In *IEEE International Conference on Computer Vision*, 2017.

[30] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[31] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[32] A. Jamal, V. P. Namboodiri, D. Deodhare, and K. Venkatesh. Deep Domain Adaptation in Action Space. In *British Machine Vision Conference*, 2018.

[33] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards Understanding Action Recognition. In *IEEE International Conference on Computer Vision*, 2013.

[34] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Action Tubelet Detector for Spatio-Temporal Action Localization. In *IEEE International Conference on Computer Vision*, 2017.

[35] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The Kinetics Human Action Video Dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[36] Y. Ke, R. Sukthankar, M. Hebert, et al. Efficient Visual Event Detection Using Volumetric Features. In *IEEE International Conference on Computer Vision*, 2005.

[37] A. Kirillov, R. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6399–6408, 2019.

[38] I. Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6129–6138, 2017.

[39] J. F. P. Kooij, N. Schneider, F. Flohr, and D. M. Gavrila. Context-based Pedestrian Path Prediction. In *European Conference on Computer Vision*, 2014.

[40] C. Lin, J. Lu, G. Wang, and J. Zhou. Graininess-aware deep feature learning for pedestrian detection. In *European Conference on Computer Vision*, pages 732–747, 2018.

[41] G. Lin, A. Milan, C. Shen, and I. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1925–1934, 2017.

[42] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[43] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. In *IEEE International Conference on Computer Vision*, 2017.

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

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

[46] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning Transferable Features with Deep Adaptation Networks. In *International Conference on Machine Learning*, 2015.

[47] I. Loshchilov and F. Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. *International Conference on Learning Representations*, 2017.

[48] J. Mao, T. Xiao, Y. Jiang, and Z. Cao. What can help pedestrian detection? In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3127–3136, 2017.

[49] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified Deep Supervised Domain Adaptation and Generalization. In *IEEE International Conference on Computer Vision*, 2017.

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

[51] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. 2017.

[52] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2017.

[53] X. Peng and C. Schmid. Multi-region Two-stream R-CNN for Action Detection. In *European Conference on Computer Vision*, 2016.

[54] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *European Conference on Computer Vision*, pages 75–91, 2016.

[55] A. Rasouli, I. Kotseruba, and J. K. Tsotsos. Are They Going to Cross? A Benchmark Dataset and Baseline for Pedestrian Crosswalk Behavior. In *IEEE International Conference on Computer Vision*, 2017.

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

[57] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7263–7271, 2017.

[58] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems*, 2015.

[59] M. D. Rodriguez, J. Ahmed, and M. Shah. Action MACH a Spatio-temporal Maximum Average Correlation Height Filter for Action Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[60] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[61] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting Visual Category Models to New Domains. In *European Conference on Computer Vision*, 2010.

[62] S. Saha, G. Singh, M. Sapienza, P. H. Torr, and F. Cuzzolin. Deep Learning for Detecting Multiple Space-Time Action Tubes in Videoss. *arXiv preprint arXiv:1608.01529*, 2016.

[63] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Strong-Weak Distribution Alignment for Adaptive Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[64] S. Sankaranarayanan, Y. Balaji, A. Jain, S. Nam Lim, and R. Chellappa. Learning from Synthetic Data: Addressing Domain Shift for Semantic Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[65] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning Transferrable Representations for Unsupervised Domain Adaptation. In *Advances in Neural Information Processing Systems*, 2016.

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

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

[68] G. Singh, S. Saha, M. Sapienza, P. H. Torr, and F. Cuzzolin. Online Real-time Multiple Spatiotemporal Action Localisation and Prediction. In *IEEE International Conference on Computer Vision*, 2017.

[69] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild. *arXiv preprint arXiv:1212.0402*, 2012.

[70] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *IEEE Intelligent Vehicles Symposium*, pages 1013–1020, 2018.

[71] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. In *IEEE International Conference on Computer Vision*, 2015.

[72] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial Discriminative Domain Adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[73] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep Domain Confusion: Maximizing for Domain Invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[74] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell. Understanding convolution for semantic segmentation. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1451–1460, 2018.

[75] P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Learning to Track for Spatio-temporal Action Localization. In *IEEE International Conference on Computer Vision*, 2015.

[76] P. Weinzaepfel, X. Martin, and C. Schmid. Towards Weakly-supervised Action Localization. *arXiv preprint arXiv:1605.05197*, 2, 2016.

[77] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun. Upsnet: A unified panoptic segmentation network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8818–8826, 2019.

[78] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang. Denseaspp for semantic segmentation in street scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3684–3692, 2018.

[79] J. Yao, S. Fidler, and R. Urtasun. Describing the scene as a whole: joint object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[80] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Learning a discriminative feature network for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1857–1866, 2018.

[81] J. Yuan, Z. Liu, and Y. Wu. Discriminative Subvolume Search for Efficient Action Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[82] Z. Zhang, S. Qiao, C. Xie, W. Shen, B. Wang, and A. L. Yuille. Single-shot object detection with enriched semantics. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5813–5821, 2018.

[83] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2881–2890, 2017.

[84] Y. Zou, Z. Yu, B. Vijaya Kumar, and J. Wang. Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training. In *European Conference on Computer Vision*, 2018.