PASS: Patch Automatic Skip Scheme for Efficient Real-Time Video Perception on Edge Devices

Qihua Zhou[1], Song Guo[1]*, Jun Pan[1], Jiacheng Liang[2], Zhenda Xu[1], Jingren Zhou[3]
[1]The Hong Kong Polytechnic University
[2]Pennsylvania State University
[3]Alibaba Group
{csqzhou, csjpan, cszxu}@comp.polyu.edu.hk, song.guo@polyu.edu.hk
ljcpsu@psu.edu, jingren.zhou@alibaba-inc.com

Abstract

Real-time video perception tasks are often challenging over the resource-constrained edge devices due to the concerns of accuracy drop and hardware overhead, where saving computations is the key to performance improvement. Existing methods either rely on domain-specific neural chips or priorly searched models, which require specialized optimization according to different task properties. In this work, we propose a general and task-independent *Patch Automatic Skip Scheme* (PASS), a novel end-to-end learning pipeline to support diverse video perception settings by decoupling acceleration and tasks. The gist is to capture the temporal similarity across video frames and skip the redundant computations at patch level, where the patch is a non-overlapping square block in visual. PASS equips each convolution layer with a learnable gate to selectively determine which patches could be safely skipped without degrading model accuracy. As to each layer, a desired gate needs to make flexible skip decisions based on intermediate features without any annotations, which cannot be achieved by conventional supervised learning paradigm. To address this challenge, we are the first to construct a tough self-supervisory procedure for optimizing these gates, which learns to extract contrastive representation, *i.e.,* distinguishing similarity and difference, from frame sequence. These high-capacity gates can serve as a plug-and-play module for convolutional neural network (CNN) backbones to implement patch-skippable architectures, and automatically generate proper skip strategy to accelerate different video-based downstream tasks, *e.g.,* outperforming the state-of-the-art MobileHumanPose (MHP) in 3D pose estimation and FairMOT in multiple object tracking, by up to $9.43\times$ and $12.19\times$ speedups, respectively. By directly processing the raw data of frames, PASS can generalize to real-time video streams on commodity edge devices, *e.g.,* NVIDIA Jetson Nano, with efficient performance in realistic deployment.

Introduction

Recent years have witnessed the unprecedented boom of video perception (Habibian et al. 2021; Ghodrati, Bejnordi, and Habibian 2021; Abati et al. 2020), which provides great prospects of enhancing multimedia entertainment, facilitating virtual collaboration and innovating industrial manufacture, by using the rich contents in video streams (Choi, Choi, and Kim 2021; Bu et al. 2021; Kondratyuk et al. 2021). Due to the rapid growth of device processing capacity and memory volume, it is a trend to transfer the video perception tasks from conventional servers to edge devices (Cai et al. 2020), *e.g.,* mobile phones and IoT sensors (Tsukada, Kondo, and Matsutani 2020). This on-device processing paradigm brings inherent benefits of addressing privacy leakage (McMahan et al. 2017), reducing communication latency (Perozzi, Al-Rfou, and Skiena 2014) and providing personalized inference (Fang, Zeng, and Zhang 2018). Unfortunately, handling video perception tasks on devices is not easy because the limited computation capacity often becomes the performance bottleneck in real-world implementation. As a result, saving computations is the key to build efficient on-device video perception systems.

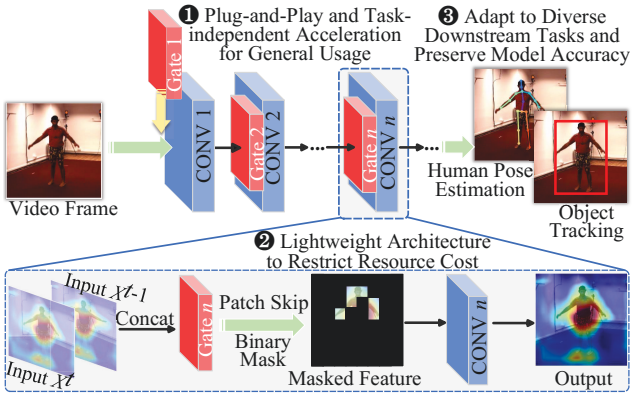

Figure 1: PASS accelerates on-device video perception tasks by safely skipping redundant computations on trivial patches, while not incurring accuracy drop on the output. We visualize CONV-n's heatmap for demonstration purpose. The black patches (*i.e.,* masked with binary flag "1" and "0" is the opposite) correspond to the skip decision learnt by the gate, where 58/64 regions (even in foreground) are skipped.

Considering the video is comprised of sequential frames, it is natural to capture the temporal similarity across frames and skip the redundant computations on the trivial regions. In video perception, the word of "region" corresponds to

*Corresponding author.

the term called *patch* (also called macroblock) (Ding et al. 2021), which is a non-overlapping square block with $n \times n$ pixels in visual. As a preliminary concept to our work, patch can serve as the basic processing units to explore intra- and inter-frame correlation (Dasari et al. 2022; Yeo et al. 2020; Narayanan et al. 2021). Different from our design philosophy at patch level, existing video acceleration models often resort to the feature propagation (Wang et al. 2021a; Li et al. 2021; Liu et al. 2021b) technique and reduce computations by selecting key frames (Ghodrati, Bejnordi, and Habibian 2021; Zhou et al. 2021). The feature extraction quality is dominated by the policy of frame selection, which often leads to huge accuracy gap between key frames and others (Habibian et al. 2021). Also, these models are optimized for specific tasks, thus involving massive modifications to network architecture (Choi, Choi, and Kim 2021) and requiring exclusive neural chips (Liu et al. 2021a; Bei, Yang, and Soatto 2021). Consequently, the dependence of domain-specific hardware and priorly optimized models limits the usage extensibility of previous methods, especially on edge devices.

These limitations raise an interesting question – *can we abstract away the computation saving problem from video perception tasks and build a task-independent acceleration methodology that can generalize to different runtime environments?* To achieve this target, we present the *Patch Automatic Skip Scheme* (PASS), which supports diverse video perception settings by decoupling acceleration and tasks (overview in Figure. 1). Considering the on-device execution environment, we develop three new quality-determining objectives, *i.e.,* ❶ **obtaining usage generalization**, ❷ **restricting computational cost**, and ❸ **preserving model quality**, to guide the design of PASS.

First, PASS exploits the temporal redundancy and skips computations at patch level, where a learnable gate is inserted to each convolution and can *selectively decide which patches could be safely skipped, without degrading the accuracy of feature extraction*. The skip decision is represented by a binary mask, which can precisely distinguish the similarity and difference across frames. Note that the gist of PASS is *not* simply segmenting the frames to remove the calculation on background. The essential is PASS can automatically skip the trivial patches even on the foreground according to the representation semantics of the neural network and significantly saves computations with no dependency on network architecture. As the basic matrix multiplication instructions inside the gate are inherently supported by commodity edge devices, PASS provides a general acceleration methodology for diverse video perception settings.

Second, PASS introduces the patch embedding technique to common convolutions. Recall that a patch represents a non-overlapping square block in visual and is the basic processing unit for feature extraction, we can effectively control the time cost of tensor operations by managing the computations at the patch level. Also, each gate is optimized with tiny structure to restrict the number of *Multiply Accumulate* (MAC) operations and parameter size. As a result, these gates provide powerful learning capacity while not aggravating the system runtime overhead.

Third, PASS is established with a two-stage optimization. During the task-independent pre-training, the gate is optimized via the self-supervised learning paradigm, which could extract the most essential feature semantics without any labels. By developing the binary mask to reflect the skip decision, we generate the pairs of contrastive samples to push the gate filter out the redundant patches with a high skip rate. Also, as to the task-oriented fine-tuning, the CNN backbone is further updated based on a small-scale labeled data and finally becomes more tolerant to patch skip. The fine-tuning procedure is *optional* and developer can selectively enable this procedure according to practical demands. Therefore, these high-capacity gates can serve as a plug-and-play module for CNN backbones to implement patch-skippable networks, and automatically generate proper skip strategy to accelerate different downstream tasks.

We implement PASS in PyTorch (PyTorch 2023), with easy-of-use APIs to handle the patch skip and accelerate video perception. Developers can import PASS in their downstream tasks and interact with torch neural engines. Thus it is easy to port PASS to commodity edge devices, *e.g.,* the NVIDIA Jetson Nano series (NVIDIA 2023). Evaluation shows that PASS can effectively save computational cost with good inference accuracy for different downstream tasks. Specifically, as to 3D human pose estimation, PASS achieves the 42.48% patch skip rate, 62.61% MAC reduction rate and $9.43\times$ processing speedup, over state-of-the-art (SOTA) MobileHumanPose (MHP) (Choi, Choi, and Kim 2021) method. Meanwhile, as to multiple object tracking, PASS achieves the 59.17% patch skip rate, 73.16% MAC reduction rate and $12.19\times$ processing speedup, over SOTA FairMOT (Zhang et al. 2021) method.

To the best of our knowledge, PASS is the first task-independent framework to implement patch-level computation saving mechanism for real-time video perception tasks on commodity edge devices, without massive code modifications on original CNN architecture. The project of PASS will be open-source and constantly contributes to the further development of mobile vision in practice. Overall, the key contributions of our work are as follows:

- We propose a task-independent computation saving methodology to accelerate real-time video perception tasks and present the *Patch Automatic Skip Scheme* (PASS) to build high-efficiency systems on commodity edge devices, without compromising the model accuracy.
- We develop the lightweight gate with high learning capacity to selectively generate flexible skip strategy for each convolution. Specifically, the gate is optimized with tiny structure to restrict system runtime overhead and serve as a plug-and-play module for CNN backbones. Also, the gate learns the invariant high-level feature semantics from contrastive samples, and can precisely distinguish the similarity and difference in video streams.
- We conduct extensive experiments in realistic challenging scenarios, including 3D human pose estimation with up to 3.6 million images and real-time object tracking with over 128 objects. Evaluation on commodity edge devices shows that PASS achieves higher system perfor-

mance over SOTA video perception methods in both processing speedup and model accuracy, which verifies the effectiveness of our approach.

Preliminary

Most video perception models are established on CNN backbones, where convolution (CONV) layers are the fundamental parts to extract features from frames and dominate the major computational overhead (*e.g.,* often up to 90%) of the entire model (Zhu et al. 2020). Therefore, conducting optimization on convolutions is the key to save computations for video perception tasks, where the temporal redundancy of input could be exploited – skipping trivial patches.

How patch skip saves computations? Under the transformation of `im2col`, each convolution layer generates the output by conducting the matrix multiplication on input X and the $K \times K$ kernel with stride S. For convenience, we assume the number of input channel and output channel is one, and ignore the impact of padding. The MAC count of a vanilla convolution is:

$$\text{MAC} = K^2 \cdot \frac{(X_w - K + S)}{S} \cdot \frac{(X_h - K + S)}{S}, \quad (1)$$

where X_w and X_h are the width and height of input, respectively. By making the stride S equal to the kernel size K, we can eliminate the overlap of different patches and entirely divide the input X into $\frac{X_w}{K} \times \frac{X_h}{K}$ patches in total, where K is carefully aligned to make X_w and X_h divisible by it. Then, the MAC count can be simplified as: $\text{MAC} = X_w \cdot X_h$. The above division procedure is called *Patch Embedding* (Tolstikhin et al. 2021; Trockman and Kolter 2022; Melas-Kyriazi 2021), which can significantly simplify the calculation rule of patch-level convolutions and is used in the architecture design of our learnable gate. If N patches is marked as trivial and the computations on them should be skipped, the MAC count will be reduced as: $\text{MAC} = X_w \cdot X_h - N \cdot K^2$, Based on the above analysis, we can figure out the overall patch skip rate $\mathcal{R}_p$ and the corresponding MAC reduction rate $\mathcal{R}_m$ (compared with vanilla convolution) as:

$$\mathcal{R}_p = \frac{N \cdot K^2}{X_w \cdot X_h}, \quad (2)$$

$$\mathcal{R}_m = 1 - \frac{S^2 \cdot (X_w \cdot X_h - N \cdot K^2)}{K^2 \cdot (X_w - K + S) \cdot (X_h - K + S)}. \quad (3)$$

These two terms directly reflect the computation saving efficiency and are key metrics in performance evaluation.

Summary. PASS first filters out trivial patches according to the temporal redundancy of input, then skips the corresponding matrix multiplication to achieve less MAC count and time cost. In practice, we can leverage the *Block-wise Sparse Convolution* (Ren et al. 2018; Verelst and Tuytelaars 2021) technique to handle the engineering implementation.

Method: Patch Automatic Skip Scheme

We first present the gate construction for generating binary mask. Then, we discuss PASS's two-stage learning pipeline.

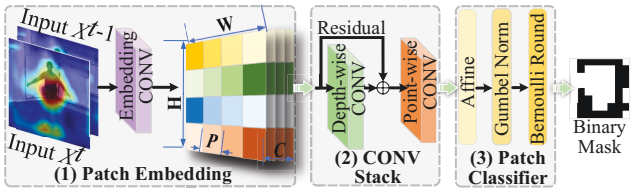

Figure 2: Structure details of the binary-mask gate.

Construction of Binary Mask Gate

After understanding the rationale of PASS's computation saving, our next step is to correctly determine which patches of a convolution could be safely skipped. We build a learnable gate to generate proper skip strategy according to input and layer characteristics. The design principle is to extract high-level semantics from input and yield a binary mask to represent skip strategy. As to each convolution layer, the gate analyzes current input (*i.e.,* the video frame in first layer and the input feature to others) and filters out which patches are trivial and should be skipped. As shown in Figure. 2, the gate is carefully constructed with a tiny architecture, where the output is a binary mask identifying the skip strategy of each patch. A patch is marked as skipped and reserved by value 1 and 0, respectively. The gate architecture is comprised of the following three key components.

1. Patch embedding. Recall that we divide the input into several patches without overlap to simplify the matrix multiplication of convolutions and yield less MACs, we achieve this function by setting the kernel and stride with a equal value, which is defined as the patch size P, *i.e.,* $K = S = P$. We concatenate the two inputs of current and previous iterations (*i.e.,* X^t and X^{t-1} in Figure. 2), and send the merged input to the subsequent convolution to extract the intermediate features, where the output channel number is marked as C. As a hyper-parameter, C is usually set in the range from 64 to 256 in our experiments. Also, the patch size P should make the intermediate feature's height H and width W divisible by it. Then, the intermediate feature will be sent to the second component to extract high-level semantics.

2. Stacked depth-wise separable convolutions. This component is based on the lightweight transformation of depth- and point-wise convolutions (Chollet 2017), which simplifies the computation complexity of feature extraction. By inserting a residual branch between the depth and point convolution blocks, we can learn the connection of different patches in both spatial and channel dimensions. Inspired by ConvMixer (Trockman and Kolter 2022), we stack entire component together and recursively do the feature extraction, usually by 2 to 16 times according to the scale of dataset. After that, the high-level feature is sent to the third component.

3. Patch classifier. This component indicates whether the patches are skipped or preserved. The skip strategy is represented by a $P \times P$ binary mask, where each element corresponds to a patch inside the input. The value 1 indicates the patch at that location should be skipped and 0 is opposite. Consider the high-level semantics generated by the

second component are represented by floating-point tensors, we need to make discretization and restrict the tensor values in binary. First, we use two linear layers for *affine* transformation and make the tensor size same as the patch number. The number of hidden neuron is 4096 for a sufficient representation capacity. Then, we need an proper activation function to generate the final binary mask, which reflects the patch skip decision. We discard the conventional Sigmoid activation used in Skip-CONV (Habibian et al. 2021) and FrameExit (Ghodrati, Bejnordi, and Habibian 2021), which easily incurs gradient vanishing and makes the gate hard to converge. Instead, we design a novel *Gumbel Normalization* (GN) layer to serve as the activation, which aims to squeeze the continuous value domain into a binary domain. Specifically, the GN layer introduces a noise ϵ sampled from the *Gumbel Distribution* (Jang, Gu, and Poole 2017), which is defined as $\epsilon = -\ln(-\ln(u))$, where u follows the uniform distribution that $u \sim \text{Uniform}(0, 1)$. By adding this noise to the original tensor $\mathbf{X}$, we can smooth the subsequent discretization operations and make the gradient calculation in back-propagation be correctly handled by the *Straight-through Estimator* (Bengio, Léonard, and Courville 2013). The approximated tensor is described as $\hat{\mathbf{X}} = (\mathbf{X} + \epsilon)/\tau$, where τ is a non-zero temperature scalar and is usually set between 0.5 and 0.7. After that, we develop an asymmetric normalization to bound the value domain of $\hat{\mathbf{X}}$ within $[0, 1]$, which is described as:

$$\texttt{GumbelNormalization}(\hat{\mathbf{X}}) = \frac{\hat{\mathbf{X}} - \min(\hat{\mathbf{X}})}{\max(\hat{\mathbf{X}}) - \min(\hat{\mathbf{X}})}. \tag{4}$$

As each element x inside the tensor $\hat{\mathbf{X}}$ locates in domain $[0, 1]$, we can utilize the Bernoulli sampling to round all the tensor elements into discretize values (*i.e.,* 1 or 0). This procedure is formulated as:

$$\texttt{Bernoulli}(x) = \begin{cases} 1, & \text{w.p.} \quad x, \\ 0, & \text{w.p.} \quad 1 - x, \end{cases} \quad x \in \hat{\mathbf{X}}, \tag{5}$$

where w.p. is short for *with probability* and the entire rounded tensor corresponds to the final binary mask.

As a result, the above three modules make our gate lightweight but with efficient learning capacity to generate proper patch skip strategies.

Task-Independent Pre-training for Gate

After constructing the gate structure, we need to make the gate learn essential knowledge of patch skip. As shown in Figure. 3, we conduct a pre-training procedure on gate to achieve this target via two steps: (1) generating the contrastive samples from original input for the subsequent gate optimization, and (2) extracting contrastive representation from the output to learn high-level feature semantics.

Step 1. Generation of contrastive samples. As our gate is designed as task-independent and can adopt to different downstream tasks, we cannot rely on the conventional supervised learning paradigm because the data needs to be exclusively annotated to match the demands of specific tasks, making the gate lose the usage generalization. To conquer

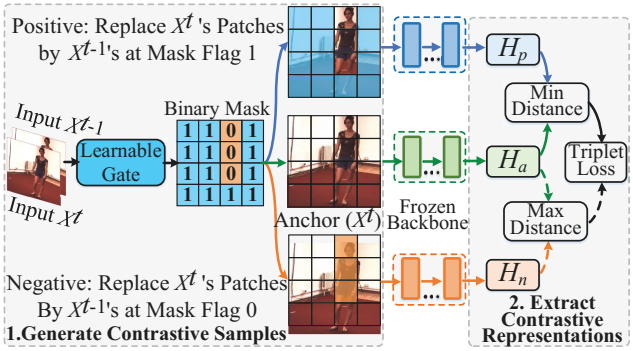

Figure 3: Task-independent pre-training to optimize gates and learn public feature semantics.

this challenge, we follow principle of self-supervised learning and generate the pairs of contrastive samples, with no need of labels. Thus, the dataset used for the gate pre-training can be significantly extended, which makes video clips with shuffled frames (*e.g.,* Human3.6M (Ionescu et al. 2014) and MPII (Andriluka et al. 2014)) can be used to help the gate learn high-level semantics. As to each convolution, we generate a pair of contrastive samples by adopting the binary mask on the original input. As shown in Figure. 3, we generate the *positive* and *negative* samples by replacing the patches with index 1 and 0, respectively. Here, "replacing patches" means to reuse the patches at the same locations in previous iteration to displace the patches in current iteration. Thus, a binary mask will generate two kinds of partial-replaced samples. Specifically, the original input is called the *anchor*, with the same grid of patch division as the contrastive pairs. We organize the three samples as a triplet, *i.e.,* $\{anchor, positive, negative\}$. By feeding this triplet to the backbone, we can get three intermediate features: (1) anchor heatmap H_a, (2) positive heatmap H_p, and (3) negative heatmap H_n. Note that the backbone is frozen and only the gate involves the parameter update. This heatmap triplet contains the most essential semantics learned by the gate, which will be used for the extraction of contrastive representation discussed next.

Step 2. Extraction of contrastive representation. A proper patch skip strategy should not degrade the final model accuracy, which means removing the convolutions on the skipped patches and reusing the features from previous iteration will generate a similar heatmap as that calculated based on the original input. As to the heatmap triplet, we need to make the positive heatmap H_p as close as to the anchor heatmap H_a, while the negative heatmap H_n is the opposite. This similarity can be reflected by the *Mean Square Error* between two heatmap tensors. The loss function is:

$$\mathcal{L} = \min : \underbrace{\text{MSE}(H_p, H_a)}_{\mathcal{L}_p} + \max : \underbrace{\text{MSE}(H_n, H_a)}_{\mathcal{L}_n}. \tag{6}$$

By conducting the gradient-descent based optimization on the triplet loss function $\mathcal{L}$, we can *minimize the distance between H_p and H_a, while maximizing the distance between H_n and H_a*. Specifically, we employ the Triplet-Margin-

Loss (Balntas et al. 2016) to handle the joint optimization of $\mathcal{L}_p$ and $\mathcal{L}_n$. This procedure makes the gate correctly distinguish the similarity and difference between any two frame batches, thus detecting which patches can be safely skipped while still generating similar heatmaps as the anchor.

Task-Oriented Fine-Tuning for Backbone

After the task-independent pre-training based on contrastive samples, we obtain a series of high-capacity gates which can be inserted into CNN backbones and generate proper skip strategies for each convolution. To enhance the performance of downstream tasks, we can further conduct the task-oriented fine-tuning based on a small-scale labeled data, following the supervised learning paradigm. For example, we can fine-tune the MHP backbone (Choi, Choi, and Kim 2021) on MPII dataset (Andriluka et al. 2014) to obtain a higher accuracy of human pose estimation. During the fine-tuning, the gates are frozen and only the parameters of backbone will be updated. We first use the binary mask generated by the gate to construct the positive sample. Then, we feed it to the backbone and obtain the final prediction Y, instead of the heatmap used in gate pre-training. We calculate the gap between Y and the ground-truth label G, and use the downstream-task loss to minimize this gap, which can be formulated as $\min\ :\ \mathcal{L}(Y, G)$. By conducting the back-propagation iteratively, the backbone will keep updating its parameters and become more tolerant to patch skip. Together with the gate, the robust backbone can yield precise predictions while saving computations. Note that this fine-tuning procedure is ***optional*** because the patch-skip capacity will be well obtained after the gate pre-training. Directly deploying inference based on the pre-trained gate and vanilla backbone can also achieve an adequate performance. However, considering the further improvement on task performance, we suggest doing this when there are sufficient hardware and dataset resources.

Evaluation

We evaluate PASS on two pertinent video perception tasks, *i.e.,* 3D human pose estimation and multiple object detection, both of which utilize commodity edge devices.

3D Human Pose Estimation

Experimental setup and metrics. We conduct experiments on the large-scale Human3.6M dataset (Ionescu et al. 2014) with 3.6 million 3D human poses and corresponding frames. The major metrics adopted by existing SOTA methods are: *Mean Per Joint Position Error* (MPJPE) (Ionescu et al. 2014), *Multiply Accumulate* (MAC) operations, and number of *Frames Per Second* (FPS). These metrics are used to compare our PASS with SOTA methods.

Comparison to SOTA methods. We compare PASS with five SOTA baselines: MobileHumanPose (MHP) (Choi, Choi, and Kim 2021), Skip-CONV (Habibian et al. 2021), MoVNect (Hwang et al. 2020), VNect (Mehta et al. 2017) and PoseNet (Moon, Chang, and Lee 2019), where the average MPJPE, MAC count, FPS are used to reflect the performance of inference accuracy, computation saving and processing speed, respectively. Recall that the backbone fine-tuning stage is optional, we also evaluate PASS in two schemes: (1) with gate pre-training only (PASS-P) and (2) with additional backbone fine-tuning on top of gate pre-training (PASS-F). Table 1 reports the comparison with SOTA methods on the Human3.6M dataset. The average patch skip rate $\mathcal{R}_p$ achieved by PASS-P and PASS-F are 39.06% and 42.48%, respectively. As to the model accuracy reflected by average MPJPE (lower is better), PASS holds the lowest value smaller than 52, which outperforms the SOTA performance achieved by MHP and PoseNet. Other methods (*e.g.,* Skip-CONV, VNect and MoVNect) hold a much higher value, which indicates their video models will easily incur accuracy degradation in the scenarios of complex background and partial keypoint occlusions. Existing methods usually focus on reducing computations by sampling frames based on heuristic algorithms and threshold-based filters, which is hard to match the runtime environment of on-device video streams and finally degrades the inference accuracy. Although the previous methods employs compressed models to improve processing speed, they often couple the acceleration with specific network architectures and dedicated hardware, thus yielding higher MAC count over our PASS. For example, PASS generates a MAC reduction rate by up to 47.39% over the MHP method, which is a significant improvement for video perception on edge devices. Note that PASS-F provides better metrics with lower MPJPE, higher MAC reduction rate and higher FPS over PASS-P, indicating that the optional fine-tuning stage can further improve the inference performance.

Method	MPJPE ↓	GMAC ↓	FPS ↑
PoseNet	54.05	27.59	3
VNect	79.55	11.73	6
MoVNect	98.46	2.58	22
Skip-CONV	66.54	4.65	15
MHP	52.06	3.27	20
PASS-P	**51.92**	**2.21**	**26**
PASS-F	**51.87**	**1.72**	**34**

Table 1: Comparison with SOTA in human pose estimation.

Environmental sensitivity. To evaluate the system robustness and environment sensitivity, we inspect the performance of PASS under different video settings, with 7 action scenarios (talking, eating, sitting, walking, etc), two background types (indoor and outdoor) and two camera viewpoints (fixed and moving). Table 2 reports the comparison with SOTA methods by checking the key metric of MPJPE. We can observe that PASS outperforms the five baselines in most cases. Specifically, as to the videos with plain indoor background and fixed camera, existing methods may achieve comparable or slightly lower MPJPE over PASS-P, *e.g.,* PoseNet for eating. However, when we conducting perception in videos with complex outdoor background and high-motion frames, PASS-P significantly beats these baselines with a much lower MPJPE, especially for the videos captured by moving cameras. Note that if we enable the op-

Method	Indoor, Fixed Camera				Outdoor, Moving Camera		
	Talking	Eating	Greeting	Phoning	Photo	Walk	WalkDog
PoseNet	57.68	53.29	52.54	53.81	54.95	52.05	57.26
VNect	77.82	65.36	72.74	69.49	94.11	72.96	82.47
Skip-CONV	63.81	69.39	66.37	64.72	66.65	65.74	63.53
MHP	49.97	55.68	52.37	51.54	52.84	51.54	49.12
MoVNect	82.42	77.14	82.39	100.85	109.52	86.23	96.33
PASS-P ($\mathcal{R}_p$: 39.06%)	**49.01**	54.05	**51.69**	**50.79**	**52.38**	**51.04**	**48.04**
PASS-F ($\mathcal{R}_p$: 42.48%)	**49.31**	**53.14**	**52.05**	**50.68**	**52.65**	**50.49**	**47.97**

Table 2: Comparison of environment sensitivity in 3D human pose estimation under diverse video settings. The key metric is MPJPE (mm), which is the lower, the better.

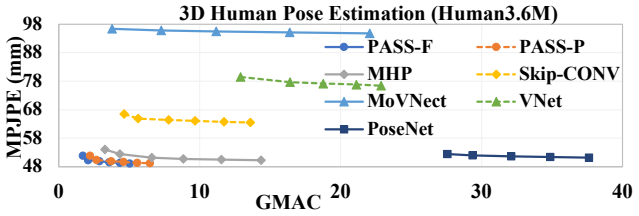

Figure 4: Comparison of MPJPE under different MAC count, where PASS outperforms existing methods.

tional backbone fine-tuning, our PASS-F can consistently achieve better performance over all the baselines. Without compromising accuracy, PASS-F holds a higher patch skip rate over PASS-P. This phenomenon indicates that PASS well adapts to the camera-moving cases, where PASS can automatically yield proper skip strategies and flexibly adjust scheduling configuration to match the high-motion frames.

Accuracy-computation tradeoff. We inspect whether increasing computational MAC count can yield higher model accuracy (*i.e.,* lower MPJPE), when using PASS and the five baselines. The MAC count is adjusted by changing the network structure. The comparison between PASS and the five baselines can be best understood by checking Figure. 4, which reports the MPJPE with different MAC count. PASS further reduces the MPJPE value with a moderate increase of MAC count, while other baselines still hold a higher MPJPE even under large MAC count. For example, Skip-CONV yields 64.09 MPJPE under 9.67 GMAC count, which is $1.92\times$ higher than the worst case of PASS (49.21 MPJPE under 6.46 GMAC). Consistently, the PASS-F scheme with additional backbone fine-tuning holds higher performance than the PASS-P scheme with gate pre-training solely. This comparison explicitly demonstrates the superiority of PASS.

Multiple Object Tracking

Experimental setup and metrics. We conduct experiments on the challenging Multiple Object Tracking (MOTChallenge) benchmark (MOTChallenge 2023; Milan et al. 2016), which is comprised of extensive high-resolution (1920×1080) video clips, with up to 128 real-time tracks. Following the most critical setting of JDE (Wang et al. 2020), we use the *Multiple Object Tracking Accuracy* (MOTA) (Wang et al. 2020), *Identification F-Score* (IDF1) (Ristani et al. 2016), *Identity Precision* (IDP) (Habibian et al. 2021) and *Identity Switches* (IDs) (Bernardin and Stiefelhagen 2008) as major evaluation metrics. Therefore, we cover the performance of multi-tracking accuracy (*i.e.,* MOTA, IDF1, IDP), prediction stability (*i.e.,* IDs), computation saving (*i.e.,* MAC), and processing speed (*i.e.,* FPS). Both two schemes of PASS-P and PASS-F are contained.

Comparison to SOTA methods. We compare PASS with six SOTA baselines: FairMOT (Zhang et al. 2021), High-Resolution Network (HRNet) (Wang et al. 2021b), RegNet (Radosavovic et al. 2020), HarDNet (Chao et al. 2019), Feature Pyramid Network (FPN) (Lin et al. 2017) and DLA (Zhou, Koltun, and Krähenbühl 2020). Table 3 reports the comparison with SOTA methods on the MOTChallenge dataset. The average patch skip rate $\mathcal{R}_p$ achieved by PASS-P and PASS-F are 54.25% and 59.17%, respectively. As to the multi-tracking accuracy, PASS outperforms the SOTA FairMOT with higher MOTA and IDF1 scores, indicating that patch skip can effectively reduce MAC count for high-FPS processing while still preserving good inference accuracy. Meanwhile, PASS simultaneously improves IDP and reduces IDs by correctly capturing temporal redundancy across frames, indicating the gates inside PASS can precisely locate each identity and yield stable tracking performance. Consistently, PASS significantly outperforms all the baselines with less MAC count, thus providing a much higher FPS.

Environmental sensitivity. We inspect the performance of PASS under different video settings, including with four kinds of venues (*i.e.,* square, pedestrian, intersection and market), two camera angles (*i.e.,* low and elevated) and two camera viewpoints (*i.e.,* fixed and moving). Table 4 reports the comparison with SOTA methods by checking the key metrics of MOTA. We can observe that PASS outperforms the baselines in all cases. As to the camera-fixed cases, PASS holds the highest tracking accuracy in different time of the day. For example, PASS can precisely track the crowded people on a night pedestrian from the elevated view, which is captured by a surveillance camera on the street light. Meanwhile, as to the more complex videos taken by moving cameras, PASS provides more prominent advantages over the

Method	MOTA ↑	IDF1 ↑	IDP ↑	IDs ↓	GMAC ↓	Mem. (M) ↓	FPS ↑
RegNetY-4.0GF	66.7%	70.4%	73.2%	781	3.86	103.84	17
HarDNet-85	68.2%	76.2%	77.9%	559	4.98	113.32	14
FPN	65.8%	70.5%	72.5%	708	6.55	126.61	9
DLA-34	72.3%	73.9%	75.1%	594	6.13	123.05	12
HRNet-W32	68.6%	75.7%	76.8%	604	3.37	99.69	20
FairMOT	83.9%	83.3%	86.3%	572	2.37	91.24	31
PASS-P	**84.1%**	**83.5%**	**86.6%**	**547**	**1.51**	**52.61**	**37**
PASS-F	**84.5%**	**83.6%**	**87.4%**	**535**	**1.47**	**49.12**	**39**

Table 3: Comparison with SOTA in multiple object tracking.

Method	Fixed Camera		Moving Camera (high-motion frames)	
	People walking around a large square	Night pedestrian street from elevated view	Filmed from a bus on busy intersections	Forward moving camera in a busy market
RegNetY-4.0GF	65.79%	66.23%	63.64%	62.67%
HarDNet-85	67.30%	67.39%	65.57%	64.68%
FPN	65.20%	65.06%	61.51%	61.01%
DLA-34	71.82%	71.27%	68.71%	68.62%
HRNet-W32	68.23%	67.76%	65.05%	65.30%
FairMOT	83.36%	83.40%	79.00%	80.52%
PASS-P ($\mathcal{R}_p$: 54.25%)	**84.34%**	**84.05%**	**82.27%**	**82.78%**
PASS-F ($\mathcal{R}_p$: 59.17%)	**84.43%**	**84.55%**	**82.72%**	**82.76%**

Table 4: Comparison of environment sensitivity in multiple object tracking with key metric of MOTA (the higher, the better).

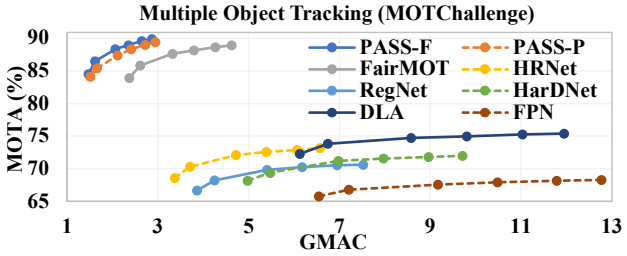

Figure 5: Comparison of MOTA scores under different MAC count, where PASS outperforms existing methods.

existing methods. For example, when moving forward in a busy market with multiple persons quickly walking past, PASS can correctly mark the bounding box of each person in real-time, thus providing a high-FPS processing results.

Accuracy-computation tradeoff. We also inspect how increasing computational MAC count impacts the model accuracy, when using PASS and the six baselines. The MAC count is adjusted by changing the network structure. The comparison between PASS and the six baselines can be quickly understood by checking the results in Figure. 5, which reports the MOTA scores with different MAC count. We can observe that our PASS achieves the best trade-off between model accuracy and MAC count, while other baselines either requires more computational overhead or degrades the model accuracy. Still, PASS-F scheme holds higher performance than PASS-P. These results clearly verifies the performance advantages of PASS.

Conclusion

This work reveals that exploiting temporal redundancy in video streams is a promising way to implement efficient real-time video perception systems on edge devices. We abstract away the computation saving problem from video perception tasks and propose a task-independent acceleration methodology that can generalize to different runtime environments. Following this principle, we develop three new quality-determining objectives for system design and present the *Patch Automatic Skip Scheme* (PASS) to support diverse video perception settings by decoupling acceleration and tasks. PASS equips each convolution layer with a learnable gate to selectively determine which patches could be safely skipped without compromising model accuracy. The gate is optimized via a tough self-supervisory procedure and holistically learns high-level semantics to distinguish similarity and difference across frames. The lightweight gate is compatible with commodity edge devices and can serve as a plug-and-play module to enable patch-skippable networks. Evaluations show that PASS explicitly outperforms SOTA solutions and can be applied to diverse downstream tasks.

Acknowledgements

This research was supported by fundings from the Key-Area Research and Development Program of Guangdong Province (No. 2021B0101400003), Hong Kong RGC Research Impact Fund (No. R5060-19), General Research Fund (No. 152203/20E, 152244/21E, and 152169/22E), the National Natural Science Foundation of China (61872310), and Shenzhen Science and Technology Innovation Commission (JCYJ20200109142008673).

References

Abati, D.; Tomczak, J.; Blankevoort, T.; Calderara, S.; Cucchiara, R.; and Bejnordi, B. E. 2020. Conditional Channel Gated Networks for Task-Aware Continual Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3930–3939.

Andriluka, M.; Pishchulin, L.; Gehler, P. V.; and Schiele, B. 2014. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3686–3693.

Balntas, V.; Riba, E.; Ponsa, D.; and Mikolajczyk, K. 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Proceedings of the British Machine Vision Conference 2016 (BMVC)*.

Bei, X.; Yang, Y.; and Soatto, S. 2021. Learning Semantic-Aware Dynamics for Video Prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 902–912.

Bengio, Y.; Léonard, N.; and Courville, A. C. 2013. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. *arXiv preprint*, abs/1308.3432.

Bernardin, K.; and Stiefelhagen, R. 2008. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *EURASIP J. Image Video Process.*, 2008.

Bu, X.; Peng, J.; Yan, J.; Tan, T.; and Zhang, Z. 2021. GAIA: A Transfer Learning System of Object Detection That Fits Your Needs. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 274–283.

Cai, H.; Gan, C.; Wang, T.; Zhang, Z.; and Han, S. 2020. Once-for-All: Train One Network and Specialize it for Efficient Deployment. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chao, P.; Kao, C.; Ruan, Y.; Huang, C.; and Lin, Y. 2019. HarDNet: A Low Memory Traffic Network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), ICCV*, 3551–3560. IEEE.

Choi, S.; Choi, S.; and Kim, C. 2021. MobileHumanPose: Toward Real-Time 3D Human Pose Estimation in Mobile Devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2328–2338.

Chollet, F. 2017. Xception: Deep Learning with Depthwise Separable Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1800–1807.

Dasari, M.; Kahatapitiya, K.; Das, S. R.; Balasubramanian, A.; and Samaras, D. 2022. Swift: Adaptive Video Streaming with Layered Neural Codecs. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 103–118. Renton, WA: USENIX Association. ISBN 978-1-939133-27-4.

Ding, D.; Ma, Z.; Chen, D.; Chen, Q.; Liu, Z.; and Zhu, F. 2021. Advances in Video Compression System Using Deep Neural Network: A Review and Case Studies. *Proceedings of IEEE*, 109(9): 1494–1520.

Fang, B.; Zeng, X.; and Zhang, M. 2018. NestDNN: Resource-Aware Multi-Tenant On-Device Deep Learning for Continuous Mobile Vision. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom)*, 115–127.

Ghodrati, A.; Bejnordi, B. E.; and Habibian, A. 2021. FrameExit: Conditional Early Exiting for Efficient Video Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 15608–15618.

Habibian, A.; Abati, D.; Cohen, T. S.; and Bejnordi, B. E. 2021. Skip-Convolutions for Efficient Video Processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2695–2704.

Hwang, D.; Kim, S.; Monet, N.; Koike, H.; and Bae, S. 2020. Lightweight 3D Human Pose Estimation Network Training Using Teacher-Student Learning. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 468–477.

Ionescu, C.; Papava, D.; Olaru, V.; and Sminchisescu, C. 2014. Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(7): 1325–1339.

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical Reparameterization with Gumbel-Softmax. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kondratyuk, D.; Yuan, L.; Li, Y.; Zhang, L.; Tan, M.; Brown, M.; and Gong, B. 2021. MoViNets: Mobile Video Networks for Efficient Video Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 16020–16030.

Li, H.; Wu, Z.; Shrivastava, A.; and Davis, L. S. 2021. 2D or not 2D? Adaptive 3D Convolution Selection for Efficient Video Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6155–6164.

Lin, T.; Dollár, P.; Girshick, R. B.; He, K.; Hariharan, B.; and Belongie, S. J. 2017. Feature Pyramid Networks for Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 936–944. IEEE Computer Society.

Liu, Q.; Ramanathan, V.; Mahajan, D.; Yuille, A. L.; and Yang, Z. 2021a. Weakly Supervised Instance Segmentation for Videos With Temporal Mask Consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 13968–13978.

Liu, Z.; Chen, H.; Feng, R.; Wu, S.; Ji, S.; Yang, B.; and Wang, X. 2021b. Deep Dual Consecutive Network for Human Pose Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 525–534.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54, 1273–1282.

Mehta, D.; Sridhar, S.; Sotnychenko, O.; Rhodin, H.; Shafiei, M.; Seidel, H.; Xu, W.; Casas, D.; and Theobalt, C. 2017. VNect: real-time 3D human pose estimation with a single RGB camera. *ACM Trans. Graph.*, 36(4): 44:1–44:14.

Melas-Kyriazi, L. 2021. Do You Even Need Attention? A Stack of Feed-Forward Layers Does Surprisingly Well on ImageNet. *arXiv preprint*, abs/2105.02723.

Milan, A.; Leal-Taixé, L.; Reid, I. D.; Roth, S.; and Schindler, K. 2016. MOT16: A Benchmark for Multi-Object Tracking. *arXiv preprint*, abs/1603.00831.

Moon, G.; Chang, J. Y.; and Lee, K. M. 2019. Camera Distance-Aware Top-Down Approach for 3D Multi-Person Pose Estimation From a Single RGB Image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10132–10141. IEEE.

MOTChallenge. 2023. Multiple Object Tracking Benchmark. https://motchallenge.net/. Accessed: 2023-01-01.

Narayanan, A.; Zhang, X.; Zhu, R.; Hassan, A.; Jin, S.; Zhu, X.; Zhang, X.; Rybkin, D.; Yang, Z.; Mao, Z. M.; Qian, F.; and Zhang, Z. 2021. A variegated look at 5G in the wild: performance, power, and QoE implications. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM)*, 610–625. ACM.

NVIDIA. 2023. Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nano-developer-kit. Accessed: 2023-01-01.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. DeepWalk: online learning of social representations. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 701–710.

PyTorch. 2023. PyTorch: An Open Source Machine Learning Framework. https://pytorch.org/. Accessed: 2023-01-01.

Radosavovic, I.; Kosaraju, R. P.; Girshick, R. B.; He, K.; and Dollár, P. 2020. Designing Network Design Spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10425–10433. Computer Vision Foundation / IEEE.

Ren, M.; Pokrovsky, A.; Yang, B.; and Urtasun, R. 2018. SBNet: Sparse Blocks Network for Fast Inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8711–8720. Computer Vision Foundation / IEEE Computer Society.

Ristani, E.; Solera, F.; Zou, R. S.; Cucchiara, R.; and Tomasi, C. 2016. Performance Measures and a Data Set for Multi-target, Multi-camera Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 9914, 17–35.

Tolstikhin, I. O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; Lucic, M.; and Dosovitskiy, A. 2021. MLP-Mixer: An all-MLP Architecture for Vision. *arXiv preprint*, abs/2105.01601.

Trockman, A.; and Kolter, J. Z. 2022. Patches Are All You Need? *arXiv Preprint*, abs/2201.09792.

Tsukada, M.; Kondo, M.; and Matsutani, H. 2020. A Neural Network-Based On-Device Learning Anomaly Detector for Edge Devices. *IEEE Trans. Computers*, 69(7): 1027–1044.

Verelst, T.; and Tuytelaars, T. 2021. BlockCopy: High-Resolution Video Processing with Block-Sparse Feature Propagation and Online Policies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 5138–5147. IEEE.

Wang, H.; Cong, Y.; Litany, O.; Gao, Y.; and Guibas, L. J. 2021a. 3DIoUMatch: Leveraging IoU Prediction for Semi-Supervised 3D Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 14615–14624.

Wang, J.; Sun, K.; Cheng, T.; Jiang, B.; Deng, C.; Zhao, Y.; Liu, D.; Mu, Y.; Tan, M.; Wang, X.; Liu, W.; and Xiao, B. 2021b. Deep High-Resolution Representation Learning for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(10): 3349–3364.

Wang, Z.; Zheng, L.; Liu, Y.; Li, Y.; and Wang, S. 2020. Towards Real-Time Multi-Object Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 12356, 107–122. Springer.

Yeo, H.; Chong, C. J.; Jung, Y.; Ye, J.; and Han, D. 2020. NEMO: enabling neural-enhanced video streaming on commodity mobile devices. In *Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom)*, 28:1–28:14. ACM.

Zhang, Y.; Wang, C.; Wang, X.; Zeng, W.; and Liu, W. 2021. FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking. *Int. J. Comput. Vis.*, 129(11): 3069–3087.

Zhou, Q.; Yu, C.; Wang, Z.; Qian, Q.; and Li, H. 2021. Instant-Teaching: An End-to-End Semi-Supervised Object Detection Framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4081–4090.

Zhou, X.; Koltun, V.; and Krähenbühl, P. 2020. Tracking Objects as Points. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 12349, 474–490. Springer.

Zhu, F.; Gong, R.; Yu, F.; Liu, X.; Wang, Y.; Li, Z.; Yang, X.; and Yan, J. 2020. Towards Unified INT8 Training for Convolutional Neural Network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1966–1976. Computer Vision Foundation / IEEE.