Skip-Convolutions for Efficient Video Processing

Amirhossein Habibian Davide Abati Taco S. Cohen Babak Ehteshami Bejnordi
Qualcomm AI Research*
{habibian,dabati,tacos,behtesha}@qti.qualcomm.com

Abstract

We propose Skip-Convolutions to leverage the large amount of redundancies in video streams and save computations. Each video is represented as a series of changes across frames and network activations, denoted as residuals. We reformulate standard convolution to be efficiently computed on residual frames: each layer is coupled with a binary gate deciding whether a residual is important to the model prediction, e.g. *foreground regions, or it can be safely skipped,* e.g. *background regions. These gates can either be implemented as an efficient network trained jointly with convolution kernels, or can simply skip the residuals based on their magnitude. Gating functions can also incorporate block-wise sparsity structures, as required for efficient implementation on hardware platforms. By replacing all convolutions with Skip-Convolutions in two state-of-the-art architectures, namely EfficientDet and HRNet, we reduce their computational cost consistently by a factor of* $3 \sim 4\times$ *for two different tasks, without any accuracy drop. Extensive comparisons with existing model compression, as well as image and video efficiency methods demonstrate that Skip-Convolutions set a new state-of-the-art by effectively exploiting the temporal redundancies in videos.*

1. Introduction

Is a video a sequence of still images or a continuous series of changes? We see the world by sensing changes, and process information whenever the accumulated differences in our neurons exceed some threshold. This trait has inspired many efforts to develop neuromorphic sensors and processing algorithms, such as event-based cameras [40] and spiking neural networks [12]. Despite their efficiency for video processing, spiking nets have not been as successful as conventional models, mostly due to the lack of efficient training algorithms. There have been several works on mapping spiking nets to conventional networks, but these works have been mostly limited to shallow architectures and simple problems, such as digit classification [66, 36, 35]. Representing videos by changes through residual frames is common in video compression codecs, such as HEVC [46], because residual frames normally have less information entropy and therefore require fewer bits to be compressed.

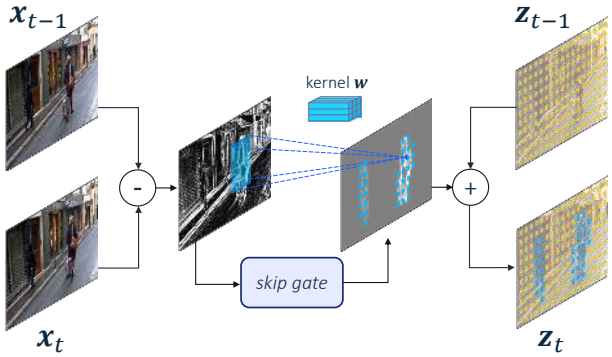


Figure 1: Skip-Convolution illustration for the input layer. Convolutions are computed only on a few locations in the residual features determined by a gate function (blue dots). In other locations, output features are copied from the previous time step (yellow dots). Frames taken from [65].

For stream processing applications, that require spatially dense predictions for each input frame, deep convolutional networks still process a sequence of still images as input. Each frame is processed entirely by sliding convolutional filters all over the frame, layer by layer. As a result, the overall computational cost grows linearly with the number of input frames, even though there might be not much new information in the subsequent frames. This inherent inefficiency prohibits using accurate but expensive networks for real-time tasks, such as object detection and pose estimation, on video streams.

This paper proposes *Skip-Convolutions*, in short, Skip-Convs, to speed up any convolutional network for inference on video streams. Instead of considering a video as a sequence of still images, we represent it as a series of changes across frames and network activations, denoted as residual frames. We reformulate standard convolution to be efficiently computed over such residual frames by limiting the computation only to the regions with significant changes while skipping the others. Each convolutional layer is cou-

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

pled with a gating function learned to distinguish between the residuals that are important for the model accuracy and background regions that can be safely ignored (Fig. 1).

By applying the convolution kernel on sparse locations, Skip-Convs allow to adjust efficiency depending on the input, in line with recent studies on conditional computation in images [25, 7, 42, 51, 53]. However, we hereby argue that distinguishing the important and non-important regions is more challenging in still images. Indeed, residual frames provide a strong prior on the relevant regions, easing the design of effective gating functions. As a result, Skip-Convs achieve a much higher cost reduction in videos ($300 \sim 400\%$), compared to what has been previously reported for images ($15 \sim 60\%$ in [51], $27 \sim 41\%$ in [53]).

To summarize, the main contributions of this work are: *i)* a simple reformulation of convolution, which computes features on highly sparse residuals instead of dense video frames. *ii, iii)* Two gating functions, Norm gate and Gumbel gate, to effectively decide whether to process or skip each location. Norm gates do not have any trainable parameter, thus can be easily plugged into any trained network obviating the need for further fine-tuning. On the contrary, Gumbel gates are trainable: they are learned jointly with the backbone model with the Gumbel reparametrization [20, 32], and allow to achieve even more efficiency. We extend these gates to generate structured sparsity as required for efficient hardware implementations. *iv)* A general formulation of Skip-Conv, which extends the idea to a broader range of transformations and operations. *v)* extensive experiments on two different tasks and state-of-the-art network architectures, showing a consistent reduction in cost by a factor of $3 \sim 4\times$, without any accuracy drop.

2. Related work

Efficient video models Exploiting temporal redundancy is the key to develop efficient video models. A common strategy is feature propagation [44, 69, 26, 68], which computes the expensive backbone features only on key-frames. Subsequent frames then adapt the backbone features from key-frames directly [44] or after spatial alignments via optical flow [69, 19], dynamic filters [26, 34], or self-attention [15]. Similarly, Skip-Conv also propagates features from the previous frame, however: *i)* feature propagation models depend on the alignment step, which is potentially expensive, *e.g.* for accurate optical flow extraction. *ii)* These methods propagate the feature only at a single layer, whereas Skip-Conv propagates features at every layer. *iii)* Skip-Conv selectively decides whether to propagate or compute at the pixel level, rather than for the whole frame. *iv)* differently from feature propagation methods that imply architectural adjustments, Skip-Conv does not involve any modifications to the original network.

Another strategy is to interleave deep and shallow backbones between consecutive frames [19, 29, 34]. The deep features, extracted only on key-frames, are fused with shallow features extracted on other frames using concatenation [19], recurrent networks [29], or more sophisticated dynamic kernel distillation [34]. This strategy usually leads to an accuracy gap between key-frames and other frames.

Several works aim for efficient video classification by developing faster alternatives for 3D convolutions, such as temporal shift modules [27] and 2+1D convolutions [49], neural architecture search [39, 8], or adaptive frame sampling [4, 55, 33]. These methods are mostly suitable for global prediction tasks where a single prediction is made for the whole clip. Differently, we target stream processing tasks, such as pose estimation and object detection, where a spatially dense prediction is required for every frame.

Efficient image models The reduction of parameter redundancies, *e.g.* in channels and layers, is a fundamental aspect for obtaining efficient image models. Model compression methods [24], such as low-rank tensor decomposition [18, 67], channel pruning [13, 30], neural architecture search [47, 48], and knowledge distillation [14, 43], effectively reduce the memory and computational cost of any network. Instead of exploiting weight redundancies, as addressed by model compression, Skip-Conv leverages temporal redundancies in activations. As verified by our experiments, these are complementary and can be combined to further reduce the computational cost.

Conditional computation has recently shown great promise to develop efficient models for images [2]. It enables the model to dynamically adapt the computational graph per input to skip processing unnecessary branches [16], layers [50], channels [1, 11], or non-important spatial locations such as background [9, 25, 7, 42, 51, 53]. However, distinguishing the important vs. non-important regions is difficult in images. Skip-Conv leverages residual frames as a strong prior to identify important regions in feature maps based on their changes, outperforming their image counterparts by a large margin as validated by our experiments.

3. Skip Convolutions

Instead of treating a video as a sequence of still images, we represent it as a series of residual frames defined both for the input frames and for intermediate feature maps. In section 3.1, we reformulate the standard convolution to be efficiently computed on residuals. Section 3.2 proposes several gating functions to decide whether to process or skip each location in residual frames. Gating functions are crucial to reduce the computation without losing much accuracy. Finally, section 3.3 discusses how Skip-Conv can be generalized to a broader set of transformations beyond residuals as a direction for future developments.

3.1. Convolution on Residual Frames

Given a convolutional layer with a kernel $\mathbf{w} \in \mathbb{R}^{c_o \times c_i \times k_h \times k_w}$ and an input $\mathbf{x}_t \in \mathbb{R}^{c_i \times h \times w}$, the output feature map $\mathbf{z}_t \in \mathbb{R}^{c_o \times h \times w}$ is computed for each frame as[1]:

$$\mathbf{z}_t = \mathbf{w} * \mathbf{x}_t. \tag{1}$$

In Eq. 1 (and in the remainder of this section) $\mathbf{z}_t$ refers to the result before the application of a non-linear activation function. Using the distributive property of convolution as a linear function, the output can be obtained by:

$$\begin{aligned} \mathbf{z}_t &= \mathbf{w} * \mathbf{x}_{t-1} + \mathbf{w} * \mathbf{x}_t - \mathbf{w} * \mathbf{x}_{t-1} \\ &= \mathbf{z}_{t-1} + \mathbf{w} * (\mathbf{x}_t - \mathbf{x}_{t-1}) \\ &= \mathbf{z}_{t-1} + \mathbf{w} * \mathbf{r}_t, \end{aligned} \tag{2}$$

where $\mathbf{r}_t$ represents the residual frame as the difference between the current and previous feature maps $\mathbf{x}_t - \mathbf{x}_{t-1}$. Since $\mathbf{z}_{t-1}$ has been already computed for the previous frame, computing $\mathbf{z}_t$ reduces to summing the term $\mathbf{w} * \mathbf{r}_t$. Due to the high correlation of consecutive frames in a video, the residual frame $\mathbf{r}_t$ is often sparse and contains non-zero values only for the regions that changed across time, *i.e.* moving objects as visualized in Figure 2. This sparsity can effectively be leveraged for efficiency: for every kernel support filled with zero values in $\mathbf{r}_t$, the corresponding output will be trivially zero, and the convolution can be skipped by copying values from $\mathbf{z}_{t-1}$ to $\mathbf{z}_t$.

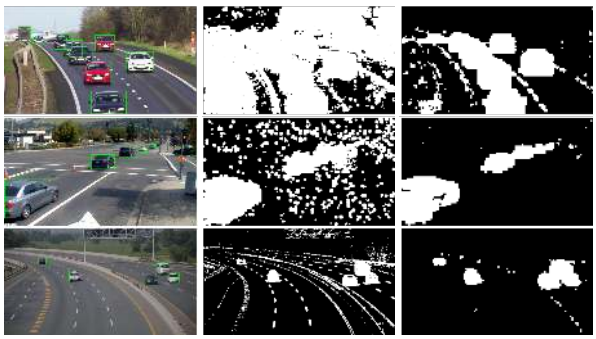
We use residuals to represent features at every convolutional layer. For the first frame, the residual $\mathbf{r}_1$ will be the same as the frame content $\mathbf{x}_1$, so the feature map is computed over the whole frame. Instead, consecutive frames update their features only at locations with non-zero residuals while reusing past representations elsewhere.

Although residuals are inherently sparse, they may still contain lots of locations with small non-zero values that prevent skipping them. To save even further, we introduce a gating function for each convolutional layer, $g : \mathbb{R}^{c_i \times h \times w} \rightarrow \{0, 1\}^{h \times w}$, to predict a binary mask indicating which locations should be processed, and taking only $\mathbf{r}_t$ as input. Using $\mathbf{r}_t$ as input provides a strong prior to the gating function, making it effective even with a fairly simple form. Putting it all together, our proposed Skip-Conv is defined as:

$$\tilde{\mathbf{z}}_t = \tilde{\mathbf{z}}_{t-1} + g(\mathbf{r}_t) \odot (\mathbf{w} * \mathbf{r}_t), \tag{3}$$

where $\odot$ indicates a broadcasted Hadamard (i.e. element-wise) product and the $\sim$ symbol highlights that $\tilde{\mathbf{z}}_t$ is an approximation of $\mathbf{z}_t$, as it skips negligible but non-zero residuals. The gating function is further described next.

[1]To avoid notational clutter, we describe the case in which $\mathbf{x}_t$ and $\mathbf{z}_t$ have the same resolution.



Input frame Layer 3 Layer 30

Figure 2: Gating masks for video object detection. Gates become more selective at deeper layers, concentrating on task specific regions. Frames from [61, 60, 62].

3.2. Skipping Non-zero Residuals

We propose two gating functions: *i)* Norm gate, that decides to skip a residual if its magnitude (norm) is small enough. This gate does not have any learnable parameter and does not involve any training. As a result, it can be easily plugged into any trained image network without any labeled video or training resources required. *ii)* Gumbel gate, that has parameters trained jointly with the convolutional kernels. The learned parameters can make the Gumbel gate more effective at the cost of fine-tuning the model.

3.2.1 Norm Gate

A naive form of gating is based on applying a scalar threshold ϵ to the norm of each output pixel:

$$g(\mathbf{r}_t, \mathbf{w}, \epsilon) = \text{round}\left(\sigma(\|\mathbf{w} * \mathbf{r}_t\|_p - \epsilon)\right), \tag{4}$$

where $\sigma(\cdot)$ indicates a sigmoid function, p represents the order of the norm, and the norm is computed over all channels for each position. However, such a gating function requires the computation of the convolution at each pixel of the residual, which would reintroduce inefficiency. We therefore propose to approximate Eq. 4 by considering the norm of each kernel support in the residual as:

$$g(\mathbf{r}_t, \epsilon) = \text{round}\left(\sigma(\|\mathbf{r}_t\|_p - \epsilon)\right), \tag{5}$$

We refer to this function as *Input-Norm gate*. The norm $\|\mathbf{r}_t\|_p$ in Eq. 5 is to be intended for local convolutional supports rather than pixel-wise. As such, it is computed by applying an absolute value function to $\mathbf{r}_t$ then taking sum within the $d_i \times k_h \times k_w$ neighborhood (*i.e.* $p = 1$).

A more accurate approximation can be achieved without computing the full convolution, by involving the norm of the weight matrix $\mathbf{w}$. Considering Young's inequality [58] we get an upper bound on the norm of the convolution of

two vectors $\mathbf{f}$ and $\mathbf{g}$:

$$\|\mathbf{f} * \mathbf{g}\|_r \leq \|\mathbf{f}\|_s \cdot \|\mathbf{g}\|_q, \quad \text{where} \quad \frac{1}{s} + \frac{1}{q} = \frac{1}{r} + 1. \tag{6}$$

By following Eq. 6, we define a more precise approximation, based on the norms of the input residual $\mathbf{r}_t$ and the weight matrix $\mathbf{w}$, in what we refer to as *Output-Norm gate*:

$$g(\mathbf{r}_t, \mathbf{w}, \epsilon) = \text{round}\left(\sigma(\|\mathbf{w}\|_p \cdot \|\mathbf{r}_t\|_p - \epsilon)\right), \tag{7}$$

where the norm $\|\mathbf{w}\|_p$ is computed over all four dimensions. We set the order p for both input-norm and output-norm gates to 1 (i.e., l_1 norm), and we share the margin ϵ between all layers. More flexible strategies such as layer-specific ϵ can potentially yield better results at the cost of more hyperparameter tweaking.

3.2.2 Gumbel Gate

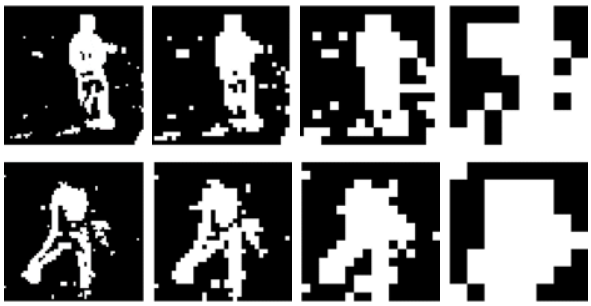
Residual norms indicate regions that change significantly across frames. However, not all changes are equally important for the final prediction (*e.g.* changes in background). This observation suggests that a higher efficiency can be gained by introducing some trainable parameters within gates, which are learned to skip even large residuals when they do not affect the model performance.

For each convolutional layer l we define a light-weight gating function $f(\mathbf{r}_t; \phi_l)$, parameterized by ϕ_l, as a convolution with a single output channel. Such an addition imposes a negligible overhead to the convolutional layer, which normally has dozen to hundreds of output channels. To generate masks of the same resolution, the gate function uses the same kernel size, stride, padding, and dilation as its corresponding layer. The gating function f outputs unnormalized scores that we turn into pixel-wise Bernoulli distributions by applying a sigmoid function. During training, we sample binary decisions from the Bernoulli distribution, whereas we round the sigmoids at inference:

$$g(\mathbf{r}_t, \phi_l) \begin{cases} \sim \text{Bern}(\sigma(f(\mathbf{r}_t; \phi_l))) & \text{at training,} \\ = \text{round}\left(\sigma(f(\mathbf{r}_t; \phi_l))\right) & \text{at inference} \end{cases} \tag{8}$$

We employ the Gumbel reparametrization [20, 32] and a straight-through gradient estimator [3] in order to backpropagate through the sampling procedure. The gating parameters are learned jointly with all model parameters by minimizing $\mathcal{L}_{task} + \beta\mathcal{L}_{gate}$. The hyper-parameter β balances the model accuracy, measured by $\mathcal{L}_{task}$, vs the model efficiency as measured by $\mathcal{L}_{gate}$. We define the gating loss as the average multiply-accumulate (MAC) count needed to process T consecutive frames as:

$$\mathcal{L}_{gate}(\phi_1, \ldots, \phi_L) = \frac{1}{T-1} \sum_{t=2}^{T} \sum_{l=1}^{L} m_l \cdot \mathbb{E}[g(\mathbf{r}_t, \phi_l)], \tag{9}$$



1x1 2x2 4x4 8x8

Figure 3: Exemplar masks generated by Skip-Conv for pose estimation, when trained with different block structures.

where L is the number of layers in the network, $\mathbb{E}[\cdot]$ indicates an average over spatial locations and the coefficient m_l denotes the MAC count for the l^{th} convolutional layer [2]. Similar to recurrent networks, we train the model over a fixed-length sequence of frames and do inference iteratively on an arbitrary number of frames.

Structured Sparsity Similar to sparse convolutions, an efficient implementation of Skip-Conv requires block-wise structured sparsity in the feature maps [42, 51], for two main reasons. First, block structures can be leveraged to reduce the memory overhead involved in gathering and scattering of input and output tensors [42]. Additionally, many hardware platforms perform the convolutions distributed over small patches (*e.g.* 8×8), so do not leverage any fine-grained spatial sparsity smaller than these block sizes.

Skip-Conv can be extended to generate structured sparsity by simply adding a downsampling and an upsampling function on the predicted gates. More specifically, we add a max-pooling layer with the kernel size and stride of b followed by a nearest neighbor upsampling with the same scale factor of b. This enforces the predicted gates to have $b \times b$ structure, as illustrated in Figure 3. We will illustrate in Section 4.3 how adding structure, despite significantly reducing the resolution of the gates, does not harm performances when compared with unstructured gating. Thus, structured sparsity enables more efficient implementation with minimal effect on performance.

3.3. Generalization & Future Work

Skip-Conv computes the output features in three steps: *i)* encoding the input tensor as residuals using a global subtraction transform. *ii)* efficient computation in the residual domain by leveraging the sparsity. *iii)* decoding the output back into the feature space using a global addition transform (inverse of subtraction). Here we generalize this process to a broader set of transformations beyond global subtrac-

[2] To keep $\mathcal{L}_{gate}$ at a manageable scale, we normalize m_l by dividing it by $\sum_{i=1}^{L} m_i$.

tion/additions for the interested reader, but leave these ideas to be explored in future work.

Whereas in Eq. 2 we defined the residual as $\mathbf{r}_t = \mathbf{x}_t - \mathbf{x}_{t-1}$, we may more generally define $\mathbf{r}_t = f_{\mathbf{x}_{t-1}}(\mathbf{x}_t)$ as an $\mathbf{x}_{t-1}$-dependent (approximately) invertible function f of $\mathbf{x}_t$, that produces a sparse generalized residual $\mathbf{r}_t$. As before, we may then write:

$$\mathbf{z}_t = \mathbf{w} * \mathbf{x}_t = \mathbf{w} * f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t). \tag{10}$$

Now, if convolution is equivariant to $f^{-1}_{\mathbf{x}_{t-1}}$, i.e. if the equation $\mathbf{w} * f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t) = \tilde{f}^{-1}_{\mathbf{x}_{t-1}}(\mathbf{w} * \mathbf{r}_t)$ holds for some function $\tilde{f}^{-1}_{\mathbf{x}_{t-1}}$ acting on the output space of the convolution, then we can compute $\mathbf{z}_t$ via a convolution with a sparse $\mathbf{r}_t$ followed by a transformation by $\tilde{f}$ (which should be chosen to be efficiently computable):

$$\mathbf{z}_t = \mathbf{w} * f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t) = \tilde{f}^{-1}_{\mathbf{x}_{t-1}}(\mathbf{w} * \mathbf{r}_t). \tag{11}$$

The original Skip-Conv is recovered by setting $f_{\mathbf{x}_{t-1}}(\mathbf{x}_t) = \mathbf{x}_t - \mathbf{x}_{t-1}$, so that $f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t) = \mathbf{r}_t + \mathbf{x}_{t-1}$ and the output transformation is $\tilde{f}^{-1}_{\mathbf{x}_{t-1}}(\mathbf{w} * \mathbf{r}_t) = \mathbf{z}_{t-1} + \mathbf{w} * \mathbf{r}_t$.

The question of when a convolution is equivariant to a given group of transformations has received a lot of attention in the literature [6, 23, 5]. The general answer is that $\mathbf{w}*$ can be made equivariant by linearly constraining the filters, resulting in so-called steerable filters [10]. In this case, however, the group of transformations generated by $f_{\mathbf{x}_{t-1}}$ for all $\mathbf{x}_{t-1}$ may not be known in advance, so analytically solving the linear constraints on the filters is not feasible. Nevertheless, equivariance can be encouraged via a simple loss term that pulls $\mathbf{w} * f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t)$ and $\tilde{f}^{-1}_{\mathbf{x}_{t-1}}(\mathbf{w} * \mathbf{r}_t)$ closer.

One promising choice for $f_{\mathbf{x}_{t-1}}$ is to compute a residual between $\mathbf{x}_t$ and a *warped* version of $\mathbf{x}_{t-1}$. This operation is guaranteed to be invertible (just add back the warped $\mathbf{x}_{t-1}$) and is equivariant whenever the warping operation is equivariant. Formally, let T denote a warping operation, *e.g.* bilinear interpolation of a frame at a set of points indicated by a flow field. The flow field could be computed from the network input frames, for instance. We may define $f_{\mathbf{x}_{t-1}}(\mathbf{x}_t) = \mathbf{x}_t - T(\mathbf{x}_{t-1})$, so that $f^{-1}_{\mathbf{x}_{t-1}}(\mathbf{r}_t) = \mathbf{r}_t + T(\mathbf{x}_{t-1})$ and, if the warp is equivariant, $\tilde{f}^{-1}_{\mathbf{x}_{t-1}}(\mathbf{w} * \mathbf{r}_t) = \tilde{T}(\mathbf{z}_{t-1}) + \mathbf{w} * \mathbf{r}_t$ (where $\tilde{T}$ applies the warp to the convolution output space). Other choices for the function f and $\tilde{f}$ also apply, including learning them from data.

4. Experiments

We evaluate Skip-Conv on two stream processing tasks, namely object detection and single-person pose estimation, in Section 4.1 and 4.2 respectively. Several ablation studies are reported in Section 4.3.

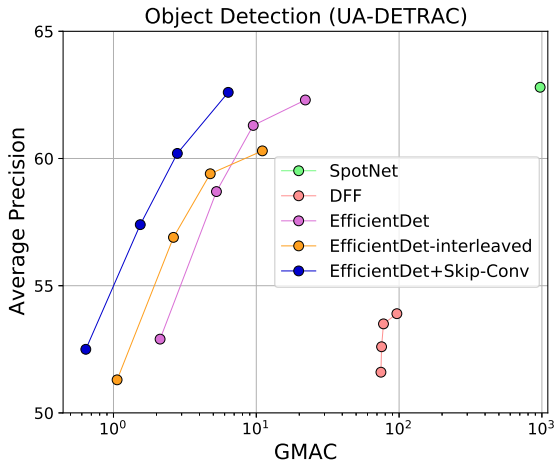


Figure 4: Comparison with the video object detection state-of-the-art. Skip-Conv reduces EfficientDet cost by 300%, consistently across the configurations D0, D1, D2, D3.

4.1. Object Detection

Experimental setup We conduct object detection experiments on UA-DETRAC dataset [54]. It consists of over 140,000 frames capturing 100 real-world traffic videos with bounding box annotations provided for vehicles at every frame. The dataset comes with a standard partitioning of 60 and 40 videos as train and test data, respectively. The performance is evaluated in terms of average precision (AP), averaged over multiple IoU thresholds varying from 0.5 to 0.95 with a step size of 0.05, similar to [48].

Implementation details We use EfficientDet [48], the state of the art architecture for object detection, and apply Skip-Conv on top of it. We conduct our experiments on D0 to D3 as the most efficient configurations [48], though more expensive configurations, *i.e.* D4 to D7, can similarly benefit from Skip-Conv. Each model is initialized with pre-trained weights from MS COCO dataset [28] and trained using SGD optimizer with momentum 0.9, weight decay $4e-5$ and an initial learning rate of 0.01 for 4 epochs. We decay the learning rate of a factor of 10 at epoch 3. All models are trained with mini-batches of size 4 using four GPUs, where synchronized batch-norm is used to handle small effective batch sizes. We use Skip-Conv with learned gates, which is trained for each EfficientDet configuration using the sparsity loss coefficient set to $\beta = 0.01$. During training we apply random flipping as data augmentation. The clip length is set to 4 frames both for training and inference.

Comparison to state of the art We compare Skip-Conv to several image and video object detectors: *i)* EfficientDet [48] as the state of the art in efficient object detection in images. We also include an EfficientDet-interleaved

	GMAC	Head	Sho.	Elb.	Wri.	Hip	Knee	Ank.	Avg
Park *et al.* [37]	-	79.0	60.3	28.7	16.0	74.8	59.2	49.3	52.5
Nie *et al.* [56]	-	80.3	63.5	32.5	21.6	76.3	62.7	53.1	55.7
Iqbal *et al.* [17]	-	90.3	76.9	59.3	55.0	85.9	76.4	73.0	73.8
Song *et al.* [45]	-	97.1	95.7	87.5	81.6	98.0	92.7	89.8	92.1
Luo *et al.* [31]	70.98	98.2	96.5	89.6	86.0	98.7	95.6	90.9	93.6
DKD *et al.* [34]	8.65	98.3	96.6	90.4	87.1	99.1	96.0	**92.9**	94.0
HRNet-w32 [52]	10.19	98.5	97.3	91.8	87.6	98.4	95.4	90.7	94.5
+S-SVD [18]	5.04	97.9	96.9	90.6	87.3	98.7	95.3	91.1	94.3
+W-SVD [67]	5.08	97.9	96.3	87.2	82.8	98.1	93.2	88.8	92.4
+L0 [30]	4.57	97.1	95.5	86.5	81.7	98.5	92.9	88.6	92.1
+Skip-Conv	5.30	**98.7**	**97.7**	**92.0**	**88.1**	**99.3**	**96.6**	91.0	**95.1**

Table 1: Comparison with the state-of-the-art on JHMDB. Skip-Conv outperforms in PCK the best image and video models, whilst requiring fewer MAC per frame.

baseline, where model predictions are propagated from keyframes to the next frames without further processing. *ii)* Deep Feature Flow (DFF) [69] as a seminal work on efficient object detection in video, *iii)* SpotNet [38] as the top performer in UA-DETRAC benchmark, which trains a joint model to detect objects and extract motion masks for improved object detection in video. Figure 4 demonstrates that Skip-Conv significantly reduces the computational cost of EfficientDet with a reasonable accuracy drop. More specifically, for D3 configuration, Skip-Conv reduces the cost from 22.06 to 6.36 GMAC with even a slight increase in AP from 62.3 to 62.6. Similarly for other configurations, Skip-Conv consistently reduces the MAC count by 330% to 350%. By comparing Skip-Conv and EfficientDet-interleaved, we observe that although interleaved detection reduces the computational cost, it leads to severe accuracy drop as there are lots of motion and dynamics in this dataset. Moreover, we observe that Skip-Conv outperforms DFF [69] both in terms of accuracy and computational cost. We hypothesize that DFF performances, solely relying on optical-flow to warp features across frames, are sensitive to the accuracy of the predicted motion vectors. However, there are lots of small objects (*e.g.* distant vehicles) in this dataset for which optical flow predictions are noisy and inaccurate. Finally, our experiments demonstrate that Skip-Conv achieves the state of the art accuracy on UA-DETRAC dataset, reported by SpotNet [38], with orders of magnitude less computes (6.36 vs 972.0 GMAC).

4.2. Human Pose Estimation

Experimental setup We conduct our experiments on the JHMDB dataset [21], a collection of 11,200 frames from 316 video clips, labeled with 15 body joints. Video sequences are organized according to three standard train/test partitions and we report average results over the three splits. We evaluate the performance using the standard PCK metric [57]. Given a bounding box of the person with height h and width w, PCK considers a candidate keypoint to be a valid match if its distance with the ground-truth keypoint is lower than $\alpha \cdot \max(h, w)$. We set $\alpha = 0.2$. Our experimental setup is consistent with prior works [45, 31, 34].

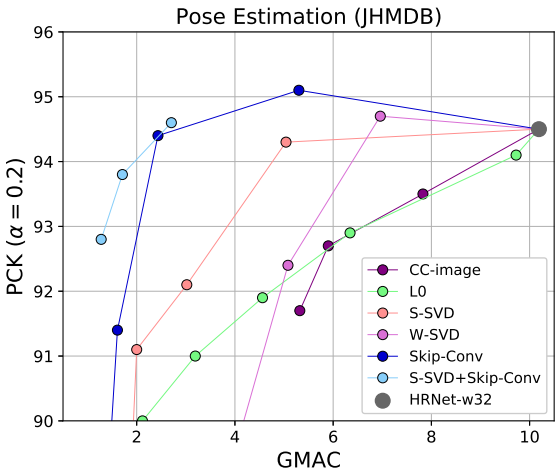


Figure 5: Comparison with model compression on JHMDB. Skip-Conv outperforms existing approaches. Applying it on top of compressed models further improves efficiency.

Implementation details We use HRNet [52], the state of the art architecture for human pose estimation, and apply Skip-Conv on top of it. We select HRNet-w32 as it performs on par with HRNet-w48, while being more efficient. All models are trained for 100 epochs with mini-batches of 16 images, using the Adam optimizer [22] with an initial learning rate of 0.001. We decay the learning rate with a factor of 10 at epochs 40 and 80. We use Skip-Conv with learned gates, which is trained using the sparsity loss coefficient set to $\beta = 1e-5$ unless specified otherwise.

We follow the setup from [45, 31, 34] for training and inference. We use standard data augmentations during training: randomly scaling using a factor within $[0.6, 1.4]$, random rotation within $[-40°, 40°]$ and random flipping. Each frame is cropped based on the ground-truth bounding box and padded to 256×256 pixels. The inference is done on a single scale. The clip length is set to $T = 8$ frames both for training and inference.

Comparison to state of the art We compare Skip-Conv to two categories of prior works: *i)* task specific methods, which are dedicated to efficient human pose estimation in video *i.e.* by dynamic kernel distillation (DKD) [34]. *ii)* task agnostic methods, which optimize the model efficiency for any task and architecture, *i.e.* by model compression and pruning. For this purpose, we apply ℓ_0 channel pruning [30], Spatial SVD (S-SVD) [18], and Weight SVD (W-SVD) [67] to compress HRNet-w32 at different efficiency vs. accuracy trade-offs. For S-SVD and W-SVD, we use greedy search to select the optimal rank per layer as implemented in [41]. For batch-norm layers in ℓ_0 channel pruning, we estimate the statistics during test on a large batch of 48 images, as it performs better in our experiments than using batch statistics from training. Finally, we compare to a

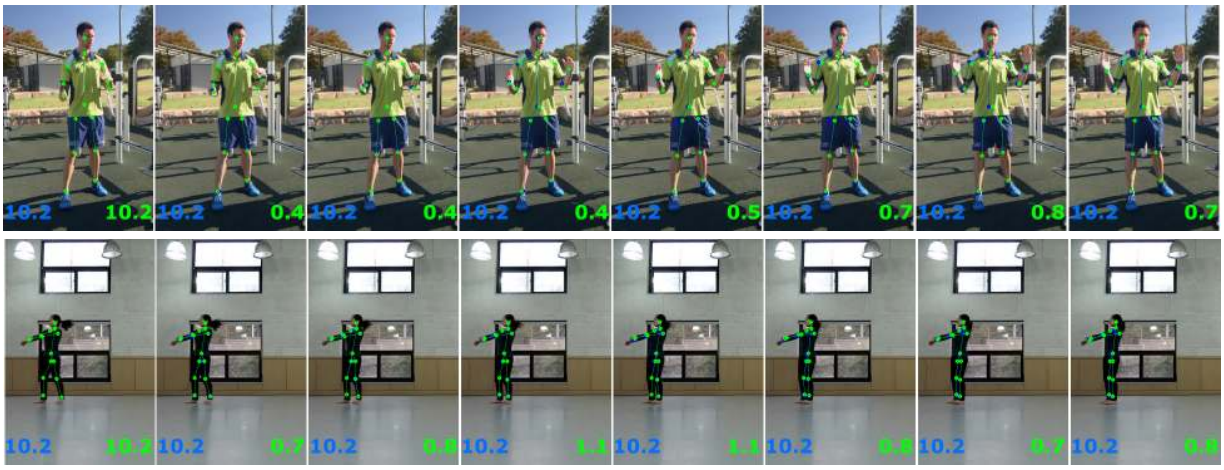


Figure 6: Qualitative comparison between HRNet and HRNet+Skip-Conv. GMAC count is reported for each frame. Skip-Conv significantly reduces the computations with minimal difference in predictions. Frames from [64, 63].

conditional computation baseline on images, $CC-image$, by applying Gumbel gates to the raw frames, instead of residual frames, similar to [51].

Table 1 reports the comparison with the state of the art models on the JHMDB dataset. By comparing Skip-Conv results with the backbone network HRNet-w32, results highlight that skipping redundant computations allows a reduction in MAC count by roughly a factor of 2, even with a remarkable improvement in PCK from 94.5 to 95.1. We attribute such a performance increase to a regularizing effect from the firing of stochastic gates during training. Moreover, when compared with DKD [34], Skip-Conv yields again a 1 point margin in PCK, with a relative cost reduction of 38.7%. Finally, out of the model compression baselines, S-SVD excels by halving the MAC count of HRNet-w32 with a minimal reduction in accuracy, even outperforming DKD in terms of the PCK vs cost trade-off. Notably, W-SVD and L0 regularization achieve similar compression rates, but with more severe performance degradations.

The comparison between Skip-Conv and model compression baselines can be best understood by looking at Figure 5, that reports PCK and MAC count at different operating points. The figure clearly shows the better trade-off achieved by Skip-Conv, which is able to retain the original HRNet-w32 performance whilst reducing the cost by more than a factor 4. On the contrary, other baselines experience higher drop in performance when increasing their compression ratios, with the best trade-off achieved by S-SVD. However, we remark that model compression and Skip-Conv tackle two very different sources of inefficiency in the base model: if the former typically focuses on cross-channel or filter redundancies, the latter tackles temporal redundancies. For these reasons, a combination of the two approaches could further improve efficiency, as also testified by the cyan line in Figure 5, that we obtain by applying Skip-Conv to different S-SVD compressed models. Indeed, the combination of such strategies outperforms both of them, especially in the low-cost regime. Finally, the comparison between Skip-Conv and $CC-image$ highlights the importance of conditioning on residuals, as they provide a strong prior to distinguish relevant and irrelevant locations. Figure 6 depicts examples of Skip-Conv predictions.

	GMAC	Time (ms)	MAC Red.	Time Red.
Conv	10.19	548	1.00 ×	1.00 ×
Skip-Conv	4.07	369	2.51 ×	1.48 ×
	2.35	287	4.33 ×	1.91 ×
	1.29	134	7.92 ×	4.09 ×

Table 2: MAC count vs runtime reductions on a HRNet-w32 architecture. The MAC count reductions obtained by Skip-Conv translate to wall-clock runtimes.

Runtime speed up We investigate how the theoretical speed ups, measured by MAC count reductions, translate to actual wall clock runtimes. Following [7] we use *im2col* based implementation of sparse convolutions. This algorithm reformulates the convolution as a matrix multiplication between input tensor and convolution kernels flattened as two matrices. The multiplication is computed only on non-sparse columns while filling the other columns by zero. We report the overall wall clock time spent on conv layers vs Skip-Conv layers for a HRNet-w32 architecture. The runtimes are reported on CPU[3]. As reported in Table 2, the MAC count reductions obtained by Skip-Conv translate to wall clock runtimes. The improvements on runtimes are roughly half of the theoretical speed ups as MAC count does not count for memory overheads involved in sparse convolutions. The gap between theoretical and real runtime improvements can be further reduced through highly optimized CUDA kernels as demonstrated in [42, 51].

[3]Intel Xeon e5-1620 @ 3.50GHz.

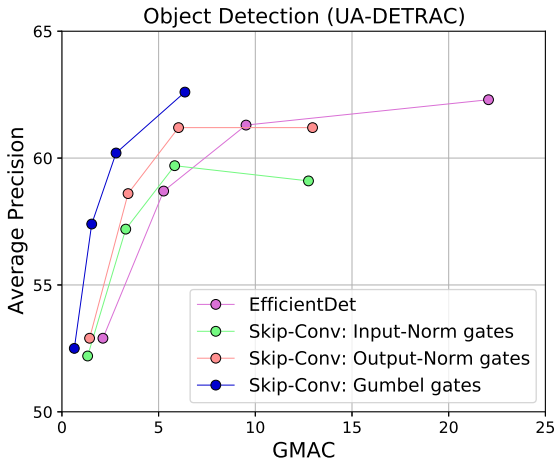


Figure 7: Comparison of different gates for Skip-Conv. Output-Norm gates improve EfficientDet, though not as effectively as Gumbel gates.

4.3. Ablation studies

Impact of gating function We study the impact of gating on Skip-Conv by evaluating EfficientDet architectures using three different gate functions (Section 3.2): *i)* Input-Norm gates with threshold value $\epsilon = 1e-2$; *ii)* Output-Norm gates, with threshold value $\epsilon = 15e-5$; *iii)* Gumbel gates are trained with sparsity coefficient $\beta = 1e-2$.

Figure 7 illustrates that Gumbel gates outperform both the Input-Norm and Output-Norm gates. This behavior is expected, as Gumbel gates are trained end-to-end with the model and they learn to skip the residuals, regardless of their magnitude, if it does not affect the task loss. Therefore, they effectively skip the big changes in background, which leads to higher computational efficiency. Moreover, we observe that output-norm gates outperform input-norm gates as they rely on a more precise approximation involving weight norms. Despite their simplicity, output-norm gates improve the efficiency of EfficientDet with reasonable accuracy drop. As an example, for D2 configuration output-norm gates reduce the cost from 9.52 to 6.03 GMAC with a similar AP of 61.2. Although output-norm gates are less effective than Gumbel gates, they are practically valuable as they can be plugged into any trained network without any labeled video or training required.

Block size	GMAC	PCK
1×1	5.06	95.0
2×2	2.43	94.5
4×4	2.98	94.9
8×8	4.18	95.0

Table 3: Impact of structured gates on pose estimation. Structured gates perform comparable to unstructured gates (1×1 blocks), while allowing for efficient implementations.

	$T_{train} = 4$		$T_{train} = 8$	
	PCK	**GMAC**	**PCK**	**GMAC**
$T_{test} = 4$	95.3	3.10	94.5	3.56
$T_{test} = 8$	94.3	1.91	94.5	2.43
$T_{test} = \infty$	89.3	1.18	94.2	1.80

Table 4: Results for pose estimation when training and testing with different clip lengths T.

Structured gating We experiment with structured gating on JHMDB (split 1) and report results in Table 3. The table reports the accuracy and efficiency of two unstructured models trained with different sparsity coefficients β, along with structured models with 4×4 and 8×8 blocks. It can be noted how adding structure to the gates does not negatively impact the model, yielding results that are inline with unstructured counterparts. This finding suggests that structured gates introduce hardware friendliness without hurting the accuracy/cost tradeoff.

Impact of clip length We study the sensitivity of Skip-Conv to clip length used during training and to reference frame reset frequency during test. Table 4 shows results on JHMDB (split 1), where we train Gumbel gates with clips of 4 or 8 frames with a sparsity factor $\beta = 5e-5$. Similarly, we instantiate a new reference frame during test every 4 or 8 frames, or even only once at the beginning of each sequence ($t = \infty$). As one can expect, the table shows how decreasing the number of expensive reference frames improves efficiency. This comes, however, at a minor cost in PCK, with a drop of 0.3 PCK for processing up to 40 frames sequences when training with clips having length $T = 8$.

5. Conclusion

We propose Skip Convolutions to speed up convolutional nets on videos. Our core contribution is the shift of the convolution from the content frames to the residual frames, both at input and intermediate layers. Operating on residual frames allows to skip most of the regions in the feature maps, for which representations can simply be copied from the past. We further encourage this regime by per layer gating functions, for which we propose several trainable and off-the-shelf designs.

As a potential limitation, we highlight it is unclear how our model would perform in the presence of severe camera motion. In such situations, residual frames wouldn't bear that much information about relevant regions, thus a higher burden would be put on the gating function. Coupling Skip-Conv with learnable warping functions helps compensating for severe camera motions, and is deferred to future work.

Acknowledgements We thank Arash Behboodi, Fatih Porikli, and Max Welling for feedback and discussions.

References

[1] B. E. Bejnordi, T. Blankevoort, and M. Welling. Batch-shaping for learning conditional channel gated networks. In *ICLR*, 2020. 2

[2] E. Bengio, P.-L. Bacon, J. Pineau, and D. Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015. 2

[3] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 4

[4] V. Campos, B. Jou, X. Giró-i Nieto, J. Torres, and S.-F. Chang. Skip rnn: Learning to skip state updates in recurrent neural networks. In *ICLR*, 2018. 2

[5] T. Cohen, M. Geiger, and M. Weiler. A general theory of equivariant CNNs on homogeneous spaces. In *NeurIPS*, 2019. 5

[6] T. S. Cohen and M. Welling. Group equivariant convolutional networks. In *ICML*, 2016. 5

[7] X. Dong, J. Huang, Y. Yang, and S. Yan. More is less: A more complicated network with less inference complexity. In *CVPR*, 2017. 2, 7

[8] C. Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *CVPR*, 2020. 2

[9] M. Figurnov, M. D. Collins, Y. Zhu, L. Zhang, J. Huang, D. Vetrov, and R. Salakhutdinov. Spatially adaptive computation time for residual networks. In *CVPR*, 2017. 2

[10] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *TPAMI*, 1991. 5

[11] X. Gao, Y. Zhao, Ł. Dudziak, R. Mullins, and C.-z. Xu. Dynamic channel pruning: Feature boosting and suppression. In *ICLR*, 2019. 2

[12] W. Gerstner and W. M. Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002. 1

[13] Y. He, X. Zhang, and J. Sun. Channel pruning for accelerating very deep neural networks. In *ICCV*, 2017. 2

[14] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *NeurIPS Workshops*, 2014. 2

[15] P. Hu, F. Caba, O. Wang, Z. Lin, S. Sclaroff, and F. Perazzi. Temporally distributed networks for fast video semantic segmentation. In *CVPR*, 2020. 2

[16] G. Huang, D. Chen, T. Li, F. Wu, L. van der Maaten, and K. Weinberger. Multi-scale dense networks for resource efficient image classification. In *ICLR*, 2018. 2

[17] U. Iqbal, M. Garbade, and J. Gall. Pose for action-action for pose. In *FG*, 2017. 6

[18] M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up convolutional neural networks with low rank expansions. *BMVC*, 2014. 2, 6

[19] S. Jain, X. Wang, and J. E. Gonzalez. Accel: A corrective fusion network for efficient semantic segmentation on video. In *CVPR*, 2019. 2

[20] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *ICLR*, 2017. 2, 4

[21] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, 2013. 6

[22] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 6

[23] R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *ICML*, 2018. 5

[24] A. Kuzmin, M. Nagel, S. Pitre, S. Pendyam, T. Blankevoort, and M. Welling. Taxonomy and evaluation of structured compression of convolutional neural networks. *arXiv preprint arXiv:1912.09802*, 2019. 2

[25] X. Li, Z. Liu, P. Luo, C. Change Loy, and X. Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *CVPR*, 2017. 2

[26] Y. Li, J. Shi, and D. Lin. Low-latency video semantic segmentation. In *CVPR*, 2018. 2

[27] J. Lin, C. Gan, and S. Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, 2019. 2

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[29] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko. Looking fast and slow: Memory-guided mobile video object detection. *arXiv preprint arXiv:1903.10172*, 2019. 2

[30] C. Louizos, M. Welling, and D. P. Kingma. Learning sparse neural networks through l_0 regularization. In *ICLR*, 2018. 2, 6

[31] Y. Luo, J. Ren, Z. Wang, W. Sun, J. Pan, J. Liu, J. Pang, and L. Lin. Lstm pose machines. In *CVPR*, 2018. 6

[32] C. J. Maddison, A. Mnih, and Y. W. Teh. The concrete distribution: A continuous relaxation of discrete random variables. *ICLR*, 2017. 2, 4

[33] Y. Meng, C.-C. Lin, R. Panda, P. Sattigeri, L. Karlinsky, A. Oliva, K. Saenko, and R. Feris. Ar-net: Adaptive frame resolution for efficient action recognition. In *ECCV*, 2020. 2

[34] X. Nie, Y. Li, L. Luo, N. Zhang, and J. Feng. Dynamic kernel distillation for efficient pose estimation in videos. In *ICCV*, 2019. 2, 6, 7

[35] P. O'Connor, E. Gavves, M. Reisser, and M. Welling. Temporally efficient deep learning with spikes. In *ICLR*, 2018. 1

[36] P. O'Connor and M. Welling. Sigma delta quantized networks. *ICLR*, 2017. 1

[37] D. Park and D. Ramanan. N-best maximal decoders for part models. In *ICCV*, 2011. 6

[38] H. Perreault, G.-A. Bilodeau, N. Saunier, and M. Héritier. Spotnet: Self-attention multi-task network for object detection. *CRV*, 2020. 6

[39] A. Piergiovanni, A. Angelova, and M. S. Ryoo. Tiny video networks. *arXiv preprint arXiv:1910.06961*, 2019. 2

[40] C. Posch, D. Matolin, and R. Wohlgenannt. High-dr frame-free pwm imaging with asynchronous aer intensity encoding and focal-plane temporal redundancy suppression. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, 2010. 1

[41] *AI Model Efficiency Toolkit (AIMET)*, 2020 (accessed November 13, 2020). 6

[42] M. Ren, A. Pokrovsky, B. Yang, and R. Urtasun. Sbnet: Sparse blocks network for fast inference. In *CVPR*, 2018. 2, 4, 7

[43] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *ICLR*, 2015. 2

[44] E. Shelhamer, K. Rakelly, J. Hoffman, and T. Darrell. Clockwork convnets for video semantic segmentation. In *ECCV*, 2016. 2

[45] J. Song, L. Wang, L. Van Gool, and O. Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. In *CVPR*, 2017. 6

[46] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 2012. 1

[47] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ICML*, 2019. 2

[48] M. Tan, R. Pang, and Q. V. Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020. 2, 5

[49] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018. 2

[50] A. Veit and S. Belongie. Convolutional networks with adaptive inference graphs. In *ECCV*, 2018. 2

[51] T. Verelst and T. Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *CVPR*, 2020. 2, 4, 7

[52] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, et al. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2020. 6

[53] L. Wang, X. Dong, Y. Wang, X. Ying, Z. Lin, W. An, and Y. Guo. Learning sparse masks for efficient image super-resolution. *arXiv preprint arXiv:2006.09603*, 2020. 2

[54] L. Wen, D. Du, Z. Cai, Z. Lei, M. Chang, H. Qi, J. Lim, M. Yang, and S. Lyu. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *CVIU*, 2020. 5

[55] Z. Wu, C. Xiong, Y.-G. Jiang, and L. S. Davis. Liteeval: A coarse-to-fine framework for resource efficient video recognition. In *NeurIPS*, 2019. 2

[56] B. Xiaohan Nie, C. Xiong, and S.-C. Zhu. Joint action recognition and pose estimation from video. In *CVPR*, 2015. 6

[57] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 2012. 6

[58] W. H. Young. On the multiplication of successions of fourier constants. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 87(596):331–339, 1912. 3

[59] Youtube. Creative common license. https://support.google.com/youtube/answer/2797468. 10

[60] Youtube. https://www.youtube.com/watch?v=g7wtrxlal9w. license: [59]. 3

[61] Youtube. https://www.youtube.com/watch?v=Lce51evahUY. license: [59]. 3

[62] Youtube. https://www.youtube.com/watch?v=u9WkxauOdK0. channel: ilme aalim. license: [59]. 3

[63] Youtube. https://www.youtube.com/watch?v=wp4VVoszPU4. license: [59]. 7

[64] Youtube. https://www.youtube.com/watch?v=x0YsQLM67Js. license: [59]. 7

[65] Youtube. https://www.youtube.com/watch?v=YfejqqDJge0. license: [59]. 1

[66] D. Zambrano and S. M. Bohte. Fast and efficient asynchronous neural computation with adapting spiking neural networks. *arXiv preprint arXiv:1609.02053*, 2016. 1

[67] X. Zhang, J. Zou, K. He, and J. Sun. Accelerating very deep convolutional networks for classification and detection. *TPAMI*, 2015. 2, 6

[68] X. Zhu, J. Dai, L. Yuan, and Y. Wei. Towards high performance video object detection. In *CVPR*, 2018. 2

[69] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei. Deep feature flow for video recognition. In *CVPR*, 2017. 2, 6