Delta Distillation for Efficient Video Processing

Amirhossein Habibian[1], Haitam Ben Yahia[1], Davide Abati[1],
Efstratios Gavves[2], and Fatih Porikli[1]

[1]Qualcomm AI Research** [2]University of Amsterdam
{ahabibia,hyahia,dabati,fporikli}@qti.qualcomm.com
egavves@uva.nl

Abstract. This paper aims to accelerate video stream processing, such as object detection and semantic segmentation, by leveraging the temporal redundancies that exist between video frames. Instead of propagating and warping features using motion alignment, such as optical flow, we propose a novel knowledge distillation schema coined as Delta Distillation. In our proposal, the student learns the variations in the teacher's intermediate features over time. We demonstrate that these temporal variations can be effectively distilled due to the temporal redundancies within video frames. During inference, both teacher and student cooperate for providing predictions: the former by providing initial representations extracted only on the key-frame, and the latter by iteratively estimating and applying deltas for the successive frames. Moreover, we consider various design choices to learn optimal student architectures including an end-to-end learnable architecture search. By extensive experiments on a wide range of architectures, including the most efficient ones, we demonstrate that delta distillation sets a new state of the art in terms of accuracy vs. efficiency trade-off for semantic segmentation and object detection in videos. Finally, we show that, as a by-product, delta distillation improves the temporal consistency of the teacher model.

1 Introduction

The goal of this paper is to accelerate the processing of video streams, such as object detection and semantic segmentation. Despite the great progress in the development of efficient architectures [2,23,50,15,44,42], highly accurate models are still too expensive to process video frames in real-time. This aspect hinders the deployment of accurate models on constrained settings, *i.e.* mobile devices.

To this end, recent works apply accurate yet expensive models only on a subset of frames, referred to as key-frames, and process the remaining ones using a lighter architecture [56,18,54,24,25,26]. The representations from the light model are then aggregated with the key-frame representations from the expensive model in a recurrent structure: this step is typically performed at a deep layer, in order to leverage the representation power of the expensive model. Due to the misalignment between the current frame and key-frame, this strategy proves to be

[**] Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

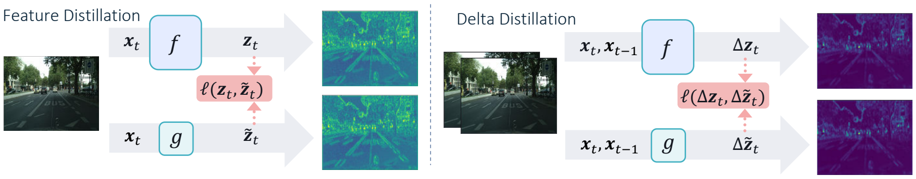

Fig. 1: Feature distillation vs. Delta distillation. Instead of distilling features $\mathbf{z}$ computed with an expensive layer f, we distill to a cheap student g their changes across frames, $\Delta\mathbf{z}$. Due to temporal correlation in videos, transitions between frames are smooth and deltas are smaller (as visible) thus easier to distill.

effective only when explicit motion alignment is carried out, typically by means of optical flow warping [56,18,54]. For this reason, feature aggregation is a viable solution under the assumption that the overhead for extracting motion vectors is lower than in computations within the bypassed feature extraction. Although this condition is easy to meet for expensive backbones such as ResNet-101, it has become less reasonable with the development of efficient models such as EfficientDet [42] or HRNet [44].

This paper introduces a novel approach to leverage the redundancies in a video to speed up the inference. Our proposal does not rely on explicit motion alignment and is applicable to any architecture, including the most efficient ones, *i.e.* EfficientDet-D0 [42], HRNet [44] and the very recent DDRNet [13]. Our approach, coined as Delta Distillation , is based on knowledge distillation [7,12], a popular technique to accelerate an expensive *teacher* network by distilling it into a lightweight *student*. Given an expensive teacher processing only key-frames, for every block of layers, we instantiate a cheap student counterpart, that is fed with a pair of frames and regresses their corresponding difference (delta) in teacher activations. During training, the teacher provides target deltas, and the regression error of the student is minimized by an ℓ_2 objective. During inference, the teacher provides the representations for the key-frame, and the student iteratively updates them by adding predicted deltas in the following frames. Due to the dense interplay between the teacher and student, happening at every block of the network, delta distillation effectively aggregates the features across frames without any explicit motion alignment.

Delta distillation has major differences to the common knowledge distillation setting, where the student learns to regress the teacher features as in Fig. 1 (left). Instead of *distilling the features*, delta distillation aims for *distilling the temporal changes in the features* as illustrated in Fig. 1 (right). Intuitively, instead of learning the feature space embedding of their teacher, the delta distillation students learn the manifolds generated by transitions between samples, which we assume to be smooth in the case of correlated video frames. We therefore hypothesize - and verify experimentally - that delta distillation, as compared to feature distillation, allows for learning much cheaper student functions for comparable performance. Moreover, in contrast to the common knowledge distillation that relies solely on the student network to process all the test samples, delta distillation leverages both teacher and student during the inference. This trait enables delta distillation to enjoy having more parameters (coming from both models) without increasing the computational cost, as each test sample is processed by either the teacher or student network.

We summarize our contributions as follows: *i)* We propose delta distillation, a novel approach to accelerate video inference without any explicit motion compensation involved. *ii)* We elaborate various design choices to learn optimal student architectures including an end-to-end learnable architecture search. *iii)* We conduct extensive experiments on two different tasks and a wide range of models, including the most efficient architectures. Our analysis demonstrates that delta distillation sets a new state of the art in terms of accuracy vs. efficiency trade-off for semantic segmentation and object detection in videos. *iv)* We show that, as a by-product, delta distillation improves the temporal consistency of the teacher model, even though it is not explicitly optimized to do so.

2 Related work

Efficiency in deep learning Improving efficiency of neural networks is an active research area studied from multiple directions, comprising: quantization, to represent weights and activations with a low bit precision [9,16,20,31], pruning, to discard unimportant or redundant channels [6,11,23], neural architecture search, to find network designs with good accuracy vs. efficiency trade-offs [2,30], and low rank kernel decompositions [11,50]. However, for models operating on videos, redundancy among consecutive frames represents the most essential leverage to improve efficiency. In recent years, several works investigated in this direction, and they represent the closest efforts to our proposal. To avoid extracting expensive representations at every frame, feature aggregation using optical flow was explored [56,18,54]. Nevertheless, the application of such an approach on modern efficient architectures has become harder, as it requires careful model design and a proper cost balance between the feature extraction and motion alignment. Other works aim at building powerful representations over time, by aggregating features extracted by efficient models on past frames [24,14,25,26]. These recurrent methods, however, prove more effective whenever explicit motion alignment operations are carried out [14], which incur an extra computational cost. Finally, sparse computation models limit the feature extraction to sparse spatial locations that change over time, *e.g.* at a pixel [10] or at patch level [1]. However, this strategy is not robust to highly dynamic scenes, where most pixels change. Additionally, the theoretical compute gains do not always translate to latency improvement due to the inefficiency of sparse operations in most platforms.

Knowledge distillation Another well established direction to accelerate deep neural networks is knowledge distillation [12], where an efficient student network is optimized to match the output of an expansive teacher network or model ensemble. This approach was then extended by performing such an optimization within network stages, effectively distilling intermediate functions rather than the output only [36]. After these seminal works, efforts have been spent towards online distillation methods, dropping the asynchronous training regimes of teacher and students in favor of a single optimization procedure. For instance, in [51] multiple networks learn collaboratively without any teacher, and more recent works formalize the latter as ensemble of multiple students [21,8,47]. However, all these approaches do not specifically target video use-cases, and therefore transfer knowledge between models without explicitly distilling any temporal dynamic. Differently, the proposed delta distillation directly operates on temporal changes of features and, as demonstrated by experiments, allows for much cheaper student models for comparable quality.

Furthermore, some recent works adapt the general framework to specific recognition tasks, for instance by selecting specific spatial locations for teacher-student distillation in for anchor-based object detection [5,45]. We hereby remark that our approach is task-agnostic and can be applied to any video task.

3 Delta Distillation

We start with a teacher $\mathcal{F}$ as a *spatial* network generating accurate representations for a given downstream task, *e.g.* HRNet [44] or FasterRCNN [34]. Our goal is to distill this model into a more efficient *spatio-temporal* equivalent. We first break $\mathcal{F}$ down into a composition of L parametric blocks, as:

$$\mathcal{F} = f^L \circ \cdots \circ f^2 \circ f^1.$$

We describe delta distillation for a single block, however highlighting that we carry out the procedure in all blocks within a given network. The l-th block f^l takes the form:

$$\mathbf{z}^l = f^l_{\theta_l}(\mathbf{x}^l),$$

where $\mathbf{x}^l$ and $\mathbf{z}^l$ denote the input and output of the block respectively, and θ_l describes its learnable parameters. To simplify the notation, we will omit the block index l as we focus on a single block ($\mathbf{z} = f_\theta(\mathbf{x})$), and will reintroduce it in Sec. 3.3, where we define the overall network.

Feature Distillation Feature distillation [36] treats every block f_θ as a teacher block, providing target feature maps to supervise a student block g_ϕ, parametrized by ϕ, typically designed to be much cheaper. For instance, a distillation objective optimizes the expected ℓ_2 norm of the error between f_θ and g_ϕ:

$$\mathcal{L}_d(\mathbf{x};\phi) = \mathbb{E}_\mathbf{x}\left[\left\|f_\theta(\mathbf{x}) - g_\phi(\mathbf{x})\right\|_2\right]. \tag{1}$$

Delta Distillation Given a sequence of inputs $\mathbf{x}_t$, the output of a block f_θ at a time-step t can be written as:

$$\mathbf{z}_t = \mathbf{z}_{t-1} + \Delta\mathbf{z}_t,$$

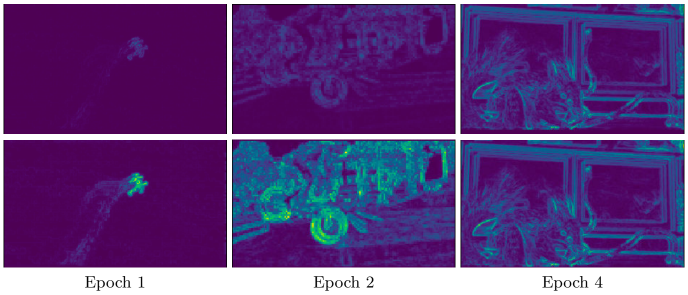

Fig. 2: The student deltas $\varDelta\tilde{\mathbf{z}}_t$ (top) vs. teacher deltas $\varDelta\mathbf{z}_t$ (bottom). Thorough training the student learns to approximate the deltas from its teacher.

where $\varDelta\mathbf{z}_t$ represents how the output of the teacher changes over time. Considering the correlation between the consecutive samples in a video, we hypothesize that $\varDelta\mathbf{z}_t$, being a transition function in the feature manifold, has a lower rank compared to the mapping f_θ. For example, in the extreme case of identical frames, $\varDelta\mathbf{z}_t$ will be of rank 0. We argue that $\varDelta\mathbf{z}_t$ can be distilled more effectively than $\mathbf{z}_t$ by a student with the same number of parameters, as verified by our experiments. In delta distillation, the student approximates the deltas given the current and previous frames as:

$$\varDelta\tilde{\mathbf{z}}_t \approx g_\phi(\mathbf{x}_t, \mathbf{x}_{t-1}).$$

This idea represents the core of our proposal that shifts the perspective from distilling the function, in Eq. 1, to distilling its temporal changes as:

$$\mathcal{L}_{dd}(\mathbf{x}_t, \mathbf{x}_{t-1}; \phi) = \mathbb{E}_{\mathbf{x}_t, \mathbf{z}_t}\left[\left\|\varDelta\mathbf{z}_t - g_\phi(\mathbf{x}_t, \mathbf{x}_{t-1})\right\|_2\right]. \tag{2}$$

By optimizing this objective, the student deltas $\varDelta\tilde{\mathbf{z}}_t$ converge to the teacher deltas $\varDelta\mathbf{z}_t$ as shown in Fig. 2.

3.1 Student architectures

As in any distillation approach, delta distillation admits diverse architectural choices [7], in terms of *i)* granularity at which it should operate, *e.g.* distilling single convolutions vs distilling residual blocks or branches and *ii)* possible architectures for the student block. In our implementation, we consider the following:

Linear blocks that define the teachers and students at every convolution. In this case, we only feed the student with the input residual $\varDelta\mathbf{x}_t = \mathbf{x}_t - \mathbf{x}_{t-1}$:

$$g_\phi(\mathbf{x}_t, \mathbf{x}_{t-1}) = g_\phi^{conv}(\varDelta\mathbf{x}_t).$$

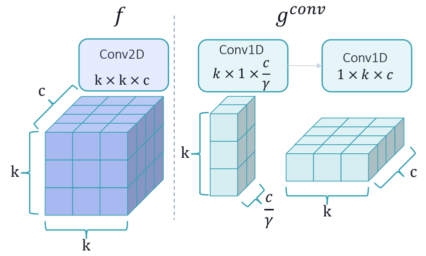

Fig. 3: Student architecture for a linear block, obtained by decomposing the teacher kernel as a sequence of two 1D kernels with $\gamma\times$ less number of intermediate channels. For simplicity, we only visualize the output channels.

This choice is motivated by the Taylor approximation of f_θ where, if the function f_θ is linear, only the first order term is non-zero, and the derivative $\nabla f_\theta(\mathbf{x}_{t-1})$ is a constant:

$$\Delta\mathbf{z}_t = \nabla f_\theta(\mathbf{x}_{t-1})\Delta\mathbf{x}_t + \frac{1}{2}\nabla^2 f_\theta(\mathbf{x}_{t-1})\Delta\mathbf{x}_t^2 + \dots \tag{3}$$

As the student architecture, we rely on a spatial kernel factorization as depicted in Fig. 3. Similar to the spatial SVD [17], we decompose each 2D kernel as a sequence of two 1D kernels while reducing the number of intermediate channels by a compression factor γ.

Non-linear blocks that define the teachers and students at a coarser granularity as a sequence of residual blocks. In this case, according to the Taylor approximation in Eq. 3, we parameterize the student as a function of both the previous input $\mathbf{x}_{t-1}$ and input residual $\Delta\mathbf{x}_t$ concatenated along the channel dimension:

$$g_\phi(\mathbf{x}_t, \mathbf{x}_{t-1}) = g_\phi^{block}(\mathbf{x}_{t-1}, \Delta\mathbf{x}_t).$$

As the student architecture, we envision two strategies: *i) channel reduction*, where the student mirrors the teacher but with fewer channels: we add two pointwise convolution to the block as first and last layer to shrink and expand the channels respectively by a fixed factor. *ii) spatial reduction*, where the student resembles the teacher but operates on a smaller resolution: we add a strided pointwise convolution and a pixel shuffle up-sampling to the beginning and the end of each block, respectively.

3.2 Student architecture search

Different layers within a network may be compressible to different extents. We empirically found that there might exist a few critical layers [1] that are not amenable to distillation: if compressed, they hinder the model performance.

[1] as an example, transition layers in HRNets [44]

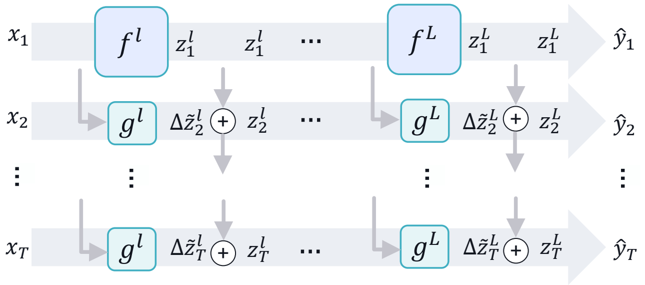

Fig. 4: Delta distillation at inference. The teacher computes the features for the key-frame, while the student updates the features by predicting deltas over time.

Therefore, instead of committing to the same student architecture for all the blocks, we introduce two candidates: *i)* non-compressed architecture, identical to the teacher, preferred for hard-to-compress blocks. *ii)* compressed architecture using the techniques introduced above for linear and non-linear blocks.

To find optimal student architectures, we introduce a learnable parameter $\psi \in \mathbb{R}^2$ per block, that is learned jointly with the student parameters. During training, architectures are sampled from a categorical distribution q_ψ over the two candidate models, obtained by feeding ψ to a softmax layer. We rely on the Gumbel-softmax [19,28] reparametrization to estimate gradients. To encourage the search algorithm to opt for the compressed architecture, we introduce a sparsity regularization objective as:

$$\mathcal{L}_s(;\psi) = \mathbb{E}_{g_\phi \sim q_\psi}\left[\text{FLOPs}(g_\phi)\right], \tag{4}$$

where FLOPs is the complexity of the sampled architecture g_ϕ in terms of the number of floating point operations. This loss encourages the model to select the more efficient student architecture as much as possible. If not regularized, ψ would converge to the trivial solution of selecting the non-compressed architecture as it has a higher capacity and a better distillation performance.

3.3 Training and Inference

Given a trained teacher network $\mathcal{F}$, comprising the teacher blocks, and a training set of labeled clips with T frames $\{(\mathbf{x}_{1:T}, \mathbf{y}_{1:T})\}$, we train the delta distillation by optimizing the following objective:

$$\mathbb{E}_{\mathbf{x}_{1:T},\mathbf{y}_{1:T}}\left[\frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{L}_t + \alpha\sum_{l=1}^{L}\mathcal{L}_{dd}^l + \beta\sum_{l=1}^{L}\mathcal{L}_s^l\right)\right] \tag{5}$$

where $\mathcal{L}_{dd}^l$ and $\mathcal{L}_s^l$ denote the delta distillation and the sparsity regularization losses, computed for the l-th block, as defined in Eq. 2 and 4 respectively. The hyper-parameters α and β balance the contribution of the losses to learn a model yielding the best accuracy vs. efficiency trade-off.

$\mathcal{L}_t(\mathbf{x}_t, \mathbf{y}_t; \Theta, \Phi)$ is the task loss used to train the network $\mathcal{F}$, where $\Theta = \{\theta_1 \dots \theta_L\}$ and $\Phi = \{\phi_1 \dots \phi_L\}$ denote all the learnable parameters in the teacher and student, respectively. Although we envision an unsupervised training, by excluding $\mathcal{L}_t$ from the objective, we conduct all our experiments while including $\mathcal{L}_t$ as it yields a better performance.

As illustrated in Fig. 4, delta distillation calls both teachers and students during the inference. More specifically, the key-frame is passed through the teacher to compute initial features. Then, subsequent frames are fed to the students to update the previous features by predicting their deltas across the frames. This is different from typical knowledge distillation settings, such as feature distillation, where the teacher is only used during training and not during inference. Pseudo code for training and inference is provided in the supplementary material.

3.4 Temporal Consistency

Another aspect of increasing importance for video streaming tasks is temporal consistency of model responses [22,27,33]. Indeed, flickering predictions can make decision-making difficult in critical scenarios. Although our main motivation is to improve the model efficiency, we argue that delta distillation can also improve the temporal consistency in predictions (as verified experimentally in Sec. 4.2). As an explanation, we argue that delta distillation converts the *spatial* teacher network to a *spatio-temporal* model, as it propagates states from one timestep to the next, and explicit temporal dynamics improve the overall temporal stability, as also noted in several prior works [33,40]. Moreover, in delta distillation, the students have a regularization effect on the teacher. More specifically, the teacher should generate smooth and low-rank features so as to be learnable by a student with a limited number of parameters. This penalizes the teacher from generating hard to distill deltas, *e.g.* from representations that are inconsistent over time.

4 Experiments

We evaluate delta distillation on two video tasks: object detection and semantic segmentation, in Sec. 4.1 and 4.2, respectively. Several ablation studies are reported in Sec. 4.3. Further analysis are provided in the supplementary material.

4.1 Object Detection

Dataset and metrics We experiment with Imagenet VID [37], which contains 3862, 555, and 937 annotated snippets for training, validation, and test, respectively. All frames come with bounding boxes belonging to 30 target categories. Following the standard protocol [3,54,55,56], we augment the training snippets with still images from ImageNet DET sampled at $1:1$ ratio, and report results on the validation set. We rely on mean Average Precision (mAP) with an IoU threshold of 0.5 as the accuracy metric. To be hardware-agnostic, we report the efficiency of models in terms of number of floating point operations (FLOPs[2]).

[2] FLOPs denotes number of multiply-adds.

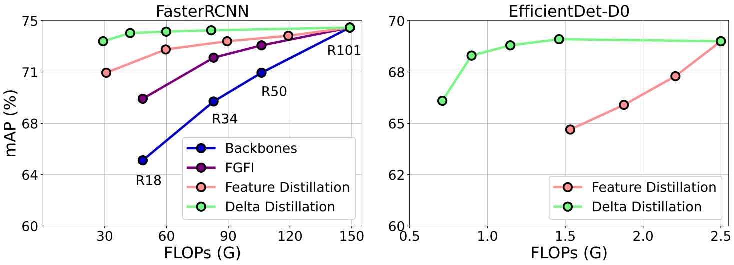

Fig. 5: Comparisons to knowledge distillations on ImageNet VID. Delta distillation outperforms the alternatives. The gap is higher for EfficientDet-D0 that is already highly optimized so is more challenging to be accelerated further.

Training details We conduct our experiments using a single-stage and a two-stage object detector, namely EfficientDet-D0 [42] and Faster-RCNN [34] with a ResNet-101 backbone. We first train the teacher networks $\mathcal{F}$ using a SGD optimizer with a learning rate of 0.01 for 7 epochs. The learning rate is reduced by a factor of 10 at epochs 2 and 5. Clips are resized to have a longer side of 512 px and 600 $\sim$ 1000 px for EfficientDet and Faster-RCNN, respectively. We train the models on four GPUs using a batch size of 16 and 4 for the EfficientDet and Faster-RCNN, respectively. Starting from the trained teacher network and randomly initialized students, we optimize the training objective, Eq. 5, setting α and β to 1000 and 10 for Faster-RCNN, and to 100 and 10 for EfficientDet. We report the experiments for distilling at every linear block though similar conclusions hold for distilling non-linear blocks as a stack of multiple layers. We use a compression ratio $\gamma = 16$, as illustrated in Fig. 3, to define the student architectures.

Evaluation details Following [56], we split each video into sequences of equal length $T = 10$. The first frame in a sequence (key-frame) is processed by the teacher while other frames are processed by the student networks. We report mAP and FLOPs averaged over all frames in the sequence. In the supplement, we report detailed analysis teacher and student cost, illustrating per-frame FLOPS as well as their capacity in terms of parameters.

Comparison to knowledge distillations We evaluate our key hypothesis, namely that $\varDelta\mathbf{z}_t$ can be distilled more effectively than $\mathbf{z}_t$, by training students using two different distillation objectives: feature vs. delta distillation as defined in Eq. 1 and 2. Additionally, we compare to Fine Grained Feature Imitation (FGFI) [45], as a knowledge distillation devised for object detection that distills the features around the object anchor locations from a ResNet-101 backbone using a lighter student backbone (R50, R34, and R18). We omit applying FGFI on EfficientDet-D0 as there is no more efficient backbone available to serve as the student for this detector. We study how the model accuracy responds to reducing the computational cost *i.e.* by using a cheaper backbone for FGFI, and a higher sparsity regularization β for feature and delta distillation.

As reported in Fig. 5, our results demonstrate that FGFI is effective when compared to the optimization of the student backbones from scratch (blue plot), yet it underperforms with respect to the feature distillation that shares all components of our model, *e.g.* SVD kernel decomposition and student architecture search. Additionally, the results verify that feature distillation has a limited effectiveness in reducing the model complexity especially for a highly optimized architecture such as EfficientDet-D0. However, by leveraging temporal redundancy, delta distillation reduces the compute cost by 2× with a negligible drop in the accuracy. For EfficientDet-D0, it reduces the GFLOPs from 2.5 to 1.14 with a negligible mAP drop, from 69.0 to 68.8.

Model	**FLOPs (G)**	**mAP**
TAFM [25]	1.18	64.1
PatchWork [1]	0.97	57.4
Skip-Conv [10]	0.78	62.9
PatchNet [29]	0.73	58.9
Mobile-DFF [54]	0.71	62.8
EfficientDet-D0 [42]	2.50	69.0
+ **Delta Distillation**	**0.71**	66.1
DFF [56]	34.9	72.5
PatchNet [29]	34.2	73.1
Faster-RCNN [34]	149.1	74.5
+ **Delta Distillation**	**29.2**	73.5

Table 1: Comparisons with efficient video object detectors on ImageNet VID, for some light (top) and heavy (bottom) networks. Delta distillation achieves the lowest FLOPs while being more accurate than others.

Comparison to state of the art We compare to the state of the art video object detectors that leverage temporal redundancies to speed up the inference, as categorized into: *i)* feature warping, *i.e.* DFF [56] and Mobile-DFF [54], that bypass feature computation by warping the previous features using optical flow. *ii)* feature aggregation, *i.e.* TAFM [25], that instead of extracting expensive features at every frame, it relies on an aggregation of cheaper features over time. *iii)* feature sparsification, *i.e.* PatchWork [1] and Skip-Conv [10], that restrict the feature computation only to a sparse set of regions or pixels that change significantly across frames. *vi)* detection by tracking, *i.e.* PatchNet [29], that interleaves running an expensive detector with cheap object trackers. As reported in Tab. 1, we divide object detectors into two groups: light detectors (top) developed for mobile devices, using MobileNet-v2 [38] and EfficientNet [41] backbones, and expensive detectors (bottom), using ResNet-101 backbone.

Our results show that delta distillation achieves the lowest FLOPs while being more accurate than both the light and expensive detectors. Moreover, despite the differences between FasterRCNN-R101 and EfficientDet-D0 in terms of architectures and computational bottlenecks, delta distillation consistently reduces their FLOPs without any architectural adaption, *i.e.* from 2.5 to 0.71 and from 149.1 to 29.2 respectively. This is not the case for feature warping methods, that require a careful architecture design to find a right cost and accuracy balance between feature extractor and optical flow model.

4.2 Semantic Segmentation

Dataset and metrics We conduct experiments on the Cityscapes dataset [4] that is partitioned into 2975, 500 and 1525 snippets as training, validation, and test splits respectively. We rely on the standard training split to train and report results on the validation set. The dataset provides pixel-level annotations into 19 classes for one frame per snippet. We extract the remaining per-frame psuedo-annotations, which are required to train video models, by applying an off the shelf segmentation network [43] on unannotated frames in the training set. We evaluate the accuracy using mean intersection-over-union (mIoU). As a mean to compare computational cost, we rely again on FLOPs also reporting latency measurements in the supplementary materials.

Training details We conduct experiments using two state-of-the-art segmentation model: HRNet [44] and DDRNet [13]. We follow the same training protocol as in [13,44]: models are initialized with ImageNet weights and trained using a SGD optimizer with a learning rate of 0.01, that is reduced using a polynomial decay policy with a power of 0.9. Training runs for 484 epochs using a batch size of 12 on four GPUs and SyncBN. The models are trained on random crops of 512×1024 and tested on 1024×2048. For DDRNet [13], we use online hard example mining [39] as in [13]. Starting from the trained teacher and randomly initialized students, we optimize Eq. 5, using α and β set to 1 and 0.5 respectively. We rely on linear blocks with a compression ratio γ of 4 and 8 for DDRNets and HRNet, as illustrated in Fig. 3, to define the student architectures.

Model	**FLOPs (G)**	**mIoU**
FANet-34 [15]	65.0	76.3
BiseNet-v1-18 [49]	55.3	74.8
FANet-18 [15]	49.0	75.0
LedNet [46]	45.8	71.5
ICNet [52]	28.2	67.7
FasterSeg [2]	28.2	73.1
ERFNet [35]	27.7	70.0
SwiftNet-18 [32]	26.0	70.2
BiseNet-v2 [48]	21.1	73.4
TDNet-PSPNet [14]	541.0	79.9
DFF [56]	109.3	69.2
TDNet-BiseNet [14]	101.3	76.4
Skip-Conv [10]	29.0	75.5
DDRNet-39 [13]	282.0	79.5
+ **Delta Distillation**	140.0	79.9
DDRNet-23 [13]	143.7	78.7
+ **Delta Distillation**	71.8	78.9
HRNet-w18-small [44]	77.9	76.1
+ **Delta Distillation**	34.1	75.7
DDRNet-23-slim [13]	36.6	76.1
+ **Delta Distillation**	**17.9**	76.2

Table 2: Comparison with efficient image (top) and video (middle) based models on Cityscapes validation set.

Evaluation details Each video is split into sequences of equal length T. We fix the sequence length $T = 3$ as it yields the best trade-off between accuracy vs. efficiency though the model is relatively robust to longer sequences as reported in Fig 6. Since the videos in Cityscapes have temporally sparse annotation, we repeat evaluations by opting each annotated frame in all possible positions within the sequence and report the averaged mIoU following [14]. Similarly we report FLOPs averaged over all frames in the sequence counting for the both teacher and student costs.

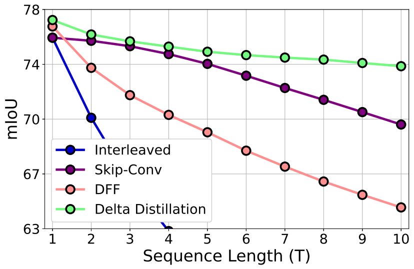

Fig. 6: Robustness to temporal variations. By increasing the distance to the key-frame, delta distillation retains better performances of prior methods.

Comparison to state of the art We first assess the effect of delta distillation on different segmentation backbones with varying computational cost: HRNet-W18-small, DDRNet-23-slim, DDRNet-23, and DDRNet-39. As reported in Tab. 2 (bottom), delta distillation consistently reduces computational cost by a factor of $\sim 2\times$ for all the backbones, with no or small drop in accuracy.

Tab. 2 compares delta distillation with efficient image[3] (top) and video (middle) based semantic segmentation models. The results show that delta distillation outperforms all the image-based models while being at the same time more efficient. Compared to BiseNet-v2 [48], the most efficient frame-based model, delta distillation achieves a mIoU of 76.2 vs. 73.4, with a lower cost of 17.9 vs 21.1 GFLOPs. Moreover, delta distillation achieves a more favorable accuracy vs. efficiency trade-off compared to the other video-based models. For instance, at the same mIoU of 79.9, delta distillation is 3.8× more efficient (541 vs 140 GFLOPs) than a TDNet [14] with PSPNet backbone. Finally, the delta distilled DDRNet-23-slim model significantly outperforms DFF [56] and Skip-conv [10], both in terms of accuracy and efficiency. Per-class analysis, reported in the supp. material, highlights that accuracies are retained on both static and dynamic classes.

Robustness to temporal variations. We evaluate the impact of sequence length (T) on the performance of delta distillation as reported in Fig 6. The longer the sequence is, the less frequently features are refreshed by running the teacher model. This drops the accuracy as the student model has a limited capacity compared to the teacher. However, as highlighted in the results, the performance drop is smaller for delta distillation, especially on longer sequences, compared to competing methods. As a lower bound, we include an interleaved baseline that copies the predictions from the key-frame to consecutive frames. These findings verify the effectiveness of delta distillation in handling long range temporal variations.

Choice of backbone. In Tab. 2 we evaluate video-based models using the backbones originally used by authors *i.e.* ResNet-101 for DFF and ResNet-50 for TDNet-PSPNet. Since these backbones are too expensive to run in real-time we

[3] We limit our comparisons to efficient models with less than 100 GFLOPs.

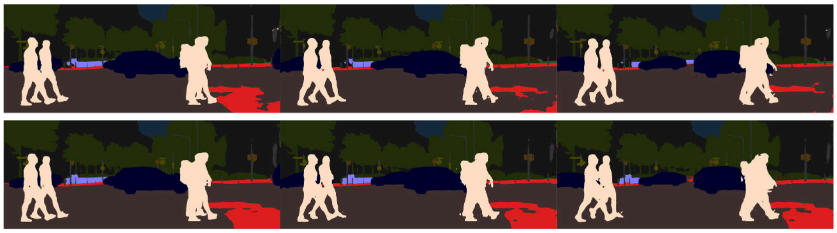

Fig. 7: Example predictions on Cityscapes, with DDRNet23 on top and DDRNet23 + Delta Distillation below. The latter shows more consistency over the baseline model. More examples are reported in the supplementary material.

rely on more efficient backbones, *e.g.* DDRNet, that are arguably more challenging to be further accelerated. As a further analysis, we implement several video-based models using DDRNet-23-slim backbone as reported in Fig 6. The figure confirms the superiority of delta distillation as compared to alternative video efficiency models all using the same backbone.

Temporal consistency As motivated in Sec. 3.4, we evaluate the effectiveness of delta distillation at improving the temporal consistency (TC) on the validation set. We follow the TC metric introduced in [27]: in a nutshell, it computes the average IoU among model predictions across successive frames, after motion compensation by optical flow warping. Our results, presented in Tab. 3, suggests several insights: First, the training procedure of delta distillation effectively regularizes the teacher model towards more temporally consistent predictions, even when run on different frames independently (T), as testified by the improvement of 1.8 points with respect to the baseline. Furthermore, by using the inference procedure comprising of both teacher and student (T+S), the TC metric further improves. Finally, we compare to the temporal consistency reported by ETC [27] which explicitly includes a temporal consistency loss in the optimization. Fig. 7 shows qualitative results of DDRNet23 both with and without delta distillation.

4.3 Ablation studies

Student architecture We analyze the effect of different choices of architecture, following the designs described in Sec. 3.1. In Tab. 4, we show the results of

Table 3: Temporal consistency (TC) measured on Cityscapes.

Model		**TC**	**Δ TC**
PSPNET18 [53]		83.3	-
+ ETC [27]		84.6	+1.3
DDRNet23 [13]		82.6	-
+ **Delta Distillation**	(T)	84.4	+1.8
+ **Delta Distillation**	(T+S)	**85.2**	**+2.6**

Table 4: Ablation on student architecture designs.

Student Architecture	**FLOPs (G)**	**mIoU**
DDRNet23	143.7	78.7
+ Linear	71.8	78.9
+ Non-Linear (Spatial)	110.3	78.4
+ Non-Linear (Channel)	84.3	79.0
+ Non-Linear (Channel + Spatial)	96.13	78.7

training *Linear* vs *Non-Linear* blocks as well as the *channel* vs *spatial* reduction. First, we note that delta distillation, regardless of architectural choice, has lower GFLOPs than the base Image Model. This trait is desirable, as it suggests the computational savings are not bound to a unique student architecture. However, we do see some notable differences within architecture differences themselves. By comparing both *Linear* and *Non-Linear* - *Channel*, we appreciate that the former enjoys a slightly smaller computational footprint (71.8 vs 84.3 GFLOPs) with similar mIoU. We hypothesize this difference could be due to the fact that linear functions are easier to distill. Finally, when we compare the *channel* and *spatial* variants, we observe the latter architecture performs slightly worse.

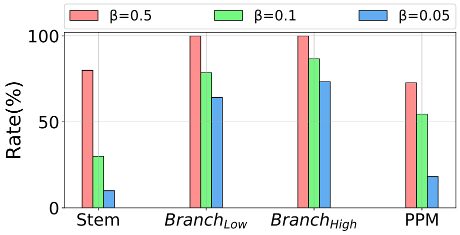

Fig. 8: Ablation on student architecture search. We report the proportion of the compressed layers per stage.

Student architecture search We study how the architecture search, (Sec. 3.1), selects the student architectures. For this purpose, we observe the effect of gradually increasing the sparsity coefficient, β from Eq. 5. We report the proportion of compressed blocks for DDRNet23-slim grouped by its four main stages: stem convolutions at the entry, low and high-resolution branches to process the input in parallel, and a pyramid pooling module (PPM) at the end to fuse feature maps across resolutions. We normalize the number of the compressed blocks per stage and report it as the compression rate in Fig. 8. We note that a higher β indeed translates to a higher proportion of compressed blocks across all stages. Moreover, as we reduce β, the search algorithm opts for selecting less compressed blocks in the stem. We hypothesize that since all the layers follow the stem, this layer represents a single point of failure: at this stage, a non effective distillation might hinder the whole model's performance. We observe a similar pattern for the PPM stage, likely due to its closeness to the output, thus having a bigger impact on the performance.

5 Conclusion

We proposed delta distillation, a novel method for efficient video processing exploiting the temporal redundancy of frames. Our proposal optimizes the regression, by means of cheap student models, of temporal variations in feature maps computed by an expensive teacher network. During inference, the teacher provides the initial representations for the first frame; such feature maps are then iteratively refined for the next frames by adding deltas estimated by students, the latter operating at a low computational cost. We show through extensive experiments that delta distillation outperforms feature distillation for comparable student architectures, and delivers state-of-the-art results for efficient video segmentation and object detection.

References

1. Chai, Y.: Patchwork: A patch-wise attention network for efficient object detection and segmentation in video streams. In: ICCV (2019) 3, 10
2. Chen, W., Gong, X., Liu, X., Zhang, Q., Li, Y., Wang, Z.: Fasterseg: Searching for faster real-time semantic segmentation. ICLR (2020) 1, 3, 11
3. Chen, Y., Cao, Y., Hu, H., Wang, L.: Memory enhanced global-local aggregation for video object detection. In: CVPR (2020) 8
4. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016) 11
5. Dai, X., Jiang, Z., Wu, Z., Bao, Y., Wang, Z., Liu, S., Zhou, E.: General instance distillation for object detection. In: CVPR (2021) 4
6. Denil, M., Shakibi, B., Dinh, L., Ranzato, M., de Freitas, N.: Predicting parameters in deep learning. In: NeurIPS (2013) 3
7. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. IJCV (2021) 2, 5
8. Guo, Q., Wang, X., Wu, Y., Yu, Z., Liang, D., Hu, X., Luo, P.: Online knowledge distillation via collaborative learning. In: CVPR (2020) 4
9. Gupta, S., Agrawal, A., Gopalakrishnan, K., Narayanan, P.: Deep learning with limited numerical precision. In: ICML (2015) 3
10. Habibian, A., Abati, D., Cohen, T.S., Bejnordi, B.E.: Skip-convolutions for efficient video processing. In: CVPR (2021) 3, 10, 11, 12
11. He, Y., Zhang, X., Sun, J.: Channel pruning for accelerating very deep neural networks. In: ICCV (2017) 3
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015) 2, 3
13. Hong, Y., Pan, H., Sun, W., Jia, Y., et al.: Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes. arXiv preprint arXiv:2101.06085 (2021) 2, 11, 13
14. Hu, P., Caba, F., Wang, O., Lin, Z., Sclaroff, S., Perazzi, F.: Temporally distributed networks for fast video semantic segmentation. CVPR (2020) 3, 11, 12
15. Hu, P., Perazzi, F., Heilbron, F.C., Wang, O., Lin, Z., Saenko, K., Sclaroff, S.: Real-time semantic segmentation with fast attention. ICRA (2020) 1, 11
16. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: CVPR (2018) 3
17. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up convolutional neural networks with low rank expansions. BMVC (2014) 6
18. Jain, S., Wang, X., Gonzalez, J.E.: Accel: A corrective fusion network for efficient semantic segmentation on video. In: CVPR (2019) 1, 2, 3
19. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. ICLR (2017) 7
20. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:1806.08342 (2018) 3
21. Lan, X., Zhu, X., Gong, S., et al.: Knowledge distillation by on-the-fly native ensemble. NeurIPS (2018) 4
22. Lei, C., Xing, Y., Chen, Q.: Blind video temporal consistency via deep video prior. NeurIPS (2020) 8

23. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710 (2017) 1, 3
24. Li, Y., Shi, J., Lin, D.: Low-latency video semantic segmentation. In: CVPR (2018) 1, 3
25. Liu, M., Zhu, M.: Mobile video object detection with temporally-aware feature maps. In: CVPR (2018) 1, 3, 10
26. Liu, M., Zhu, M., White, M., Li, Y., Kalenichenko, D.: Looking fast and slow: Memory-guided mobile video object detection. arXiv preprint arXiv:1903.10172 (2019) 1, 3
27. Liu, Y., Shen, C., Yu, C., Wang, J.: Efficient semantic video segmentation with per-frame inference. ECCV (2020) 8, 13
28. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: A continuous relaxation of discrete random variables. ICLR (2017) 7
29. Mao, H., Zhu, S., Han, S., Dally, W.J.: Patchnet–short-range template matching for efficient video processing. arXiv preprint arXiv:2103.07371 (2021) 10
30. Moons, B., Noorzad, P., Skliar, A., Mariani, G., Mehta, D., Lott, C., Blankevoort, T.: Distilling optimal neural networks: Rapid search in diverse spaces. In: ICCV (2021) 3
31. Nagel, M., van Baalen, M., Blankevoort, T., Welling, M.: Data-free quantization through weight equalization and bias correction. ICCV (2019) 3
32. Orsic, M., Kreso, I., Bevandic, P., Segvic, S.: In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In: CVPR (2019) 11
33. Rebol, M., Knöbelreiter, P.: Frame-to-frame consistent semantic segmentation. In: Joint Austrian Computer Vision And Robotics Workshop (ACVRW) (2020) 8
34. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. NeurIPS (2015) 4, 9, 10
35. Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R.: Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. IEEE Transactions on Intelligent Transportation Systems (2017) 11
36. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. ICLR (2015) 4
37. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV (2015) 8
38. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR (2018) 10
39. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR (2016) 11
40. Sibechi, R., Booij, O., Baka, N., Bloem, P.: Exploiting temporality for semi-supervised video segmentation. In: ICCV Workshops (2019) 8
41. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: ICML (2019) 10
42. Tan, M., Pang, R., Le, Q.V.: Efficientdet: Scalable and efficient object detection. In: CVPR (2020) 1, 2, 9, 10
43. Tao, A., Sapra, K., Catanzaro, B.: Hierarchical multi-scale attention for semantic segmentation. arXiv preprint arXiv:2005.10821 (2020) 11
44. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., Xiao, B.: Deep high-resolution representation learning for visual recognition. TPAMI (2019) 1, 2, 4, 6, 11

45. Wang, T., Yuan, L., Zhang, X., Feng, J.: Distilling object detectors with fine-grained feature imitation. In: CVPR (2019) 4, 9
46. Wang, Y., Zhou, Q., Liu, J., Xiong, J., Gao, G., Wu, X., Latecki, L.J.: Lednet: A lightweight encoder-decoder network for real-time semantic segmentation. In: ICIP (2019) 11
47. Wu, G., Gong, S.: Peer collaborative learning for online knowledge distillation. In: AAAI (2021) 4
48. Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., Sang, N.: Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. IJCV (2021) 11, 12
49. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: ECCV (2018) 11
50. Zhang, X., Zou, J., He, K., Sun, J.: Accelerating very deep convolutional networks for classification and detection. TPAMI (2016) 1, 3
51. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: CVPR (2018) 4
52. Zhao, H., Qi, X., Shen, X., Shi, J., Jia, J.: Icnet for real-time semantic segmentation on high-resolution images. In: ECCV (2018) 11
53. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR (2017) 13
54. Zhu, X., Dai, J., Zhu, X., Wei, Y., Yuan, L.: Towards high performance video object detection for mobiles. arXiv preprint arXiv:1804.05830 (2018) 1, 2, 3, 8, 10
55. Zhu, X., Wang, Y., Dai, J., Yuan, L., Wei, Y.: Flow-guided feature aggregation for video object detection. In: ICCV (2017) 8
56. Zhu, X., Xiong, Y., Dai, J., Yuan, L., Wei, Y.: Deep feature flow for video recognition. In: CVPR (2017) 1, 2, 3, 8, 9, 10, 11, 12