Micro-Domain Adaptation on Long-Running Videos

Victor Sun

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2020-91
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-91.html

May 29, 2020

Micro-Domain Adaptation on Long-Running Videos

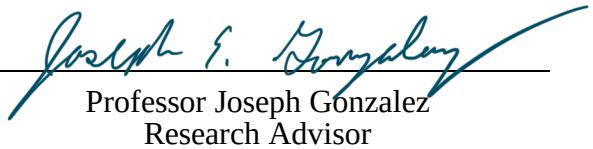
by Victor Sun

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

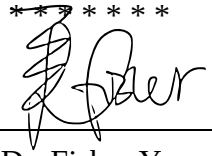


Professor Joseph Gonzalez
Research Advisor

May 28, 2020

(Date)

* * * * * *



Dr. Fisher Yu
Second Reader

05/24/2020

(Date)

Micro-Domain Adaptation on Long-Running Videos

Victor Sun
Collaborators: Xin Wang, Thomas Huang, Xiaolong Wang, Fisher Yu
Advisor: Joseph Gonzalez

Abstract

Micro-Domain Adaptation on Long-Running Videos

by

Victor Sun
Collaborators: Xin Wang, Thomas Huang, Xiaolong Wang, Fisher Yu
Advisor: Joseph Gonzalez
Masters of Science in EECS

University of California, Berkeley

Domain adaptation techniques are often used to fine-tune models and improve performance when the distribution of the test data differs from the distribution of the training data. Various domain adaptation techniques and datasets for object detection exist for images and short video sequences of length, but this has been a lot less widely studied in long-running videos of an hour or more. With longer videos there can be a significant domain gap between the beginning of the video and the end of the video, and there are currently no datasets that allow us to evaluate this. We aim to provide a diverse test dataset of long-running videos with noticeable domain shifts within the video to study micro-domain adaptation over a long sequence. These videos are taken from a variety of YouTube live-cam videos in cities around the world and contain labeled frames. We also discuss some potential self-supervised and semi-supervised online learning approaches to deal with concept drift in object-detection on longer-running videos.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Often time, models trained on a source dataset may not always generalize and perform well on a target test dataset, especially if the test samples are out-of-distribution with the training samples, in a phenomenon known as domain shift. To tackle this challenge, there have been a lot of recent works on domain adaptation in both unsupervised and semi-supervised settings. A lot of the approaches for object detection and domain adaptation are predicated on having access to large-scale datasets, such as COCO [8] and ImageNet [4] for image-based object detection and BDD100k [17], Cityscapes [2], and YT-BB [10] for video-based object detection.

Amongst object detection for video, many of the datasets have diverse video scenes and have resulted in approaches that have been fairly successful at macro-level domain adaptation. However, the video sequences are quite short and may experience worse performance as the video progresses, especially with few-shot approaches. Some of the longest video sequences are from BDD100K and even then on average only contains 40 second video sequences. [17] In general, most video datasets have sequences that last only a few seconds and this is not enough time for there to be notable shifts in micro-domain distribution over the course of the video. Domain adaptation on long-running videos of an hour or longer have not been studied much, but can provide interesting scenarios not captured through short video scenes such as concept drift and slight shifts in domain throughout the video as well. However, this is difficult to do as there are no existing datasets of long-running videos running hours or longer. As such, we aim to create a dataset of long-running videos that allows us to tackle these potential domain gaps from the beginning to the end of a long video sequence. Since we are studying the gradual change in domain over time, we will call this problem micro-domain adaptation.

This is important to examine because evaluating a trained model in the real-world, it may not be realistic to fine-tune the model for specific scenes ahead of time and we may need to constantly be updating the model over a longer period of time as a form of online continuous adaptation. For example, in the case of autonomous driving in a closed-loop simulation once the car begins driving, we may not be able to do per-sequence fine-tuning and few-shot approaches may not be updated with changes in the micro-domain shifts over

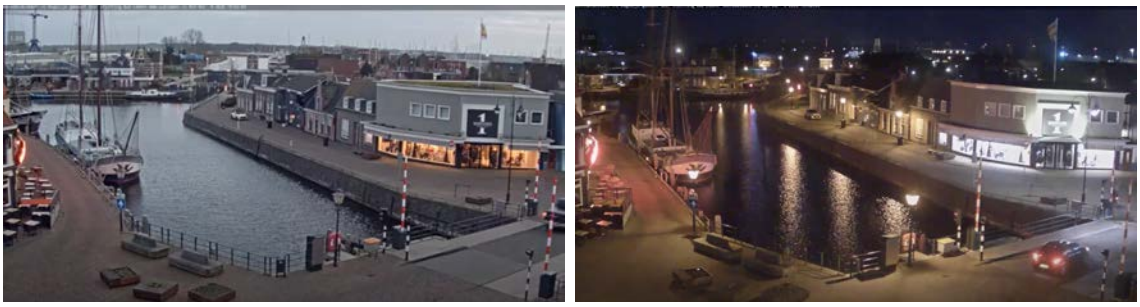


Figure 1.1: Clear micro-domain gap between beginning (left) and end (right) of sequence from a sample video in our dataset.

a longer period of time, so we will need a method to continuously update our model.

Therefore, in order to better understand the effects of domain shift on longer videos, we offer 3 main contributions:

1. **Dataset on Long-Running Videos:** We offer a collection of 120 hour-long videos, 8 of which are labeled, with dramatic domain shifts within the hour-long scene. Potential domain shifts can include changes from afternoon to evening, dawn to morning, sunny to rainy, or crowded to sparse. We provide high-quality human-annotated labels on sequences using an exponential interval starting from dense to sparse, *e.g.* 10 labeled frames in the 1st second, 10 labeled frames in the next 2 seconds, 10 labeled frames in the next 4 second, and so on with 10 labeled frames in the next 2^n seconds where n is the interval index. To our knowledge, this dataset contains some of the longest-running sequences of any video dataset. However, more importantly it is long enough to notice visible domain shifts within a sequence as demonstrated in Fig. 1.

2. **Benchmarking on Dataset:** We analyze the performance of pre-trained models on our dataset to provide a baseline and show the effects of the micro-domain shifts on performance. We also compare our dataset to other datasets and show the shortcomings of off-the-shelf models without domain adaptation and we why we may need online learning methods for effective micro-domain adaptation.

3. **Potential Online Learning Techniques:** We discuss the potential use of self-supervised and semi-supervised approaches for online continuous domain adaptation that updates the model as new frames in the video are evaluated. Approaches include using rotational supervision task [5] [13], jigsaw task [9], measuring cycle-consistency loss [15], or bootstrapping with labeled samples when available. [12]

Chapter 2

Background and Related Works

2.1 Domain Adaptation

Domain Adaptation Overview

One of the key things to understand is what exactly domain adaptation (DA) and transfer learning (TL) are, specifically in the context of visual applications. In his survey , Csurka [3],claims transfer learning into 3 main types inductive TL where the target task and source task differ, transductive TL where the source and target tasks are the same but the data representation or data distribution differ, and unsupervised TL where neither the task nor data domain match and the goal is to learn some unsupervised representation from the source that can be applied to the target. With domain adaptation, we are looking mostly at transductive TL, focused on a particular task, in our case object detection, but with different data distribution. This can be studied in both and unsupervised (where labels are only available in source dataset) as well as self-supervised setting (where some labels are available in target dataset).

Domain adaptation for image classification problems has been fairly well explored. Some traditional approaches that can be used in domain adaptation include feature augmentation, feature space alignment and transformation where PCA dimensions can be selected to minimize diverge between source and target data, and re-weighting instances based off likelihood of being a source or target example via Maximum Mean Discrepancy. Some more modern deep learning approaches include fine-tuning CNNs which require target labeled data, Generative Adversarial Networks which can be used to learn a joint distribution of images across domains, and encoder-decoder models which can learn from both supervised labels as well as unsupervised data reconstruction as an auxiliary task. [3] In our work, we definitely hope to leverage a joint model using both supervised and unsupervised tasks.

Self-Training and Self-Supervised Approaches

Object detection approaches have proved to be slightly more difficult. In Chowdhury *et al.* [11] self-training approaches have been used which start with high-confidence predictions from detectors trained on source data. They then use temporal cues from object trackers to aid with labeling difficult examples that can then be used to re-train the original model. Finally, a knowledge distillation loss is used to counteract noise with pseudo-labeled data. Knowledge distillation is a technique used to compress the knowledge and representative power of ensembles of models into a single smaller networks through transfer datasets. [7]

Self-supervised approaches for domain adaptation have also been discussed as in Xu *et al.* [16] In their approach, the model is broken down into an encoder network which serves as a general feature extractor and a decoder network using the output of encoder network to solve some specific supervised task. However, by adding various self-supervised pre-text tasks such as rotation task, that also use the same shared encoder network, they are able to update the encoder network/feature extractor with the self-supervised tasks as well, which could ultimately affect the performance of the decoder network.

Some additional techniques they use to help with self-supervision include adversarial training to help align the supervised decoder with the feature-level encoder. This is done by placing a discriminator at the output of the encoder to predict whether feature map is from source or target, with the goal of fooling the domain discriminator such that it cannot distinguish data from source or target. Another technique involves batch normalization where the batch is normalized by its mean and variance to reduce domain shift between datasets.

These self-supervised approaches are further discussed that by Sun *et al.* [14] who mentions a variety of other pretext tasks such as solving jigsaw puzzles, colorization, in-painting, and more. While traditionally self-supervised tasks were used to pre-train on unlabeled data before training with supervised data, these methods train the self-supervised tasks together with the main downstream task. This is then treated as a multi-task learning problem with multiple loss functions of downstream task on source dataset and pretext self-supervised tasks on target dataset.

While all these domain adaptation approaches work well for image datasets across various tasks such as segmentation and object detection, for video this has been quite under-explored, partly due to the difficulties in incorporating temporal information, and lack of datasets for transfer learning and domain adaptation. In our work, we hope to leverage these techniques but refine them for use-cases in video domain adaptation.

Video Domain Adapatation

One recent work that did target video domain adaptation, Chen *et al.* uses networks which attend to temporal dynamics to help bridge the gap between images and video. [1] They propose a temporal attentive adversarial network which aims to use temporal dynamics to perform domain alignment. However, in the dataset they construct and evaluate on, no

sequence is longer than 33 seconds and so they may not be able to capture some of the micro-domain shifts that occur within a video sequence. In addition, they do not take advantage of any self-supervised learning or online learning methods.

Another self-supervised approach that could potentially be applied for video domain adaptation use cases is the cycle-consistency task pitched in Wang *et al.* [15]. In this paper, visual correspondence is captured between frames from unlabeled video and is used as a supervisory signal for learning visual representations and feature maps, which can serve to train an encoder which could then be used in the downstream task.

Online Learning

Finally, online learning or test-time training methods have becoming increasingly popular for domain adaptation and could potentially be quite useful in the video setting. Sun *et al.* [13] discusses how unlabeled test instances can be used in a self-supervised setting to update models to generalize for out-of-distribution test sets. Specifically, the loss is trained jointly as in multi-task learning settings during training-time and the shared encoders or feature extractors are updated and fine-tuned based off auxiliary task loss at test-time

While none of these works incorporate long-running videos, this approach could analogously be applied to changes throughout the course of a video to help adapt to changes in setting and domain throughout the video. In our work, we hope to apply some traditional techniques in domain adaptation for images and video as well as some online learning techniques to tackle domain adaptation within long sequences. In addition, we hope to provide just a few labeled examples periodically in our video stream which can be used to update the model at test-time via model distillation or few-shot learning techniques in a semi-supervised fashion [12]

2.2 Datasets

With the growing interest for autonomous vehicles research, video-based object detection and segmentation datasets have become increasingly more available. Some examples include CityScapes [2] and BDD100K [17], both video datasets taken from driving cars in a variety of scenes. BDD100k was built to study heterogeneous multitask learning and contains labels for a variety of tasks including object detection, but also semantic segmentation, object tracking, and lane marking. It also includes image tags for weather, scene type, and time of day that can help with domain adaptation tasks. Cityscapes is another urban driving dataset containing object detection labels for videos and consists of a mix of both high quality pixel-level annotations and course annotations. Besides these two datasets, YT-BB [10] contains annotated videos from YouTube which can also be used as a baseline dataset for more everyday tasks containing labels from the MS COCO label set [8], a popular object detection dataset for still images.

Despite the increasing number of object detection video datasets, there are still very few datasets targeted at long videos and domain adaptation. Many of the previously mentioned datasets are quite large in scale with BDD100k containing 100k examples and YT-BB containing 380k video sequences, they all contain very short sequences with BDD100k containing 40 second clips and YT-BB containing 19 second clips on average. Due to their large scale, these video datasets serve as a good source dataset to work off of, however they do not contain long enough videos to target domain shifts over the course of a long video sequence.

In addition, while some of these datasets contain a wide diversity of videos, *i.e.* BDD100k contains day-time and night-time datasets and can serve as a good a baseline for out-of-distribution data, they are not specialized for domain adaptation. The setting within the dataset is still quite similar in being recorded from a car's windshield dash cam. This creates the need for a differing dataset which contains scenes beyond just driving and may contain cameras at various different angles and zoom, in addition to setting, weather, and lighting changes. While YT-BB may contain a bunch of everyday videos they are not explicitly organized like in BDD100k and as such, also are not good as a domain-adaptation focused dataset. However, because these datasets can serve as a good source dataset, we will also be using BDD100k as our dataset for training a baseline model, which we will then fine-tune for our target domain adaptation =dataset.

Two video datasets that are specialized toward video domain adaptation include those created by Chen *et al.* [1] in their work on large-scale video domain, specifically UCF-HMDB and Kinetics Gameplay. UCF-HMDB is extracted from over 3000 real-world video clips whereas Kinetics Gameplay contains scenes captured from virtual worlds in gameplay videos. In their work, they claim that their dataset was the largest dataset targeted for video domain adaptation, but even so their longest video were 33 seconds and would not be good for studying the effects of longer videos. It becomes evident there are no good datasets that are both focused on both domain adaptation and contain long-running videos. As such, we ultimately propose our own target test dataset for evaluating domain adaptation on long video sequences.

Chapter 3

Dataset Construction

3.1 Dataset Overview

Our main motivation for creating this dataset was to provide a diverse set of long-running videos with significant scene change in them to help better understand and evaluate performance for scene and domain shifts within a video. The videos were obtained from a series of web-cam surveillance videos being live streamed on YouTube and downloaded via Youtube-DL. They cover a wide range of scenes from public plazas and narrow streets to parks and ski resorts across a wide variety of regions in Europe, Asia, and the Americas. We also account for a variety of weather from sunny to snowy, and varying density of activity.

3.2 Dataset Screening Process

The videos were then screened thoroughly for several qualities including notable lighting and domain shifts within the video, static non-panning cameras, high resolution at 720p or higher, and quality of scene (with regards to how close up and visible). In addition, faces were blurred to address privacy concerns. The reason we wanted to work with static non-moving camera frames is that the the frames would have more limited views than moving cameras and would be more domain-specific. In addition, since we intended to evaluate this dataset on a model trained on BDD100k [17] which is derived from cameras on moving cars, the still cameras would provide a domain gap from the training data. For uniformity sake and for our evaluation to be fair throughout we needed to stick to either all-moving cameras or all static cameras and since our goal was to create a dataset geared toward domain adaptation, still cameras made the most sense.

Originally, we had downloaded 24 hour videos and wanted to work with sequences around this length to realize the effects of domain shift over time. However, we found that often times at night or specific times of day, there would be uninteresting or no events. Using the full 24 hours would require a lot more storage space for the dataset and would be a lot more expensive to label, especially if the videos contained a lot of empty night scenes. As a result,

we chose the best hour-long sequence within each 24 hour time frame that we downloaded with the most interesting visual changes to work with. We found 1 hour sequences to be adequate enough to display noticeable visual changes in the domain setting. Sample video sequence with lighting changes measured in difference in L-channel is shown in Fig. 3.3

3.3 Dataset Labeling

Overall, the dataset contains 120 hour-long video sequences, 8 of which has been labeled so far in the first iteration. We capture the frames from the video at 10 FPS, resulting in a total of 36,000 frames per video and 4.32 million frames in total. Given our limitations in budget, it would be infeasible to label all 36,000 frames captured in each sequence. As such the frame labeling follows an exponential interval where the first 10 frames are evenly labeled over the first second, the next 10 labeled frames spread over the following two seconds, following 10 labeled frames in following four seconds, etc. In other words, the gap between labeled frames doubles every 10 frames and labeling goes from dense to sparse. In total, we have 117 labeled frames per hour-long sequence.

The motivation for this labeling scheme is to cover all parts of the sequence but still be able to bootstrap the earlier frames with dense enough labels to learn temporal changes within the sequence via online learning. Several, other candidate labeling schemes included labeling sparsely throughout the video, but this would not allow us to see how well the model uses tracking and temporal information since adjacent frames are not being evaluated. Another approach was to have the first 5 minutes be labeled densely at a flat interval of 10 FPS and then all other frames sparely at another interval of say 1 frame per minute. While this approach is similar to the exponential interval going from dense to sparse, it has a sudden drop-off in labeling and still requires more labels without provide significantly greater benefit than the exponential interval labeling schedule.

The data annotation process was carried out via third party human annotators, using the Scalabel platform which provides an interface for high quality data annotating and labeling of 2D bounding boxes. The classes we use for labeling this dataset include the 8 of the road object classes used in BDD100k: bike, bus, car, motorcycle, person, rider, train, truck. In total, for the 8 labeled video sequences we have 31,601 labeled objects, the distribution of classes for two sample videos is shown in Fig 3.3. Throughout this process, we were ultimately bottle-necked by the human annotating of labels by our professional annotators, but hope to label many more of the sequences in the future.

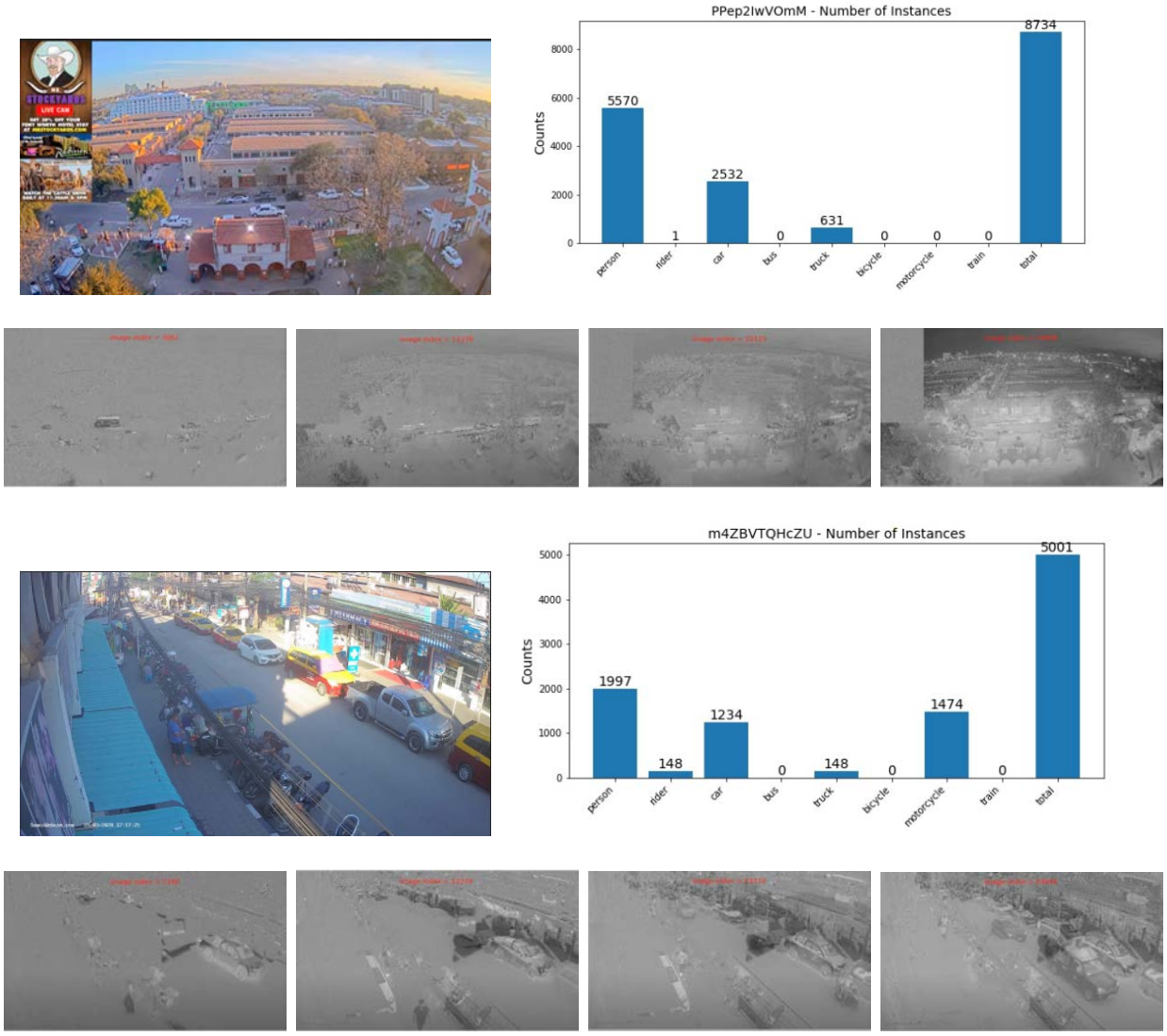


Figure 3.1: Above is the first frame of two sample-sequence (top left), the distribution of labeled bounding boxes within the sequence (top right), and the brightness over the course of the sequence (bottom) measured by difference in L-channel with respect to the starting frame using the CIELAB color space.

Chapter 4

Dataset Benchmark

4.1 Comparison to Other Datasets

We compare our Micro-Domain Adaptation Dataset (MDA) to other state-of-the-art video object detection datasets, specifically looking at length of scene and indirectly also measuring change from beginning to end of scene. As shown in Table 4.1 our sequences are significantly longer than those in other datasets such as BDD-100k, YT-BB, and UCF-HMDB. [17] [10] [1]. Obviously, we are unable to achieve the shear number of sequences and labeled data but since our dataset is purely a testing set for domain adaptation, having a large number of labeled bounding boxes is not as important for us. Furthermore, we have the option to download and label more sequences in the future iterations.

Table 4.1: A comparison of our micro domain adaptation (MDA) dataset to other video object detection dataset with known sequence lengths. We are offering a test set and so our dataset contains fewer sequences but much longer average sequence duration. Number of bounding box labels for UCF-HMDB is not known to us.

Dataset	Avg Duration (s)	Sequences	Bounding Boxes
MDA (ours)	3600	120	34k
BDD-100k	40	100k	3.3M
YT-BB	19	380k	5.5M
UCF-HMDB	33	3209	-

4.2 Performance on Dataset

In order to evaluate the need for domain adaptation in the dataset, we evaluate the performance of pre-trained model vs fine-tuned model. Overall, the AP50 scores are significantly lower for off-the-shelf model trained. Table 4.2 shows the results of a pre-trained model vs model trained 100 iterations self-supervised on our dataset. The difference in performance between the two approaches suggests a significant domain shift and strong need for domain adaptation fine-tuning techniques. In addition, our pre-trained model performs worse on our dataset than many other traditional object detection datasets such as COCO and BDD100k. [8] [17].

Furthermore, the decreasing performance as the video progresses shows that micro-domain shift becomes more apparent as scene changes over the course of the video. This decrease in performance for both models over time suggests not only the need for traditional fine-tuning but also online learning or test-time fine-tuning methods that adjust the model as the sequence is evaluated and progresses. Figure 4.2 also shows qualitatively the prediction results of pre-trained model on BDD100k without any domain adaptation, which misses quite a few labels.

Table 4.2: AP50 values for frames at different time intervals evaluated on a sequence from our dataset for model pre-trained on BDD100k and model trained with 100 iterations of update steps on the dataset. Large performance gap shows that clearly there is a significant domain shift.

Model	Overall	2s	8s	32s	128s	512s	3600s
Pre-trained	27.9	64.2	43.4	24.8	30.2	36.4	43.5
Train 100 iters	51.1	100	100	58.4	48.8	52.4	3600

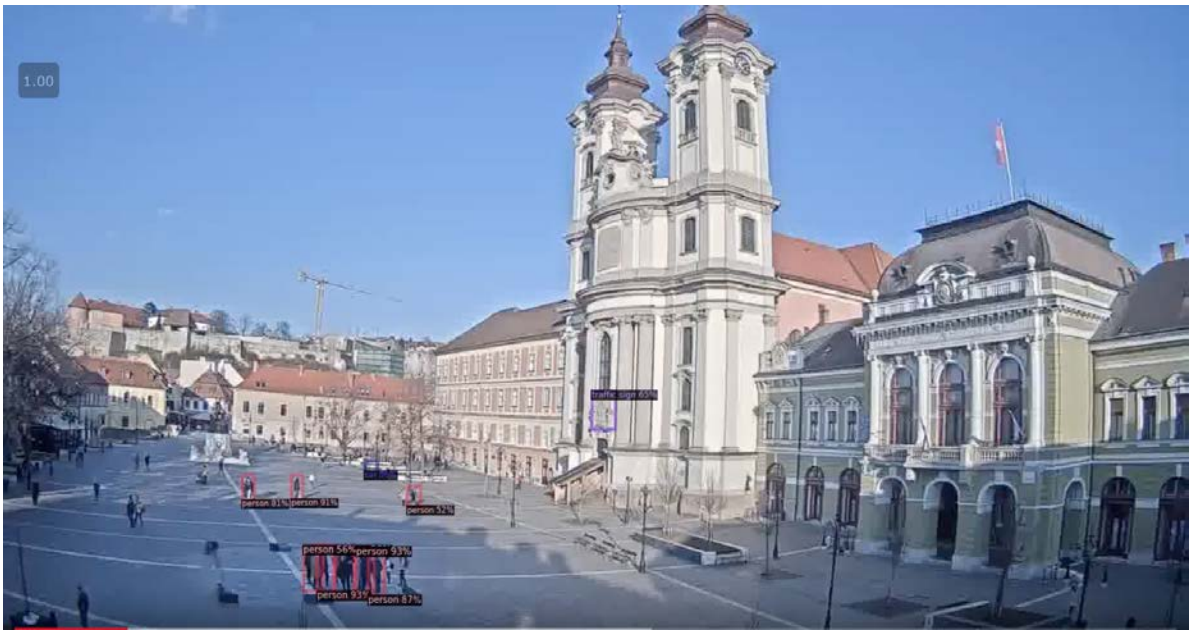


Figure 4.1: Mask-RCNN model pre-trained on BDD100K without fine-tuning, evaluated on a sequence from the dataset. As shown model struggles with domain transfer, missing many of the people on the left side of image and incorrectly classifying traffic signs and cars.

Chapter 5

Online Learning

5.1 Setup and Methods

For our evaluation, we will be using a joint supervised and self-supervised architecture as shown in Fig. 5.1. We have a state-of-the-art supervised object detection model in Mask-RCNN with ResNet50 backbone as well as feature pyramid and region proposal networks, combined with self-supervised heads whose initial layers can be treated as a shared feature extractor [6]. These self-supervised heads can include rotation task, jigsaw task, and cycle-consistency. The goal of this is to be able to use information learned from performing auxiliary self-supervised tasks to help with the main object detection task.

Amongst the self-supervised tasks we use, in the rotation task, we crop various parts of

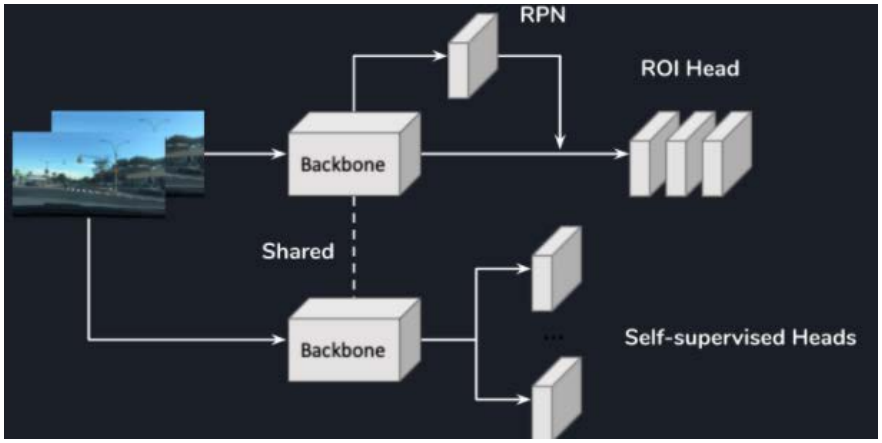


Figure 5.1: Diagram of our joint architecture containing supervised Mask-RCNN model with regional proposal network at top and self-supervised heads for fine-tuning at bottom. These self-supervised heads serve as a shared feature extractor for the model.

the image and flip the image in 4 different orientations and supervise with the rotations as label [5]. In the jigsaw task, we break up the image into tiled regions, label the regions with the location index starting from left to right and top to bottom, and randomly shuffle these regions and have the model predict the indexes of the regions given the shuffled image. [9] In the cycle-consistency task, we learn a tracking feature map and use this representation to track a feature region backward and forward in time and compute difference between coordinates at the end of cycle [15]

In our test-time training (TTT) approach, the models are pre-trained on BDD100k [17] using supervised as well as with cycle consistency tracking loss and another self-supervised head. From here, at test-time we evaluate on our out-of-distribution dataset and update the shared feature extractor by fine-tuning the self-supervised models with the data that we are evaluating on. We also use batch normalization when computing test-time updates which we found to be helpful for the performance. While not analyzed in this iteration of our project. in the future, we also hope to incorporate semi-supervised training into the test-time training, which involves fine-tuning the supervised portion with the loss from object detection prediction on the labeled frames.

5.2 Experimental Results

To better understand the number of training steps we wanted to update with in the online learning, we measured average precision and average detection confidence for a video sequence as we increased the number of training steps in the fine-tuning process for a super-

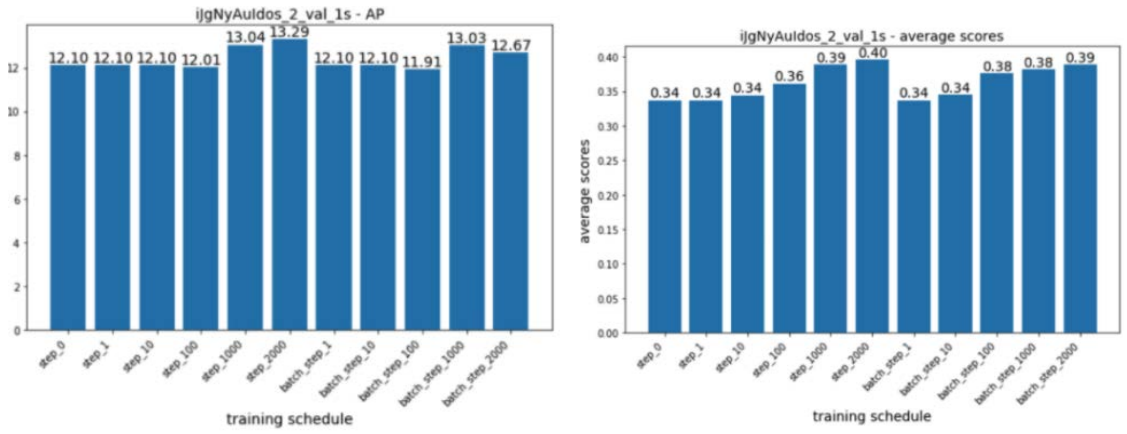


Figure 5.2: Relationship between fine-tuning update steps and average precision (AP) as well as average prediction confidence. Up to 200 steps, average confidence and AP will increase with number of update steps.

Table 5.1: Comparing Average Precision values for baseline supervised object detection and tracking model pre-trained on BDD100k, jointly supervised + self-supervised tasks, and self-supervised with test-time training (TTT) over our dataset as well as BDD100k. For the jigsaw task we use a scale of .01 and for the test-time training we use a learning rate of .001 and train over 2000 steps per image.

Model	AP	AP50	AP75
Baseline Supervised	17.780	31.2674	17.624
Rot	18.498	32.780	18.059
Jig	18.398	32.411	18.086
Rot + Jig	18.141	31.945	18.350
Rot w/ TTT	18.637	32.930	18.457

vised model jointly trained and fine-tuned with a rotation self-supervised task. As shown in Fig. 5.2. we found that up to 2000 steps, as the number of fine-tuning steps increased, so did the AP and average confidence of our model predictions. We use this result to help us determine the learning schedule we wanted for the test-time training.

We use this information to construct our model and hyper-parameters for online learning, in particular we use 2000 steps and a learning rate of .001 for fine-tuning our self-supervised model. This is on top of the standard pre-trained model used for BDD100k. Results of some of our self-supervised joint training and test-time training adaptation on videos is shown in Table 5.2. These models are trained with a baseline supervised model and tracking model from cycle consistency loss. Then, varying combinations of rotation and jigsaw auxiliary loss may also be jointly trained during training-time only or during training-time and fine-tuned at test time depending on the experiment. We can see that the models with self-supervised auxiliary tasks jointly trained outperform the baseline of just the supervised object detection and tracking model. Furthermore, we see that the rotation tasks performs slightly better than the jigsaw task for our particular setting, but that using multiple self-supervised methods at once actually hurts the performance a bit. The highest performance was achieved when we used the rotation self-supervised task in both traditional training as well as test-time training.

From these experiments, we see that online training can help with the performance of object detection models on out of distribution sequences, adapting to both micro and macro changes that occur within a video sequence. We also see that just fine-tuning with a self-supervised task without any online learning may also result in performance boosts. Both jigsaw and rotation are solid auxiliary tasks to use individually in combination with a tracking model using the cycle-consistency loss. However, more work can be done to explore different architectures and learning schedules to measure more noticeable boosts in performance.

Chapter 6

Conclusion and Future Works

In this work, we analyze domain adaptation on object detection tasks for long video, something that has been relatively unexplored for long-videos. These videos may exhibit significant changes in its scene from the beginning to end of frame whether this be from shadow lighting change, sunset, rain, fog, or other external factors. This elicits the need for domain adaptation within a sequence as well, which test-time training can help us tackle. We create and partially label a dataset of hour-long sequences to evaluate domain adaptation as well as test various self-supervised auxiliary tasks to fine-tune the model to the specific domain. Some tasks we used include, rotation task, jigsaw task, and cycle-consistency, of which we found the rotation task to be the most helpful. We were also able to find that test-time training can help overcome some of the challenges with domain adaptation for video.

In the future, we also hope to explore semi-supervised techniques such as bootstrapping online labels to fine-tune the model and test various schemes such that we can optimize partially labeled datasets. Ideally, we would want to be able to use first few labeled data sequences as a means for capturing and fine-tuning feature-level information from a specific target video. Then, periodic labeled updates could be used to continuously correct the model as the scene shifts and could provide even more information than just the self-supervised tasks. In addition, we hope to explore some of the other potential self-supervised tasks such as colorization and in-painting and its effect on online test-time training for domain adaptation. While unlikely to outperform the rotation task, we hope to perform more rigorous ablation testing to determine the best self-supervision tasks in each situation. Finally, we hope to explore adversarial training methods that can help our model with potential alignment issues with the feature extractor and task-specific model.

In terms of the dataset, we hope to expand the current dataset to include more labeled sequences and more labels per sequence. This can help us more effectively evaluate and experiment with various semi-supervised techniques. We hope to also provide more concrete filtering and tagging based off the location of the various scenes in our dataset, time of day, as well as the various setting changes that occur throughout them. In addition, we hope to gather another dataset with videos that contain non-static, panning cameras and compare the domain distributions and various model performances on the the two datasets,

the current still-camera dataset as well as a panning dataset. As a parallel to this, we hope to perform more comprehensive profiling and analysis of the dataset including testing various other computer vision tasks such as segmentation, as well as various other model architectures and their performances on the dataset.

Overall, there are still a lot of interesting problems to solve in the realm of domain adaptation. Our contribution of a dataset, dataset profiling, and initial exploration of modeling approaches, has provided the initial groundwork for further exploration on domain adaptation for long-running videos.

Bibliography

[1] Min-Hung Chen et al. "Temporal Attentive Alignment for Large-Scale Video Domain Adaptation". In: *CoRR* abs/1907.12743 (2019). arXiv: 1907.12743.

[2] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[3] Gabriela Csurka. "Domain Adaptation for Visual Applications: A Comprehensive Survey". In: *CoRR* abs/1702.05374 (2017). arXiv: 1702.05374. URL: http://arxiv.org/abs/1702.05374.

[4] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[5] Spyros Gidaris, Praveer Singh, and Nikos Komodaki. "Unsupervised Representation Learning by Predicting Image Rotations". In: *CoRR* (2018). arXiv: 1803.07728.

[6] Kaiming He et al. "Mask R-CNN". In: *CoRR* abs/1703.06870 (2017). arXiv: 1703.06870.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. "Distilling the Knowledge in a Neural Network". In: *CoRR* abs/1503.02531 (2015).

[8] Tsung-Yi Lin et al. "Microsoft COCO: Common Objects in Context". In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312.

[9] Mehdi Noroozi and Paolo Favaro. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles". In: *CoRR* abs/1603.09246 (2016).

[10] Esteban Real et al. "YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video". In: *CoRR* abs/1702.00824 (2017). arXiv: 1702.00824. URL: http://arxiv.org/abs/1702.00824.

[11] Aruni RoyChowdhury et al. "Automatic adaptation of object detectors to new domains using self-training". In: *CoRR* abs/1904.07305 (2019).

[12] Jake Snell, Kevin Swersky, and Richard S. Zemel. "Prototypical Networks for Few-shot Learning". In: *CoRR* abs/1703.05175 (2017). arXiv: 1703.05175. URL: http://arxiv.org/abs/1703.05175.

[13] Yu Sun et al. "Test-time training for out-of-distribution generalization". In: (2019). arXiv: 1909.13231.

[14] Yu Sun et al. "Unsupervised Domain Adaptation through Self-Supervision". In: *CoRR* abs/1909.11825 (2019). arXiv: 1909.11825.

[15] Xiaolong Wang, Allan Jabri, and Alexei A. Efros. "Learning Correspondence from the Cycle-Consistency of Time". In: *CoRR* abs/1903.07593 (2019). arXiv: 1903.07593.

[16] Jiaolong Xu, Liang Xiao, and Antonio M. López. "Self-supervised Domain Adaptation for Computer Vision Tasks". In: *CoRR* abs/1907.10915 (2019).

[17] Fisher Yu et al. "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.