SMARTADAPT: Multi-branch Object Detection Framework for Videos on Mobiles

Ran Xu[1], Fangzhou Mu[2], Jayoung Lee[1], Preeti Mukherjee[1],
Somali Chaterji[1], Saurabh Bagchi[1], Yin Li[2]
[1]Purdue University [2]University of Wisconsin-Madison
[1]{xu943, lee3716, mukher57, schaterji, sbagchi}@purdue.edu [2]{fmu2, yin.li}@wisc.edu

Abstract

*Several recent works seek to create lightweight deep networks for video object detection on mobiles. We observe that many existing detectors, previously deemed computationally costly for mobiles, intrinsically support adaptive inference, and offer a multi-branch object detection framework (MBODF). Here, an MBODF is referred to as a solution that has many execution branches and one can dynamically choose from among them at inference time to satisfy varying latency requirements (*e.g. *by varying resolution of an input frame). In this paper, we ask, and answer, the wide-ranging question across all MBODFs: How to expose the right set of execution branches and then how to schedule the optimal one at inference time? In addition, we uncover the importance of making a content-aware decision on which branch to run, as the optimal one is conditioned on the video content. Finally, we explore a content-aware scheduler, an Oracle one, and then a practical one, leveraging various lightweight feature extractors. Our evaluation shows that layered on Faster R-CNN-based MBODF, compared to 7 baselines, our* SMARTADAPT *achieves a higher Pareto optimal curve in the accuracy-vs-latency space for the ILSVRC VID dataset.*

1. Introduction

Object detection is arguably one of central problems in computer vision. Much progress has been made over the past few years in deep learning based object detectors. Despite their impressive accuracy results on standard benchmarks, these models come at a price of their complexity and computational cost. This imposes a major barrier to deploy these models under resource-constrained settings with strict latency requirements, such as real-time detection in streaming videos on mobile or embedded devices. Several recent works seek to address this challenge by designing light-weight models on mobiles [11,14,35,44,58], in particular, for video object detection [6,21,22,24]. The assumption and the common belief are that the object detectors that are optimized for accuracy, such as Faster R-CNN with a ResNet-50 backbone, are too expensive for mobile vision.

Indeed, detectors optimized for accuracy are rather complex, often trained with different input resolutions, and equipped with multiple stages (*e.g.* proposal generation). It is perhaps not surprising that these detectors can adapt to different settings at inference time. Consider the example of Faster R-CNN [30], one can reduce the input resolution or the number of proposals for a lower latency while still maintaining a reasonable accuracy. Such combinations of choices of tunable parameters would constitute a multi-branch object detection framework (MBODF). The Faster R-CNN detector using a specific input resolution and a particular number of proposals from our previous example could be considered as one execution branch.

Our key observation is that if one is allowed to select, at inference time, from a large set of fine-grained execution branches, the detection accuracy and latency can be significantly improved (see Figure 1). Then, the key research questions are: *How to expose the right set of execution branches in an existing object detector and then how to schedule the optimal one at inference time?*

An ideal scheduler must not only consider the branches in the model and their properties (accuracy and latency), but also the input content. For example, if the input video only contains larger objects, using a lower input resolution for the detector suffices. Another example is using inexpensive tracking to replace the costly detector, if the video content remains mostly stationary. However, the design of such a scheduler is challenging for streaming videos. This is because the scheduler has to "predict" potential future content change in order to select the best branch at the current time.

In this paper, we focus on the challenging and practical task of streaming video object detection on mobile devices, and present a simple adaptive object detection method. Our key innovation is to leverage standard existing object detectors (Faster R-CNN, EfficientDet, SSD, YOLO) to con-

struct an MBODF for adaptive video object detection. An MBODF combines an object detector and an object tracker, and provides many execution choices (branches).

We demonstrate that our method, notwithstanding its simplicity, can adapt to a wide range of latency requirements, ranging from 10 to 50 FPS on a mobile GPU device — the NVIDIA Jetson TX2 (a widely used device for embedded/mobile vision benchmarks [19, 33, 41, 43]) with only a minor accuracy loss. For example, our method when running at 20 FPS in streaming mode on TX2, achieves an mAP of 70% on a large-scale dataset (ImageNet video object detection benchmark). In contrast, the best performing detector, MEGA [5] that supports streaming videos, has an mAP of 75.4%, and runs only at 1.2 FPS. Further, SMARTADAPT achieves 20.9% to 23.6% higher mAP than our previous multi-branch algorithm [17] given the same constraint on the streaming latency (33-100 msec per frame) (Figure 6, FR+MB vs. FastAdapt).

Next, we uncover the importance of making a content-aware decision on which branch to run, as the optimal one is conditioned on the video content. We explore a content-aware scheduler — an Oracle one, and then a practical one, which uses various light-weight feature extractors, to adapt (at runtime) to the content. We show that our content-aware Oracle scheduler achieves 6.6%–8.3% higher mAP than a content-agnostic one (Table 3, FR+MB+Oracle vs. FR+MB). With our realistic content-aware scheduler (CAS), the gains are more modest, albeit, still present, ranging from 0.1%–2.3% mAP (Table 3, FR+MB+CAS vs. FR+MB, FastAdapt+CAS vs. FastAdapt). The strength of SMARTADAPT is due to the synergistic use of a carefully orchestrated set of features that demonstrate both a low computational overhead and high accuracy, and expose a fine-grained set of branches using MBODF.

Thus, our contributions can be summarized as:

1. We point out that modern detectors are intrinsically adaptive, and can be re-purposed as an MBODF, achieving varying latency and accuracy tradeoffs at inference time, using a set of (individually) proven adaptive attributes.
2. Different from previous solutions [6, 54–56], our work provides a systematic study on the design of MBODF. The result is SMARTADAPT, which combines a set of knobs (*e.g.*, the input resolution and number of proposals), and tunes the ranges and step sizes of these knobs in a fine-grained manner. SMARTADAPT enables deploying existing detectors on a mobile device to span a wide range of latency requirements with minor accuracy drops.
3. We show that an MBODF can achieve significant performance gains when the choice of the execution branches is optimally conditioned on the input content (*e.g.* size and speed of objects). We also take an exploratory step towards practical *content-aware* adaptive object detection.

2. Related Work

Efficient Object Detection Models. Efficiency is paramount on embedded or mobile devices, where power is limited. Many solutions design more efficient network architectures (*e.g.*, [11, 12, 18, 32, 36, 44, 58]), leading to object detectors tailored for mobiles such as SSDLite [32], SqueezeDet [45], Pelee [42], EfficientDet [37], and MobileDets [49]. Several recent works explore temporal redundancy to accelerate video object detection on mobiles, by fusing features from nearby frames [21] or using a gating function that allows convolutions to run on a sparse set of locations [10]. While these methods can effectively reduce the computation cost in terms of FLOPS, they are rarely evaluated on mobile GPUs. Further, reduction in FLOPs does not always translate to reduction in latency [10, 35].

Another stream of studies is to combine a costly object detection module with a relatively inexpensive object tracker module via the "tracking-by-detection" scheme [8, 57]. Our method also builds on "tracking-by-detection", yet significantly extends existing studies to consider a wide range of configurations that may impact the detection performance while focusing on mobile vision.

Adaptive Inference for Image Recognition. These models leverage the content characteristics from the input images and make execution decisions conditioned on them. Previous works integrate several sub-networks [16, 40], or design a network with multiple exits [15, 38, 51], or select input resolution at inference time [55]. However, these works are limited to adaptation in one dimension and with a narrow range, and do not optimize the execution choice by ingesting a rich set of input features available to object detection pipelines. Further, they do not consider video recognition.

Adaptive Inference for Video Recognition. Unlike images, videos exhibit intrinsic temporal redundancy among neighboring frames. Several recent works leverage this redundancy for efficient video classification. These efforts include designing efficient 3D networks [7, 34, 39], dynamically selecting input frame or intermediate feature resolution [25, 47], skipping redundant frames [9, 48], reusing features from previous frames [26], or exploring scheduling strategies in the high-dimensional parameter space [50, 52–54]. Such efforts greatly address challenges in mobile and IoT systems [1].

Only a handful of previous works considered adaptive video object detection, which is fundamentally distinct from video classification. These works include ST-Lattice [4], AdaScale [6], Skip-Conv [10], and our previous works of ApproxDet [54] and FastAdapt [17]. Among these work, ST-Lattice [4] is not designed for mobile vision. AdaScale [6] and Skip-Conv [10] can not achieve explicit tradeoffs between latency and accuracy. The key difference between SMARTADAPT, ApproxDet, and FastAdapt is the ability to switch to multiple fine-grained execution branches

using a content aware scheduler.

3. SmartAdapt: Method and Design

Our goal is to maximize the accuracy of video object detection models for streaming videos at stringent latency constraints (*e.g.* 33 msec) on mobile GPUs. We now present our solution design and describe our techniques.

3.1. Multi-branch Object Detection Framework

Tracking-by-detection. This is our starting point to significantly reduce the latency of the object detection models with a minor accuracy reduction. Rigorously, we define a Group of Frames (GoF) as a sequence of di (detection interval) consecutive frames in a streaming video, in which we run object detectors (*e.g.* Faster R-CNN, EfficientDet, YOLO), on the first frame, and run object tracker (*e.g.* MedianFlow, KCF) on the remaining frames. In the streaming scenario, as we process the video frame-by-frame, an object detector can run on any frame with no prerequisite while an object tracker depends on the detection results, either from a detector, or from the tracker on the previous video frame. Considering our implementation on a Faster R-CNN object detector (in PyTorch [27], with mobile GPU) and a MedianFlow object tracker (in OpenCV [3], with mobile CPU), the tracker runs up to 114X faster, boosting the efficiency.

Tuning Knobs at Inference Time. To further improve the efficiency and avoid a large accuracy reduction, we design tuning knobs for this tracking-by-detection scheme. Our design explores the accuracy-latency tradeoff in five independent dimensions: (1) the detector interval (di), controlling how often an object detector is triggered, (2) the input resolution of the detector (rd), controlling the shape of the resized image fed into the object detector, (3) the number of proposals ($nprop$), controlling the maximum number of region proposals generated from the RPN module of the Faster R-CNN detector, (4) the input resolution of the object tracker (rt), controlling the shape of the resized image fed into the object tracker, and (5) the confidence threshold to track (ct), controlling a minimum threshold on the confidence score of the objects below which the objects are not tracked and output by the tracker. The multi-knob design leads to a combinatorial configuration space as we can tune each knob independently and in various step sizes. This allows for a wide range of adaptations and is key to SmartAdapt's impressive empirical results in what follows.

MBODF. We name the multi-knob tracking-by-detection scheme, with the range and step sizes for each knob, a *Multi-branch Object Detection Framework (MBODF)*. An execution branch in the MBODF is an instance of the values of each knob. Note that not every branch in the configuration space is valid, *e.g.* for branches that run object detector on every frame ($di = 1$), the rt and ct knobs (which are specific to the object tracker) are not relevant.

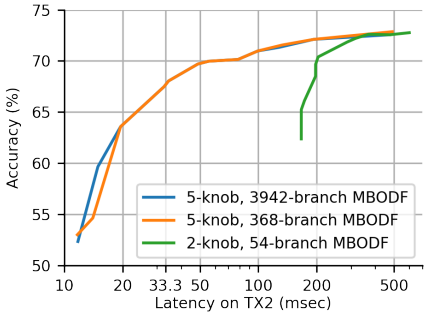


Figure 1. Accuracy comparison of a 5-knob MBODF and a 2-knob sub-framework (input resolution, number of proposals).

Figure 1 shows the accuracy comparison between a 2-knob 54-branch, a 5-knob 368-branch, and a 5-knob 3,942-branch MBODF, where each point on the Pareto optimal curve stands for the accuracy and latency performance of a single branch on the ILSVRC VID dataset. A 5-knob MBODF is much more efficient than a 2-knob MBODF (rd and $nprop$). It achieves a 6.1X speedup, with only a 2.41% mAP reduction, compared to 3.0X speedup, with a 2.37% mAP reduction in the 2-knob MBODF. In contrast, the 5-knob MBODF with 10X more branches (3,942) is only slightly better than the one with a subset of branches (368) at any given value of a latency constraint. The root cause of such reduced accuracy improvement is the *lack of* smarts in choosing the execution branch conditioned on the video content. In other words, if only applying a single static branch on an entire dataset, without finer-grained content revelations (as revealed from Figure 3), one cannot reap the benefit of the much larger-scaled MBODF.

3.2. Branch Selection Problem

A scheduler is a critical component in an MBODF that decides which branch is the optimal one to run, subject to some criteria. Considering an MBODF with $m = |\mathcal{M}|$ independent execution branches $b \in \{b_1, b_2, ..., b_m\}$ that are capable of finishing the object detection task on streaming videos, we use the latency of the branch as the constraint, and maximize its accuracy as the optimization goal:

$$b_{opt} = \arg\max_{b} a(b, \hat{X}), \quad s.t. \quad l(b, \hat{X}) \leq l_0, \tag{1}$$

where $\hat{X}$ denotes the input video frame, l_0 denotes the latency constraints for each frame on average, and $a(b, \hat{X})$ and $l(b, \hat{X})$ represent the accuracy and latency of the branch b. Figure 2 shows the workflow of SmartAdapt where the scheduler takes the video frame as an input and determines the execution branch in the MBODF to run. Inside the scheduler, the workflow is as follows: (1) extracts the content features, (2) predicts the accuracy with a content-aware accuracy predictor, and then (3) uses a branch selector to choose the optimal branch. Particularly, given the tracking-by-detection scheme in the MBODF, where a GoF

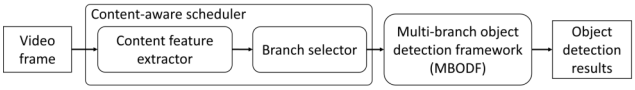


Figure 2. Workflow of SMARTADAPT.

is a unit for scheduling, $\hat{X}$ is relaxed to a GoF. In the streaming scenario, a scheduler should be able to make a decision at any frame x_t in the streaming video where the $\hat{X}$ is the GoF starting from the frame x_t.

As the optimal branch selection is conditioned on the current frame and a few future frames[1], a content-aware scheduler leverages the content characteristics in such a GoF to maximize accuracy. In contrast, a *content-agnostic* scheduler considers the average accuracy of different branches across the entire dataset, which loses the nuances of the snippet-level video characteristics.

In Figure 3, we show the Pareto optimal branches for three randomly selected video snippets of different content characteristics, and the one that inputs the entire dataset for $\hat{X}$. We glean that the accuracy-latency frontiers vary significantly from snippet to snippet and are different from the "average" for the entire dataset (red curve). This motivates the use of a content-aware scheduler for identifying the execution branches for a video object detection pipeline. According to our study, 83.4% branches in the MBODF are most accurate for at least one video snippet at any latency requirement. Among a dataset of 1,256 video snippets, derived from the ILSVRC VID dataset, we find 627 unique sets of accuracy-latency frontier branches. Thus, we conclude that it is important to determine the optimal branch for a given video snippet rather than use a single branch for an entire dataset. This latter approach is also observed with benefit in some prior works, addressed either by using a content-agnostic scheduler [17], or enabling multiple sub-models to choose from [55, 56].

3.3. Content-aware Oracle Scheduler

We define a perfect content-aware scheduler for an MBODF $\mathcal{M}$ an "Oracle" scheduler. Such a scheduler selects the optimal branch b_{opt} to execute. The accuracy-latency performance of an Oracle scheduler establishes the upper-bound performance of a content-aware scheduler, something that has not been established up until now.

To realize an Oracle scheduler, we grant three impractical powers to it — (1) it has access to the future frames in the GoF, (2) it has the annotation of the objects to calculate the ground truth accuracy $a(b, f(\hat{X}))$ so that no predictions are performed, (3) it exhaustively tests all available branches and selects the most accurate one, subject to the latency constraint. Figure 4 shows the performance of the Oracle scheduler on two 5-knob MBODF instantiations, with 3,942 and 368 (a subset) branches and compares

[1]The size of the GoF is typically between 1 and 100 in the MBODF.

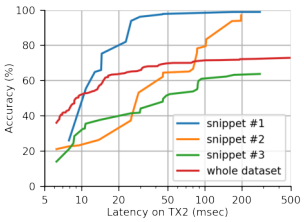


Figure 3. Pareto optimal branches for three selected video snippets of different content characteristics and the whole dataset.

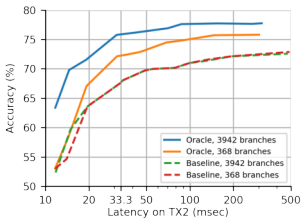


Figure 4. Upper bound performance of a content-aware scheduler, *i.e.* an Oracle.

with a content-agnostic scheduler, which chooses a single static branch for the entire dataset. We observe that the Oracle scheduler has a 3.2% to 4.6% mAP improvement in the 368-branch MBODF at 10, 20, 30, and 50 FPS, four typical latency constraints on mobile devices. This is relative to the baseline with 368 branches. Interestingly, the mAP improvement of the Oracle scheduler is higher for the 3,942-branch MBODF, 6.6%–8.3%, compared to the above-mentioned 3.2%–4.6% (which is for the 368-branch MBODF). In contrast, such large-scaled MBODF has no benefit in the content-agnostic setting. The large gap motivates a content-aware scheduler that can adapt over a large and fine-grained range of knobs.

3.4. Designing a Content-aware Scheduler (CAS)

Our goal is to design a light-weight scheduler to determine the content-specific execution branches on-the-fly, bereft of the impractical powers that we granted to the Oracle. As Eq. 2 suggests, the branch selector in the scheduler requires a latency predictor and an accuracy predictor to solve the optimization. The former has been studied in our previous work [54] through a resource contention sensor and a content-aware latency predictor on each execution branch. In this work, we focus on the design of a *content-aware accuracy predictor* based on simple content features.

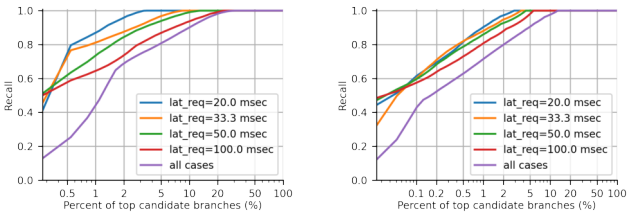
Content Feature Extractors. A content feature extractor aims to build a mapping $f(\cdot)$ from the frame representation $\hat{X}$ to its feature representation since the frame representation carries too much redundancy. The content feature extractor is expected to be discriminative so that the feature values it carries can be used to predict the content-specific accuracy of each execution branch. Then, a content-aware accuracy prediction model aims to build a mapping $a(\cdot)$ from the feature representation $f(\hat{X})$ to the accuracy of a given execution branch b. Thus, the scheduler model can be formulated as follows:

$$b_{opt} = \arg\max_{b} a(b, f(\hat{X})), \quad s.t. \quad l(b, \hat{X}) \leq l_0. \qquad (2)$$

A well-designed content feature extractor should be rich in content characteristics, discriminative enough, and light-weight in the computation. Table 1 summarizes the list of our content features, specs, and descriptions. We start from

Name	Dim.	Trainable	Description
light	4	No	Composed of height, width, number of objects, averaged size of the objects
HoC	768	No	Histograms of Color on red, green, blue channels
HOG	5400	No	Histograms of Oriented Gradients
ResNet50	1024	No	ResNet50 features from the object detector in the MBODF, average pooled over height and width dimensions, and only preserving the channel dimension
CPoP	31	No	"Class Predictions on the Proposal" feature (CPoP) from the object detector of the MBODF, averaged pooled over all region proposals, and only preserving the class dimension (including a background class)
MobileNet	1280	**Yes**	Efficient, effective feature extractor, average pooled from the feature map before the fully-connected layer, and only preserving the channel dimension

Table 1. Feature extractors in SMARTADAPT's content-aware scheduler.



(a) A 368-branch MBODF (b) A 3,942-branch MBODF

Figure 5. Recall of top candidate branches (by percentage of the number of branches in MBODF) from the Optimal Branch Election.

some light features that come with no cost to extract, *i.e.* the height and width of the video frame, the number of objects, and average size of the objects. We then choose two traditional vision features — Histograms of Color (HoC) and Histograms of Oriented Gradients (HOG) to characterize the color and gradient information. As the object detector itself is a neural network with intermediate features, we average pool one from the layer after the feature extractor head of Faster R-CNN backbone, *i.e.* ResNet-50, and one from the prediction logits on the object classes. These two features are attractive as they incur no extra computation cost, yet encode the object information within videos. Finally, we propose to use a widely used DNN-based feature extractor, MobileNetV2 [32]. It is lightweight in terms of the computation cost and jointly trainable with the downstream content-aware accuracy predictor. Naturally, at inference time, the scheduler has to run ahead of the MBODF and thus has to rely on extracted content features from the previous GoF. Due to the temporal smoothness in video frames [9, 25], this simplification works well in practice.

Content-aware Accuracy Predictor. A content-aware accuracy predictor infers the accuracy of all branches in the MBODF given a feature vector. We use a 5-layer fully connected neural network (NN) with ReLU, 256 neurons in all hidden layers, and residual connections [13]. As the dimensions of the light features and other features vary significantly in 1 to 3 orders of magnitude, we add a feature projection layer before the features are concatenated and fed into the 5-layer NN. The feature projection layer projects both light features and other high-dimensional features to fixed 256-dimensional vectors so that they are equally representative in the accuracy predictor. We use MSE loss and train the NN on a derived snippet-granularity dataset from ILSVRC VID (see Sec. 4 for details), where the ground truth accuracy of the branches are profiled offline.

Joint Modeling of Content and Latency Requirement. We additionally explore a network that jointly models content and latency requirement for branch selection. Different from the previous design, this model does not pair with a latency predictor and thus is simpler in design. Specifically, we begin by embedding content and latency requirement into separate feature vectors using multi-layer perceptrons (MLPs). Following FiLM [28], our model regresses a set of affine weights γ and biases β from the latency feature F_l using another MLP and subsequently transforms the content feature F_c as $F_c^{'} = \gamma \cdot F_c + \beta$. In doing so, our model adapts to the current latency requirement through the modulation of content features. An MLP further processes the modulated content features $F_c^{'}$ and predicts accuracy of all branches. We train our model using the same MSE loss as before, except that we set the target accuracy of a branch to zero when latency requirement is violated. We show in our experiments that this joint modeling scheme is most effective under tight latency constraint.

Candidate Branches. Predicting on thousands of execution branches is challenging. SMARTADAPT narrows down the number of candidate execution branches in the design phase to top K. The intuition is that the top K execution branches should cover the majority of optimal branches across videos of different content characteristics and different latency constraints, for properly chosen K. We use the method called *Optimal Branch Election* (OBE) to select the K candidate branches. Figure 5(a) shows the recall of using K branches (*i.e.* proportion where the optimal branch belongs to one of the top K), rather than all 368 branches. We see that in the 368-branch MBODF, 10.1% branches suffice to achieve 90% recall. Also, if we consider the candidate branches *for a particular latency constraint*, even fewer can be considered. To achieve a 90% recall, the percentages of K branches are 1.4%, 2.7%, 3.3%, and 7.1%, given 20, 33.3, 50, and 100 msec latency constraints. Figure 5(b) shows such relation on a larger-scaled MBODF with 3,942 branches, with a lower ratio of branches that need to be considered. Thus, using top K candidates can effectively reduces the cost of online scheduling and offline profiling.

4. Implementations

All models are trained on a server with two NVIDIA P100 GPUs; evaluated on NVIDIA TX2 with a 256-core NVIDIA Pascal GPU on a 8GB unified memory. Our method is implemented in PyTorch, yet further speedup using TensorRT might be possible.

Profiling. Once the tuning knobs are determined for an object detector, it is important to determine the ranges and step

di	*rd*	*nprop*	*rt*	*ct*
1,2,4,8	224*,352,384,288,320,	3*,5*,10*,20*	25%,50%	0.05,0.1
20,50,100*	416*,448*,480*,512*	100, 1000	100%	0.2,0.4*

Table 2. Choices of the tuning knobs in the MBODF with Faster R-CNN object detector in the 368-branch variant (* indicates additional choices in the 3,942-branch variant).

sizes of values for each knob. We profile the multi-knob tracking-by-detection scheme and evaluate the accuracy-latency relation on each knob. We then determine the ranges and step sizes according to the monotonic range of such relation and the constraints of each knob. Finally, we implement our MBODF on top of Faster R-CNN (a 368-branch and a 3,942-branch variant), EfficientDet, YOLOv3, and SSD. Specifically, we implement 5 tuning knobs for the Faster R-CNN object detector (Table 2). Those for the other detectors are in the **Supplement**.

Snippet-granularity Dataset. We derive a snippet-granularity dataset to study the content-aware accuracy of the execution branches. Given a video dataset $\{v_1, v_2, ...v_h\}$ with h videos, we clip each video into l-frame video snippets, and each video snippet is our unit for evaluating content-specific accuracy. Too small an l value makes mAP meaningless, and too large an l, reduces the content-aware granularity. We choose $l = 100$ for the ILSVRC 2015 VID dataset. To further enlarge the training dataset, we use sliding windows to extract more video snippets. Supposing a temporal stride of s frames, every l-frame snippet starting at the frame whose index is the multiple of s is selected as a video snippet (we use $s = 5$), enlarging the training dataset by a factor of l/s.

Training content-aware scheduler model. We train our content-aware accuracy predictors for 400 epochs, with a batch size of 64, a weight decay of 0.01, and an SGD optimizer of fixed learning rate of 0.01, and momentum of 0.9.

5. Experiments

Our experimental results are composed of three parts. *First*, we evaluate our best performing models over multiple backbone object detectors and compare with the *content-agnostic baselines*. *Second*, we perform ablation studies of our techniques over the MBODF with Faster R-CNN (FR+MB+CAS) and FastAdapt (FastAdapt+CAS) protocols and study the impact of *content-aware techniques*. *Finally*, we discuss the benefit of post-processing methods on the accuracy and latency cost of both the offline profiling and the online scheduler. We report results on the ILSVRC 2015 VID dataset and a snippet-granularity derivative of the dataset (only Table 4), and use different latency constraints to demonstrate the strength of our method. We achieve 70% mAP accuracy at 20 FPS and lead the accuracy frontier at a wide range of latency constraints. Before we present our results, we summarize our evaluation scenario, dataset and metrics, and naming convention for the protocols.

Streaming Inference. As we study the efficient and adaptive object detection systems on mobiles, the typical usage scenario is to process the videos at the speed of their source, *i.e.* 30 FPS, in the streaming style. This means (1) one cannot use the raw video frame or features of video frames in the future to refine the detection results on the current frame, (2) one cannot refine the detection results of past frames, and (3) the algorithm should process the video frame-by-frame in the timestamp order. We discuss the comparison with other protocols in the offline mode with post-processing techniques in Sec. 5.2.

Dataset and Metrics. We use ILSVRC 2015 VID dataset for our evaluation. Particularly, we train our feature extractors and accuracy predictors on our snippet-granularity dataset derived from the ILSVRC 2015 VID training dataset, which contains 3,862 videos. Our snippet-granularity dataset of 1,256 video snippets is derived from 10% videos in the training dataset, considering the significant amount of execution branches in our MBODF. We evaluate our models on both ILSVRC 2015 VID validation dataset and our snippet-granularity dataset. The former contains 555 videos, and we evaluate object detection performance by reporting (1) mean Average Precision (mAP) at IoU=0.5 as the accuracy metric and (2) mean execution latency per frame on the NVIDIA Jetson TX2 as the latency metric. The latter has 1,965 video snippets. Here we evaluate our accuracy prediction results, and report Mean Squared Error (MSE), Spearman Rank Correlation (SRC), and Recall of the most accurate branches between the predicted accuracy and the ground truth accuracy.

Protocols. We formulate several protocols that implement a set of techniques for efficient video object detection. We replicate the SOTA object detectors and create MBODF for each model by designing tuning knobs and determining ranges and step sizes for each knob (Sec. 4). The variants of SMARTADAPT (anything with "MB" or "CAS" in the name) and baselines are as follows:

- **FR+MB** Our MBODF on top of the Faster R-CNN [30] object detector with ResNet-50 [13] and FPN [20]. We have a 368-branch and a 3,942-branch variant due to the different ranges and step sizes in each knob.
- **ED+MB**: Our MBODF on EfficientDet [37].
- **YL+MB**: Our MBODF on YOLOv3 [29].
- **SSD+MB**: Our MBODF on SSD [23].
- **FastAdapt [17]**: An adaptive object detection system with 1,036 approximation branches and a content-agnostic scheduler.
- **ApproxDet [54]**: Another adaptive object detection system, but less efficient than FastAdapt.
- **FR+MB+CAS**: Our content-aware scheduler with our MBODF on top of Faster R-CNN.
- **FastAdapt+CAS**: Our content-aware scheduler with an

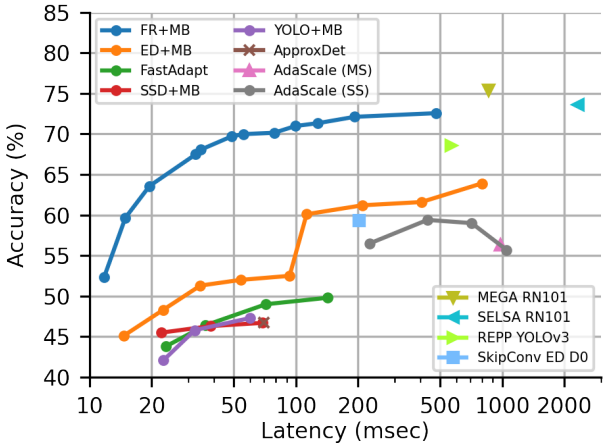


Figure 6. Accuracy-latency frontier. Our MBODF on top of Faster R-CNN (FR+MB) achieves higher accuracy at a wide latency range (2 - 85 FPS).

off-the-shelf adaptive object detection system.

- **AdaScale [6]**: an adaptive and efficient video object detection model with a scale knob. We evaluate a multi-scale (MS) variant as its main design, and include several single scales (SS) for comparison.
- **Skip-Conv ED D0 [10]**: We use the norm-gate variant of Skip-Conv on top of an EfficientDet D0 model. The original implementation only shows MAC and wall time reduction on *CPUs*. We evaluate Skip-Conv on the mobile GPU to compare with SmartAdapt.
- **MEGA RN101 [5]**: ResNet 101 version of MEGA. In our streaming inference scenario, we cannot access the frames or features in the future or refine the detection in the past. Thus, we report the accuracy of the still-image object detection baseline in MEGA. This applies to SELSA RN101 and REPP YOLOv3 as well.
- **SELSA RN101 [46]**: ResNet-101 version of SELSA.
- **REPP YOLOv3 [31]**: YOLOv3 version of REPP.

5.1. Performance of MBODFs

Figure 6 shows the accuracy and latency performance of each protocol, in which the latency scale is logarithmic to include a large variety of protocols. We can observe that our FR+MB protocol leads the accuracy-latency frontiers compared to baselines and other MBODFs in our work. Particularly, FR+MB achieves 67.5% mAP at 30 FPS, 69.7% mAP at 20 FPS, 71.0% mAP at 10 FPS on the TX2. The adaptation range is 40.5x in latency (9.8x within 3% accuracy reduction) and the accuracy is superior to all other protocols given the same latency constraint. On the other hand, our ED+MB, YL+MB, and SSD+MB also enhance the efficiency to achieve the real-time inferencing speed (30 FPS). As for baseline protocols, MEGA and SELSA, with their deeper ResNet 101 kernel, they are 2.9% and 1.1% more accurate than our most accurate branch in FR+MB and much slower than us (running at 1.2 and 0.4 FPS). REPP, SkipConv, AdaScale, FastAdapt and ApproxDet are both worse than our FR+MB protocol with lower accuracy and higher latency. To conclude, our MBODFs on top of four popular object detectors can greatly enhance the efficiency to achieve real-time speed and the best of them, FR+MB, leads the accuracy-latency frontier and has comparable accuracy with the accuracy optimized models.

Protocols	20.0 ms	33.3 ms	50 ms	100 ms
FR+MB+Oracle (3,942 br.)	71.5%	75.8%	76.3%	77.6%
FR+MB+Oracle (368 br.)	67.1%	72.1%	72.9%	74.8%
FR+MB+CAS	**64.1%**	**68.3%**	**69.8%**	**71.1%**
FR+MB	63.6%	67.5%	69.7%	71.0%
FastAdapt+CAS	N/A	**46.1%**	**47.1%**	**50.3%**
FastAdapt	N/A	43.8%	46.4%	49.0%
ED+MB	45.1%	51.3%	52.0%	52.5%
SSD+MB	N/A	45.5%	46.3%	46.7%
YL+MB	N/A	42.1%	45.8%	47.3%
ApproxDet	N/A	N/A	N/A	46.8%

Table 3. Accuracy comparison of SmartAdapt over all efficient baselines given stringent latency constraints on the ILSVRC VID validation dataset. The object detectors FR, ED, SSD, and YOLO cannot meet the 100 msec latency constraint with a MBODF and thus not shown. N/A means that the accuracy is unusably low.

metrics	MSE		SRC		Recall	
features	368 br.	3,942 br.	368 br.	3,942 br.	368 br.	3,942 br.
baseline	0.091	0.109	0.377	0.376	0.354	0.343
light	0.083	0.109	0.385	**0.385**	0.368	0.347
HoC	0.083	0.109	**0.387**	**0.385**	**0.369**	**0.348**
HOG	0.084	0.103	0.386	0.384	0.347	**0.348**
MobileNet	**0.082**	**0.102**	0.385	**0.385**	0.368	0.347
MobileNet Tr.	0.083	N/A	0.385	N/A	0.361	N/A

Table 4. Evaluation of our content-aware MBODF on top of Faster R-CNN object detector with different content extractors against the content-agnostic MBODF (baseline) on the snippet-level dataset. N/A means the training cannot finish in a reasonable time.

We then look into all adaptive and efficient protocols that are able to run within 100 msec per frame (10 FPS speed) and examine the accuracy at 50, 30, 20, and 10 FPS in Table 3. The results show that FR+MB+CAS achieves marginally better accuracy results than FR+MB by 0.1% to 0.8% mAP through its content-aware scheduler. Compared to the FastAdapt baseline, our content-aware scheduler achieves a higher benefit, 0.7% to 2.3% mAP improvement. Note that our CAS results are still far from our Oracle results because (1) we cannot exhaustively run every branch and select the most accurate one, (2) we cannot access annotations online to calculate the ground truth accuracy, and (3) we cannot access future frames and the scheduler's choice is based on the features of current and past frames. To summarize, in addition to the illuminating results in Figure 6, our exploration on the content-aware design boosts the accuracy-latency frontier further.

We further evaluate the CAS with different feature extractors. On the snippet-level dataset, Table 4 shows the MSE, SRC, and recall of our full stack of techniques with different off-the-shelf and trainable feature extractors, on top of a 368-branch and a 3,942-branch FR+MB. The re-

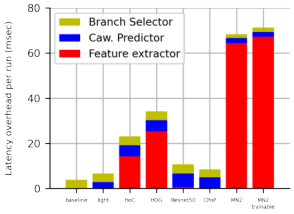


Figure 7. Latency breakdown in the CAS, per-run, measured on the TX2 board. Note that the CAS does not run on every frame. Thus higher cost is acceptable.

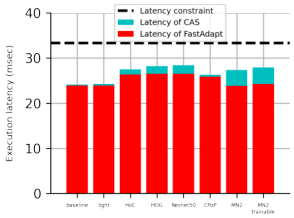


Figure 8. Latency of the content-aware scheduler averaged along the video, on top of the latency of the execution kernel of FastAdapt.

sults show consistent lower MSE, higher SRC, and recall in the CAS of all feature extractors compared to the content-agnostic baseline. We include a further detailed ablation study with feature extractors, candidate branches, and joint training and modeling in the **Supplement**.

5.2. Further Discussions

Impact of Post-processing & Accuracy in Offline Mode. To fairly compare SMARTADAPT with the accuracy optimized models, we apply REPP [31] and Seq-BBox Matching (SBM) [2] post-processing methods to our detection results in the offline mode. The averaged mAP improvement over FR+MB is 2.60% and 2.38%. We show the details of the accuracy improvement on each accuracy-latency frontier branches of the two approaches in the **Supplement**. While the latency cost of these processing techniques is heavily dependent on the number of objects in a given video, the overall averaged latency cost per frame is 24 msec for REPP and 9 msec for SBM. Furthermore, we have evaluated these post-processing methods in online mode.

We find that the latency cost in such online mode is more than 100X higher than that in the offline mode since the post-processing is iteratively done on every frame. This points to the difficulty of improving accuracy in streaming content through existing post-processing techniques.

Offline Profiling Cost. The cost of profiling MBODF to realize an Oracle scheduler and derive a snippet-granularity dataset to study content-aware accuracy is significant. Considering the 3,942-branch MBODF on top of the FRCNN object detector, in the basic case, we need to run every branch on the training and test datasets to collect its latency and accuracy. We deploy a set of engineering techniques to speed up the profiling by parallelizing it (accuracy profile can run on the server, not on the mobile device) and reusing results across branches (details in **Supplement**). Combining these techniques, we are able to finish the profiling within 5 days on two servers (specs in Sec. 4).

Overhead of the Content-aware Scheduler (CAS). While the CAS improves accuracy-latency frontier of the MBODF, we further evaluate its latency overhead because a naïve design will result in additional overhead of the scheduler on top of the latency of MBODF. Figure 7 shows the latency breakdown in the CAS. The cost of light feature is zero, and the cost of ResNet50 and CPoP feature extractors are minor, since ResNet50 and CPoP features come from the object detector itself. The costs of the HoC and HOG features are intermediate, between 20 to 35 msec per run, adding a minor overhead considering its triggering frequency ranges from every 8 to 50 frames. The cost of a MobileNetV2 features, whether trainable or not, is around 65 msec per run.

Figure 8 further evaluates FastAdapt+CAS with a 33.3 msec latency constraint. The latency of the execution kernel is almost the same and summed latency meets the latency budget for all feature extractors (including the most expensive MobileNetV2), owing to a conservative branch selection strategy where the branch selector uses 95th percentile latency as the criteria to choose the branch. Furthermore, we find that the latency cost of MobileNetV2 can be reduced by 20% using a smaller input resolution of 64x64x3, with similar performance —one of many optimizations that we can leverage to further reduce the cost.

6. Conclusion

We have demonstrated in a multi-branch (video) object detection framework (MBODF) how to expose the right set of execution branches, and then, how to schedule the optimal one at inference time. We uncovered the importance of making a *content-aware* decision on the execution branch to run. Finally, we explored a content-aware scheduler SMARTADAPT, an Oracle one, and then a practical one, which uses various light-weight feature extractors, to adapt to the content at runtime. We demonstrated that our method, notwithstanding its simplicity, can adapt to a wide range of latency requirements (range of 40X), on a mobile GPU device, NVIDIA Jetson TX2. SMARTADAPT, integrated with Faster R-CNN as the MBODF, leads the Pareto optimal accuracy-vs-latency frontier over 7 baselines. SMARTADAPT outperforms a content-agnostic MBODF baseline, FastAdapt, by 20.9%–23.6% mAP. Our work can benefit from ongoing work on better video feature extractors (to be used in our content-aware scheduler), improved video object detection and tracking models, and better post-processing algorithms.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation under Grant Numbers CCF-1919197, CNS-2038986, CNS-2038566, and CNS-2146449 (NSF CAREER award), by the Amazon Web Service AI award, and by the Army Research Lab under contract numbers W911NF-20-2-0026 and W911NF-2020-221. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Saurabh Bagchi, Tarek F Abdelzaher, Ramesh Govindan, Prashant Shenoy, Akanksha Atrey, Pradipta Ghosh, and Ran Xu. New frontiers in iot: Networking, systems, reliability, and security challenges. *IEEE Internet of Things Journal*, 7(12):11330–11346, 2020. 2

[2] Hatem Belhassen, Heng Zhang, Virginie Fresse, and El-Bay Bourennane. Improving video object detection by seq-bbox matching. In *VISIGRAPP (5: VISAPP)*, pages 226–233, 2019. 8

[3] Gary Bradski. The opencv library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, 25(11):120–123, 2000. 3

[4] Kai Chen, Jiaqi Wang, Shuo Yang, Xingcheng Zhang, Yuanjun Xiong, Chen Change Loy, and Dahua Lin. Optimizing video object detection via a scale-time lattice. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 7814–7823, 2018. 2

[5] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. Memory enhanced global-local aggregation for video object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10337–10346, 2020. 2, 7

[6] Ting-Wu Chin, Ruizhou Ding, and Diana Marculescu. AdaScale: Towards real-time video object detection using adaptive scaling. *Proceedings of Machine Learning and Systems (MLSys)*, 1:431–441, 2019. 1, 2, 7

[7] Christoph Feichtenhofer. X3D: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 203–213, 2020. 2

[8] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 3038–3046, 2017. 2

[9] Amir Ghodrati, Babak Ehteshami Bejnordi, and Amirhossein Habibian. FrameExit: Conditional early exiting for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15608–15618, 2021. 2, 5

[10] Amirhossein Habibian, Davide Abati, Taco Cohen, and Babak Ehteshami Bejnordi. Skip-convolutions for efficient video processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2695–2704, 2021. 2, 7

[11] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. GhostNet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1580–1589, 2020. 1, 2

[12] Kai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing Xu, and Tong Zhang. Model rubik's cube: Twisting resolution, depth and width for tinynets. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:19353–19364, 2020. 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016. 5, 6

[14] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for MobileNetV3. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 1

[15] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[16] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision (ECCV)*, pages 646–661. Springer, 2016. 2

[17] Jayoung Lee, Pengcheng Wang, Ran Xu, Venkat Dasari, Noah Weston, Yin Li, Saurabh Bagchi, and Somali Chaterji. Benchmarking video object detection systems on embedded devices under resource contention. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, pages 19–24, 2021. 2, 4, 6

[18] Yunsheng Li, Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Lu Yuan, Zicheng Liu, Lei Zhang, and Nuno Vasconcelos. MicroNet: Improving image recognition with extremely low flops. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 468–477, 2021. 2

[19] Ji Lin, Chuang Gan, and Song Han. TSM: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7083–7093, 2019. 2

[20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 2117–2125, 2017. 6

[21] Mason Liu and Menglong Zhu. Mobile video object detection with temporally-aware feature maps. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 5686–5695, 2018. 1, 2

[22] Mason Liu, Menglong Zhu, Marie White, Yinxiao Li, and Dmitry Kalenichenko. Looking fast and slow: Memory-guided mobile video object detection. *arXiv preprint arXiv:1903.10172*, pages 1–10, 2019. 1

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 9907, pages 21–37, 2016. 6

[24] Hao Luo, Wenxuan Xie, Xinggang Wang, and Wenjun Zeng. Detect or track: Towards cost-effective video object detection/tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8803–8810, 2019. 1

[25] Yue Meng, Chung-Ching Lin, Rameswar Panda, Prasanna Sattigeri, Leonid Karlinsky, Aude Oliva, Kate Saenko, and

Rogerio Feris. AR-Net: Adaptive frame resolution for efficient action recognition. In *European Conference on Computer Vision (ECCV)*, pages 86–104. Springer, 2020. 2, 5

[26] Yue Meng, Rameswar Panda, Chung-Ching Lin, Prasanna Sattigeri, Leonid Karlinsky, Kate Saenko, Aude Oliva, and Rogerio Feris. AdaFuse: Adaptive temporal fusion network for efficient action recognition. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems (NeurIPS)*, pages 8024–8035, 2019. 3

[28] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 5

[29] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, pages 1–6, 2018. 6

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 91–99, 2015. 1, 6

[31] Alberto Sabater, Luis Montesano, and Ana C Murillo. Robust and efficient post-processing for video object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10536–10542. IEEE, 2020. 7, 8

[32] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 4510–4520, 2018. 2, 5

[33] Karthick Shankar, Pengcheng Wang, Ran Xu, Ashraf Mahgoub, and Somali Chaterji. JANUS: Benchmarking commercial and open-source cloud and edge platforms for object and anomaly detection workloads. In *Proceedings of the IEEE International Conference on Cloud Computing (CLOUD)*, pages 590–599, 2020. 2

[34] Ximeng Sun, Rameswar Panda, Chun-Fu Richard Chen, Aude Oliva, Rogerio Feris, and Kate Saenko. Dynamic network quantization for efficient video inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7375–7385, 2021. 2

[35] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. MnasNet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2820–2828, 2019. 1, 2

[36] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 2

[37] Mingxing Tan, Ruoming Pang, and Quoc V Le. EfficientDet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10781–10790, 2020. 2, 6

[38] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE, 2016. 2

[39] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6450–6459, 2018. 2

[40] Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–18, 2018. 2

[41] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. CSPNet: A new backbone that can enhance learning capability of CNN. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391, 2020. 2

[42] Robert J Wang, Xiang Li, and Charles X Ling. Pelee: A real-time object detection system on mobile devices. *Advances in neural information processing systems (NeurIPS)*, 31, 2018. 2

[43] Diana Wofk, Fangchang Ma, Tien-Ju Yang, Sertac Karaman, and Vivienne Sze. FastDepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108. IEEE, 2019. 2

[44] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10734–10742, 2019. 1, 2

[45] Bichen Wu, Forrest Iandola, Peter H Jin, and Kurt Keutzer. SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 129–137, 2017. 2

[46] Haiping Wu, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Sequence level semantics aggregation for video object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9217–9225, 2019. 7

[47] Zuxuan Wu, Caiming Xiong, Yu-Gang Jiang, and Larry S Davis. LiteEval: A coarse-to-fine framework for resource efficient video recognition. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:7780–7789, 2019. 2

[48] Zuxuan Wu, Caiming Xiong, Chih-Yao Ma, Richard Socher, and Larry S Davis. Adaframe: Adaptive frame selection for

fast video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1278–1287, 2019. 2

[49] Yunyang Xiong, Hanxiao Liu, Suyog Gupta, Berkin Akin, Gabriel Bender, Yongzhe Wang, Pieter-Jan Kindermans, Mingxing Tan, Vikas Singh, and Bo Chen. MobileDets: Searching for object detection architectures for mobile accelerators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3825–3834, 2021. 2

[50] Ran Xu, Jinkyu Koo, Rakesh Kumar, Peter Bai, Subrata Mitra, Sasa Misailovic, and Saurabh Bagchi. VideoChef: efficient approximation for streaming video processing pipelines. In *2018 USENIX Annual Technical Conference (USENIX ATC)*, pages 43–56, 2018. 2

[51] Ran Xu, Rakesh Kumar, Pengcheng Wang, Peter Bai, Ganga Meghanath, Somali Chaterji, Subrata Mitra, and Saurabh Bagchi. ApproxNet: Content and contention-aware video analytics system for embedded clients. *ACM Trans. Sen. Netw. (TOSN)*, 18(1), 2021. 2

[52] Ran Xu, Subrata Mitra, Jason Rahman, Peter Bai, Bowen Zhou, Greg Bronevetsky, and Saurabh Bagchi. Pythia: Improving datacenter utilization via precise contention prediction for multiple co-located workloads. In *Proceedings of the 19th International Middleware Conference*, pages 146–160, 2018. 2

[53] Ran Xu, Haoliang Wang, Stefano Petrangeli, Viswanathan Swaminathan, and Saurabh Bagchi. Closing-the-Loop: A data-driven framework for effective video summarization. In *Proceedings of the 22nd IEEE International Symposium on Multimedia (ISM)*, pages 201–205, 2020. 2

[54] Ran Xu, Chen-lin Zhang, Pengcheng Wang, Jayoung Lee, Subrata Mitra, Somali Chaterji, Yin Li, and Saurabh Bagchi. ApproxDet: content and contention-aware approximate object detection for mobiles. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems (SenSys)*, pages 449–462, 2020. 2, 4, 6

[55] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. Resolution adaptive networks for efficient inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2369–2378, 2020. 2, 4

[56] Le Yang, Haojun Jiang, Ruojin Cai, Yulin Wang, Shiji Song, Gao Huang, and Qi Tian. CondenseNet V2: Sparse feature reactivation for deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3569–3578, 2021. 2, 4

[57] Chun-Han Yao, Chen Fang, Xiaohui Shen, Yangyue Wan, and Ming-Hsuan Yang. Video object detection via object-level temporal aggregation. In *European conference on computer vision (ECCV)*, pages 160–177. Springer, 2020. 2

[58] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 6848–6856, 2018. 1, 2