Decode-What-Matters: Frame-Level Parallel Generative Decoding to Accelerate Large-Scale Video Analytics

Xiaokun Wang*
Nanjing University
Nanjing, China
xiaokun.wang@smail.nju.edu.cn

Yuting Yan*
Nanjing University
Nanjing, China
yuting.yan@smail.nju.edu.cn

Sheng Zhang†
Nanjing University
Nanjing, China
sheng@nju.edu.cn

Andong Zhu
Nanjing University
Nanjing, China
andongzhu@smail.nju.edu.cn

Ning Chen
Soochow University
Suzhou, China
ningc@suda.edu.cn

Yu Chen
Nanjing University
Nanjing, China
yu.chen@smail.nju.edu.cn

Zhuzhong Qian
Nanjing University
Nanjing, China
qzz@nju.edu.cn

Sanglu Lu
Nanjing University
Nanjing, China
sanglu@nju.edu.cn

Yu Liang
Nanjing Normal University
Nanjing, China
liangyu@njnu.edu.cn

Abstract

Video analytics pipelines (VAPs) have been a paradigm for large-scale video analytics. Due to temporal redundancy in video, frame filtering is widely used in VAPs to reduce analysis workload. However, existing works overlook a limitation: while inference operates only on selected frames, decoders must still process many redundant frames due to codec dependencies, leading to over-decoding trap. This limitation stems from the reference-based design in modern codecs, which require decoding preceding frames to reconstruct any selected one. As a result, over-decoding has become the practical bottleneck in VAPs using modern decoders, highlighting a critical but under-explored problem. To address this issue, we propose ParaDeco, a high-throughput video analytics framework featuring a novel frame-level parallel generative decoder. Unlike traditional decoders, ParaDeco adopts a *decode-what-matters* approach with decoupled frame dependencies. To decode arbitrary frames independently, ParaDeco generates frame-wise features as standalone skeletons using compressed video metadata, then predicts pseudo frames maintaining semantic consistency with original frames. Moreover, ParaDeco identifies which frames truly matter for analysis via delicate contribution-based frame filtering. We implement ParaDeco on a cloud server and evaluate it on large-scale real-world video datasets. Our experimental results show that ParaDeco achieves a 2.76× speedup on average compared to state-of-the-art VAPs.

CCS Concepts

• **Information systems** → **Multimedia information systems**; • **Computing methodologies** → *Parallel computing methodologies*.

Keywords

Video Analytics; Compressed Domain; Neural Video Decoder

ACM Reference Format:
Xiaokun Wang, Yuting Yan, Sheng Zhang, Andong Zhu, Ning Chen, Yu Chen, Zhuzhong Qian, Sanglu Lu, and Yu Liang. 2025. Decode-What-Matters: Frame-Level Parallel Generative Decoding to Accelerate Large-Scale Video Analytics. In *Proceedings of the 33rd ACM International Conference on Multimedia (MM '25), October 27–31, 2025, Dublin, Ireland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3746027.3755186

1 Introduction

With the surge in video data from diverse sources [29], scalable and efficient video analytics are essential for applications including security [5], traffic management [6], and smart cities [36]. Numerous works [7, 17, 47, 56, 57] have deployed VAPs for efficient video analytics, which contain several basic stages, including video decoding, inference, and result storage. Specifically, it decodes compressed video data into raw frames, applies a deep learning network (DNN) model to each frame, and saves the results.

To accelerate VAPs, filtering has become a typical practice that removes similar frames in temporally redundant video to reduce workload and boost throughput. Unlike general-purpose acceleration techniques, such as model pruning [16, 27], specialization [21, 39], or hardware solutions [32, 42, 43] that target specific modules and may incur extra cost or accuracy loss, filtering directly leverages inherent video redundancy for end-to-end optimization, delivering cascading benefits across the pipeline. Specifically, filtering acceleration is *multiplicative*; for instance, a 90% filtering ratio can bring up to a 10× improvement in throughput, as only 1/10 of frames are selected to be processed. As a low-cost, platform-independent technique tapping into essential characteristics of video content, filtering methods are widely adopted for efficiency. For example,

*Both authors contributed equally to this research.
†Corresponding author.

MM '25, Dublin, Ireland

ACM ISBN 979-8-4007-2035-2/2025/10
https://doi.org/10.1145/3746027.3755186

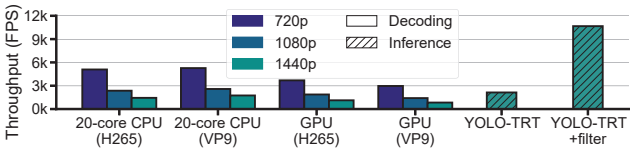


Figure 1: Throughput of decoding and inference in VAP.

difference-based methods [14, 24, 48] filter out frames with minimal changes, while network-based methods [7, 56, 60] use lightweight models to discard frames that reuse prior analysis results.

Over-Decoding Trap. While filtering effectively reduces inference load to accelerate VAPs, we observe that the decoding stage remains largely untouched and bottlenecks the system. Since most filtering methods [9, 48, 56, 60] depend on pixel-domain information and thus operate after decoding, they offer no reduction in decoding workload. Frustratingly, even advanced VAPs such as COVA [17] and PacketGame [57], which implement pre-decoding filtering, still suffer from *over-decoding*. For instance, although COVA selects only 1.2% of frames for analytics using object tracks, it still requires decoding 59.1% of frames due to codec dependencies (§ 2.1), resulting in only a 1.7× improvement in decoding throughput, far behind the 83.3× acceleration achieved in inference. This mismatch renders decoding the dominant bottleneck in practical VAP deployments, which is particularly problematic in large-scale scenarios, as decoding cost scales with video volume rather than analytics demand. The root cause of over-decoding lies in the reference-based design of modern codecs, which require decoding preceding frames to reconstruct any selected one. This inherent codec constraint causes a fundamental mismatch between the optimized inference stage and the rigid decoding process, highlighting a practical and under-addressed limitation in current VAPs.

To address the over-decoding trap, we propose ParaDeco, a high-throughput video analytics framework based on a *decode-what-matters* approach. ParaDeco builds on two key insights: (1) compressed-domain metadata (i.e., data stored in encoded video) contains high-level abstractions of content, particularly object contours and motion, and can be extracted rapidly; (2) modern codecs apply serial decoding to reconstruct pixel-level vision details, however, video analytics typically targets semantic-level understanding. Therefore, it motivates us to shift from the traditional serial decoding towards a parallel, independent and machine-centric way.

Leveraging the above insights, ParaDeco enables frame-level parallel generative decoding for large-scale video analytics, addressing key challenges through three core components:
(1) *Frame Skeleton Generator.* Compressed metadata provides content abstractions, but suffers from inconsistent clarity across frames due to codec design and pixel jitter. To address this, ParaDeco carefully decouples dependencies and enhances metadata to produce clear, independent skeletons that capture essential contours and motion, serving as reliable inputs for parallel frame reconstruction.
(2) *Pseudo-frame Predictor.* Predicting accurate pseudo frames from sparse metadata is challenging, as trivial predictions risk semantic discrepancy and harm downstream analytics. To tackle this, ParaDeco designs a parallelized pseudo-frame predictor composed of two lightweight DNNs for feature fusion and image repainting, learning long-term video similarity and generating machine-centric pseudo frames that retain semantics and support scalable inference.

Table 1: Preliminary study of filter-before-decoding methods.

Method	Selected (%)	Decoded (%)	Throughput (FPS)
Decoder Only	100	100	3,670
Uniform Filter	10	96.0	4,077
COVA [17]	1.4	59.1	6,220

(3) *Contribution-based Filter.* Unfortunately, prior filtering methods are incompatible with ParaDeco due to their reliance on pixel information. Using inter-frame reference in metadata, ParaDeco devises a novel task-independent frame filter to select frames with high contribution, typically with new objects. Thus, ParaDeco decodes only analysis-relevant frames in parallel, enabling scalability.

We evaluate ParaDeco across four analysis tasks using large-scale real-world video datasets in diverse scenarios. Experimental results show that ParaDeco achieves an average 2.76× throughput over state-of-the-art decoding accelerated VAPs, and 4.41× over VAPs with modern decoders with modest accuracy loss. Our key contributions can be concluded as follows:

- We identify the over-decoding bottleneck in VAPs and reveal the underlying cause, i.e., the decoded frames greatly exceed the frames to be analyzed due to reference-based codecs, which limits overall pipeline throughput.
- We introduce a novel generative frame reconstruction technique that uses compressed metadata to generate standalone frame skeletons and predict pseudo frames in parallel, eliminating redundant decoding.
- We propose ParaDeco, a novel high-throughput video analytics framework that adopts a *decode-what-matters* approach, achieving frame-level parallel generative decoding and contribution-based, precise filtering.
- Experimental results show that ParaDeco achieves 1.59×–4.58× speedup compared to state-of-the-art VAPs, with compatibility for various external settings like analysis tasks.

2 Background and Motivation

2.1 Over-Decoding Trap in VAPs

Filter-after-decoding. As the vast majority of VAPs operate filtering after frame decoding, the decoder processes every frame regardless of whether it will be discarded by subsequent filtering stages. As a result, decoding, rather than accelerated inference, becomes the system's bottleneck. As illustrated in Fig. 1, even with predominantly static video content that enables faster decoding, the decoding throughput is 2.6×–9.3× slower than the inference throughput, using Yolov7 [44] accelerated by TensorRT [42] and a typical filter InFi [56] on a 20-core CPU; and 3.6×–11.8× slower using NVDEC [30] on an Nvidia 3090 GPU. This bottleneck is prevalent with modern codecs like H.265 [13] and VP9 [12, 46].

Filter-before-decoding. A natural approach to avoid the over-decoding trap is to filter frames before decoding, which promises to reduce the decoded frames. However, our preliminary studies reveal that *the number of decoded frames still far exceeds the selected frames.* As shown in Table 1, although the uniform filter selects only 10% of frames, 96% of frames are still decoded, resulting in a modest improvement of decoding throughput by only 3,670 to 4,077 FPS, which significantly lags behind the 10× inference speedup. Furthermore, we deploy a state-of-the-art VAP, COVA [17], which

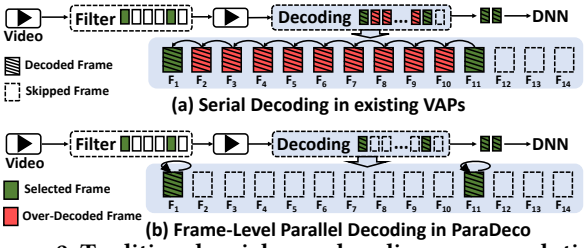


Figure 2: Traditional serial over-decoding v.s. our solution.

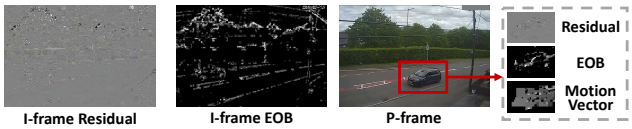


Figure 3: Visualized metadata including residual, EOB and motion vector, compared with pixel image.

selects only 1.2% anchor frames before decoding based on offline video. However, the decoder in COVA still decodes 59.1% of the frames, achieving only a 1.7× speedup compared to Origin, which is much slower than the 83.3× accelerated inference. Thus, existing filter-before-decoding methods failed to effectively reduce decoding workload, leaving the over-decoding bottleneck unresolved.

Reference-based Codecs. To understand the root cause of the over-decoding trap, we examine the design of modern codecs. Taking H.265 as a representative example (and codecs like VP9 and AV1 [3] follow similar principles), each frame is divided into macroblocks to optimize compression. Each block stores necessary data for prediction (e.g., motion vectors indicating offsets to reference blocks) and discrepancy (i.e., residuals) compared to actual pixels. While decoding, frames are reconstructed by adding predicted blocks and residuals. Based on prediction strategies, frames are categorized into I-, P-, and B-frames. I-frames are encoded independently using *intra-predicted* blocks from the same frame, while P- and B-frames use *inter-predicted* blocks referencing similar blocks in other frames. To mitigate error propagation, I-frames are periodically inserted to segment videos into groups of pictures (GOPs).

Inherent Limitations of Codecs. While such reference enhances compression efficiency, decoding process in modern codecs is inherently *serial*, as the decoding of each frame depends on preceding decoded frames. As illustrated in Fig. 2(a), to decode F_{11}, it needs to decode F_1 to F_{10}, the majority of which are over-decoded and not used for analysis. In contrast, DNNs can infer only the selected frames without such dependencies, significantly increasing throughput as the filtering ratio grows. Hence, existing filter-before-decoding VAPs, such as COVA [17] and PacketGame [57], still suffer from over-decoding due to their reliance on traditional decoders.

To address the over-decoding trap, we adopt a ***decode-what-matters*** mechanism that decodes only selected crucial frames, as shown in Fig. 2(b). Each frame is independently decodable, enabling frame-level parallel decoding, eliminating over-decoding bottleneck and unlocking higher throughput and parallelism for VAPs.

2.2 Potential for Frame-level Parallism

Multi-level Metadata Information. As video encoders perform predictive compression, the resulting metadata in compressed domain involves multi-level abstraction of video, such as residuals and motion vectors. Residual stores discrepancies between predicted and actual blocks, thereby capturing object contours and fine textures that are typically difficult to predict, as shown in Fig. 3. Further, we identify the end of block (EOB), which marks the last non-zero residual coefficient in a block (e.g., 0-64 for an 8x8 block), as a block-level aggregated residual that reflects coarse-grained object contours with less noise. In addition, motion vectors encode the moves of dynamic objects, depicting their positions and directions. Furthermore, as compressed metadata requires only partial decoding of sparse signals and is frame-independent, it can be processed in parallel with significantly lower computational cost [4, 58].

Figure 4: The analysis results of raw frames and the DNN-reconstructed pseudo frames in two scenes.

Observation #1: The metadata of each frame already contains a relatively accurate perception of the video content, especially the contours of objects and motion information.

Machine-centric Prediction. Traditional codecs focus on short-term redundancy within GOPs and thus struggle to exploit long-term temporal similarity, such as static backgrounds or recurring objects in videos. This limitation inspires us to design a DNN-based frame predictor with the capability to learn long-range semantic patterns and predict selected frames in parallel using standalone rich metadata. As video analytics tasks focus on category and semantic information, the predictor could be machine-centric, optimized for analytics rather than human viewing. In Fig. 4, we train and evaluate an image translation model, pix2pix [18], to predict the frame for two scenes respectively, using EOB of I-frames as input and pixel images as ground truth. The result shows that the predictor learns the high-level semantic features and regenerates a pseudo frame with similar object contours and textures using the contour information. Moreover, most objects are detected successfully by the pre-trained YOLOv7-e6e [44] model with lower confidence.

Observation #2: By exploiting long-range, machine-observable semantic redundancy in videos, the neural-based predictor has the potential to predict pseudo frames in batches using metadata, retaining essential class and semantic fidelity for downstream analytics.

Design Insight. These observations shed light on frame-level parallelism in decoding. An intuitive idea is to parallelize frame decoding by using network-based generative prediction solely on informative and independent metadata, enabling a *decode-what-matters* approach to eliminate the over-decoding bottleneck.

3 Design

3.1 System Overview

ParaDeco addresses over-decoding bottleneck in large-scale video analytics through a core principle of *decode-what-matters*, which integrates frame-level parallel generative decoding and a novel contribution-based frame filter. The architecture is shown in Fig. 5.

To enable frame-level parallel decoding, ParaDeco ensures that each frame relies solely on its own metadata (*frame skeleton*) to

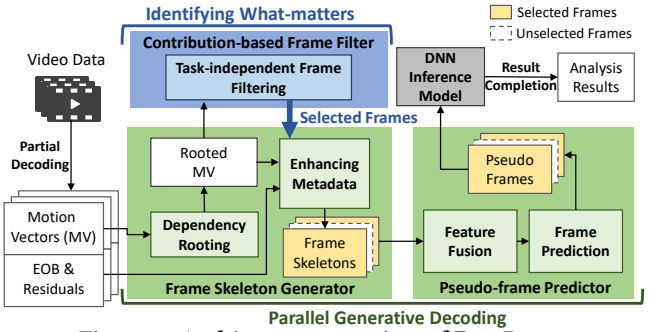


Figure 5: Architecture overview of ParaDeco.

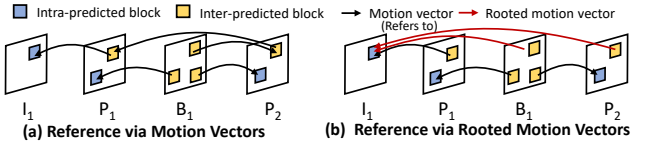


Figure 6: Reference between the macroblocks, denoted by motion vectors and rooted motion vectors.

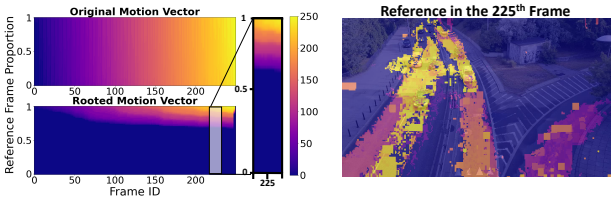
decode itself into a *pseudo frame*. Specifically, the **Frame Skeleton Generator** (§3.2) extracts metadata from compressed video through partial decoding, and inter-frame references are relaxed by identifying the root of motion vector dependency. Leveraging the rooted motion vector, the metadata is temporally enhanced to generate the decoupled, independent frame skeletons for each frame, which eliminates inter-frame dependency and facilitates parallel decoding. The **Pseudo-Frame Predictor** (§3.3) employs a DNN model to learn high-level features and long-term semantic information in video for frame reconstruction. Specifically, it fuses multi-level skeleton features and predicts pseudo frames in a machine-centric manner, which are fed into inference models for analysis results. Notably, both modules are parallelized for each frame, ensuring that only the selected frames are decoded, thereby preventing over-decoding. To identify the most critical frames for analysis, the **Contribution-based Frame Filter** (§3.4) measures each frame's contribution using the reference relationship revealed by rooted motion vector. Hence, it selects only the critical frames for reconstruction, which significantly contribute to the analysis.

3.2 Frame Skeleton Generator

3.2.1 *Dependency Rooting in Metadata.*

Metadata Extraction. While compressed video contains rich metadata, extracting unnecessary metadata increases decoding overhead, hindering large-scale processing. Inspired by prior vision research [25, 38, 62], we identify object edges and contours as critical semantic and high-level feature carriers. Leveraging observations (§2.2), we extract object contours from residuals and EOB. We also incorporate motion vectors, which encode object motion and inter-frame similarity, essential for frame filtering and reconstruction. Following [17], we perform partial decoding to rapidly extract selected metadata by executing only a few decoding steps.

Rooting Motion Vectors. Modern codecs use motion vectors to propagate similar blocks from reference frames, optimizing compression by reusing content from nearby frames. However, it also introduces long, multi-hop serial dependencies between frames and primarily captures short-term similarity, thereby failing to exploit



(a) Reference Proportion (b) Rooted Reference

Figure 7: Detailed reference frame distribution of each frame in a GOP, with frame IDs distinguished by color.

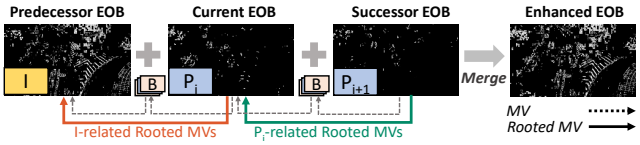


Figure 8: Process of metadata enhancing.

long-term temporal redundancy critical for video analytics. To relax inter-frame dependencies and uncover long-term similarity, we propose tracing each block's root reference, as depicted in Fig. 6. Intra-predicted blocks serve as root blocks, as they do not rely on the others. For example, in Fig.6(a), an inter-predicted block in frame B_1 forms a multi-hop dependency chain (B_1->P_2->P_1->I_1). Similar to path compression in the Union-Find algorithm, the dependency can be shrunk into one-hop (B_1->I_1) as a rooted motion vector in Fig. 6(b). As frames are traversed in reference (i.e., decoding) order, the rooted motion vectors can be computed efficiently in one pass.

Fig. 7(a) compares reference frames before and after motion vector rooting, where each frame ID is color-coded. Before rooting, references are spread across nearby frames, while after rooting, most blocks converge to a few key frames, revealing long-term similarity. For example, in frame 225, nearly 70% of blocks reference I-frame (frame 0), while the rest primarily reference four other key frames, as shown by distinct color bands. As visualized in Fig. 7(b), background regions predominantly reference the I-frame, indicating long-range static content propagation, whereas dynamic regions like roads retain shorter references due to frequent intra-prediction. Hence, rooted motion vectors decouple multi-hop dependencies and expose key reference, facilitating metadata processing and filtering.

3.2.2 *Enhancing Metadata as Frame Skeleton.*

Problem and Goal. While metadata contains rich information, the clarity of residual and EOB is inconsistent across frames. First, since background and static objects change little, only I-frame reflects all object contours with intra-frame prediction, while other frames mainly contain the dynamics. Second, the clarity is also affected by random pixel jitter and prediction errors. To address this, we enhance metadata to construct independent and clean frame skeleton.

Temporal Enhancement. As motion vectors indicate reference between similar blocks across frames, intuitively, they can be used to aggregate and enhance the contours in residuals and EOBs. Therefore, we enhance residuals and EOBs of a frame through bidirectional temporal propagation to get complete object contours. Fig. 8 depicts the process of enhancing EOB as an example.

(1) Propagation from Predecessor. In Fig. 7(b), rooted motion vectors of static objects and background regions typically point to preceding frames (often I-frames) due to minimal change over time. Hence,

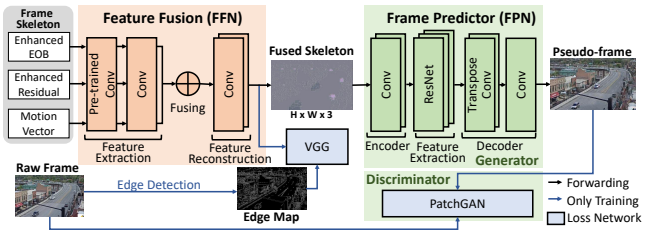


Figure 9: The architecture of pseudo-frame predictor.

by reversing these rooted vectors, static contour information can be propagated forward from the root frames to current frame.
(2) Propagation from Successor. Given the similarity between adjacent frames, we also enhance the current frame using its immediate successor. Specifically, we first locate P_i-related motion vectors in P_{i+1} by setting P_i as root and re-rooting motion vectors, and then propagate texture along them to enhance contours in P_i.

Frame Skeleton Generation. Through temporal enhancement, the residual and EOB of a frame contain complete contour information. We construct a frame skeleton using residuals, EOB, and motion vector magnitude and orientation, forming a 4-channel representation of structure and motion. These skeletons are then aggregated into a 4D tensor for subsequent pseudo-frame prediction.

3.3 Pseudo-Frame Predictor

3.3.1 *Frame Skeleton Fusion.*

Goal and Insights. The frame skeleton combines metadata at multiple semantic levels: motion vectors represent dynamic objects, while enhanced residuals and EOBs capture static structures and contours, making feature fusion essential. The goal is to *unify semantic information from different metadata types* to facilitate prediction. The task of frame skeleton fusion is analogous to multimodal image fusion, as both integrate heterogeneous semantic sources into a unified representation. This insight allows the use of various techniques, including convolutional neural network (CNN) based image fusion methods, for effective skeleton fusion.

Design. Based on this observation, we devise a lightweight CNN-based feature fusion network, named FFN, leveraging the similarity between multimodal image fusion [61] and frame skeleton fusion. Unlike multimodal image fusion, which operates in the pixel domain, FFN works with compressed domain data. FFN first extracts features from each level of frame skeleton using tied-weight convolutional layers, producing a 64-channel feature map. These features are then fused by elementwise maximization to retain salient object features. Finally, a 3-channel comprehensive feature image is reconstructed by the fused features, i.e., fused frame skeleton. To ensure the consistency of output and ground truth in high-dimensional features, we use a pre-trained VGG [40] as a loss network.

3.3.2 *Machine-Centric Frame Predictor.*

Goal and Insights. Using fused skeleton information, we aim to generate machine-centric pseudo-frames for inference tasks. Unlike the prediction to reconstruct pixel images, pseudo-frame prediction is designed for accurate analysis rather than human viewing. Additionally, the pseudo-frame should be generated in parallel from each frame's skeleton. To this end, we exploit an observation that the reconstruction is akin to an image repainting task, which aims to repaint comprehensive skeleton features into pixel-style images.

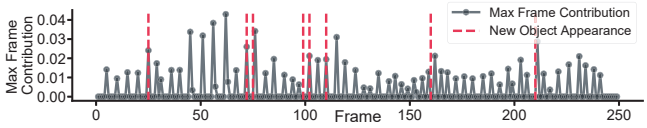


Figure 10: The maximum contribution of frames in a GOP, where I-frame is not shown with the value 0.97.

Design. We design a machine-centric frame predictor network, named FPN, based on pix2pix [18], a typical and efficient generative adversarial network (GAN) model for image repainting. FFP includes two components: (1) a generator that takes the fused skeleton as input and outputs the pseudo-frame, and (2) a discriminator that distinguishes the pseudo-frame from the real frame. The generator employs an encoder-decoder structure, as shown in Fig. 9. The encoder extracts the feature map from the fused skeleton via convolutional layers. Then we add skip-connections by ResNet [15] to extract and concatenate multi-scale features for global feature capturing. The decoder then executes transposed convolutions for upsampling and restores the original resolution. Like the typical design in pix2pix, the discriminator uses a PatchGAN structure to better capture local features. Since areas with objects are more crucial for analysis, we adjusted the loss measurement to generate machine-centric pseudo-frames. Specifically, motion vectors serve as a mask indicating the areas of moving objects, and a higher weight is assigned to the regions, ensuring the generator focuses more on objects rather than the background. The GAN model enables the predictor to produce realistic pseudo-frames, while the mask mechanism ensures better prediction for objects.

Collaborative Training. However, restoring fine-grained pixels from coarse-grained features with noise is challenging, requiring the effective union of skeleton fusion network and frame predictor. Since numerous works have found that edges are essential for revealing object features [25, 38, 62], they inspire us to leverage edge information as a bridge between FFN and FPN. Thus, we use a lightweight algorithm, Canny edge detection [37], to get an edge map from a pixel image. Specifically, the FFN uses edge maps as ground truth for feature fusion, while the FPN uses these maps alongside pixel images as input-output pairs. This collaborative training paradigm ensures input-output consistency, enabling more accurate pseudo-frame reconstruction with less training cost.

Scene Specialization. To best maintain the semantic consistency with original frames, we train a specialized pseudo-frame predictor (FFN and FPN) for each video scene, optimizing pseudo-frame generation to match the unique characteristics of the scene. This approach arises from the observation in (§2.2) that utilizing DNNs to capture long-term similarity features across videos within a scene improves the quality of video reconstruction. Note that since the model is lightweight, training can be finished in about 10 minutes, and FFN and FPN can be trained in parallel due to the collaborative training scheme. Similar to previous VAPs [2, 20, 49] that train specialized models for each video, training cost can be amortized for every subsequent query of different inference tasks.

3.4 Contribution-based Frame Filter

Problem and Goal. To filter redundant frames in analytics, the filtering method in ParaDeco must satisfy two key constraints. First, since the filtering operates before frame reconstruction, it must

Table 2: Tasks and metrics for evaluating ParaDeco.

Task	Description	Metric	Formula
Object Detection (OD)	Outputs category and bounding box.	F1 Score (F1)	$F1 = 2 \cdot TP/(2 \cdot TP + FP + FN)$
Binary Predicate (BP)	Identifies frames with a specific object class.	True Positive Rate (TPR)	$TPR = TP/(TP + FN)$
Counting (CT)	Count occurrences of a specific object class.	Relative Accuracy (RA)	$RA = 1 - \lvert y_{\text{pred}} - y_{\text{true}} \rvert / y_{\text{true}}$
Violation Detection (VD)	Detects vehicles violating traffic rules.	Precision (P)	$P = TP/(TP + FP)$

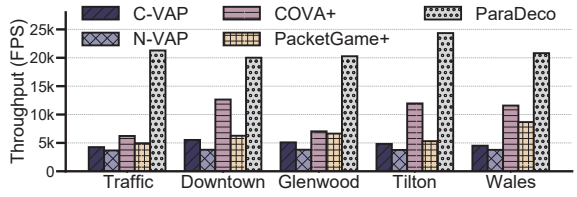


Figure 11: Comparison of throughput with other systems.

rely solely on compressed-domain metadata rather than pixel-level features. Second, given the multi-task nature of large-scale video analytics, the method should be task-independent, identifying redundant frames across diverse tasks without requiring task-specific retraining or deployment. However, existing frame filtering methods are incompatible with these requirements. Prior neural network-based approaches [7, 56] require task-specific retraining and consume many resources, while difference-triggered filtering [9, 48] requires pixel information. Therefore, ParaDeco needs a compressed-domain and task-independent frame filtering method.

Insights and Design. Inspired by [9, 56], we observe that task-independent filtering focuses on retaining frames that preserve semantic information for all objects, which can be achieved by including frames where new objects appear. To identify such frames in compressed domain, we leverage rooted motion vectors. Specifically, after the rooting process (§3.2.1), each block in a frame either serves as a root block or references root blocks in other frames. Root blocks, which are intra-predicted and generate content independently, are crucial for capturing significant changes that cannot be propagated from past frames. Thus, root blocks serve as a primary source of information for new or dynamic objects, which are later referenced to represent object motion in subsequent frames.

To identify frames strongly associated with new object appearances, containing more root blocks and contributing to subsequent frames, we adopt the *maximum contribution rate*. We define the contribution rate from frame A to frame B as the proportion of blocks in B rooted in A. The maximum contribution rate of a frame is its highest contribution to any subsequent frame. As shown in Fig. 10, frames with new object appearances exhibit elevated maximum contribution rates, especially P-frames that propagate root blocks. Notably, I-frames show even higher rates due to intra-predicted blocks. Based on this insight, we propose a simple yet effective task-independent frame selection method. First, we compute the maximum contribution for each frame in a GOP. Given a contribution threshold or desired filtering rate, we either select frames exceeding the threshold or discard those with lowest contributions. By exploiting long-term redundancy from frame reconstruction logic, our method is task-independent and requires no retraining.

Result Completion. As some frames are skipped in the analysis, the results have to be completed for these frames. The completion process essentially includes two methods [24, 56]: (1) the results of skipped frames are directly set to null; (2) the results of the nearest frame are reused, or deduced with simple calculation to apply to the skipped frame. In our work, we adopt the second method, which uses motion vectors to deduce the move of objects in skipped frames by the result of the nearest frame, requiring no pixel domain data.

Table 3: Performance of ParaDeco on four tasks.

Dataset	OD (F1)	BP (TPR)	CT (RA)	VD (P)
Traffic	0.769	1.000	0.998	0.861
Downtown	0.706	0.987	0.977	0.791
Glenwood	0.768	0.988	0.880	0.850
Tilton	0.749	0.963	0.819	0.892
Wales	0.820	0.936	0.978	0.719
Average	0.762	0.978	0.931	0.825

4 Evaluation

4.1 Implementation and Methodology

Implementation. ParaDeco is built on a cloud server with Ubuntu 20.04, equipped with a 20-core Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz, 216GB of DRAM, and an NVIDIA 3090 GPU. We implement ParaDeco using PyTorch [35] and FFmpeg [11], focusing on video decoding and DNN inference to ensure compatibility with large volumes of pre-existing video data. ParaDeco extracts metadata through lightweight partial decoding, leveraging a modified version of libavcodec [17]. All the DNN models in ParaDeco are optimized with TensorRT [42], ensuring high-performance execution.

Methodology. Following prior works [17, 23, 41], our datasets are derived from real-world live streams on YouTube [50–53, 55], captured by fixed cameras across diverse scenes, including highways, intersections, and streets. All videos are encoded in 720p, H.265 codec. Full DNN models such as YOLOv7-e6e [44] generate ground-truth labels, avoiding impractical manual annotation of hours of video. We evaluate ParaDeco on four fundamental video analytics tasks commonly used in downstream applications, with task details and metrics in Table 2. Our analytics primarily focuses on vehicles, a typical target in large-scale video analytics. For analysis, we adopt Yolov7 series [44] for OD, BP and CT tasks, and PicoDet [33] for VD task. Moreover, RTMDet [28] is used for segmentation to assess semantic consistency of pseudo frames.

Baselines. We compare ParaDeco with four baselines. First, we consider modern VAPs without decoding optimization: (1) C-VAP uses CPU-based FFmpeg decoding followed by Yolov7 inference with TensorRT, while (2) N-VAP employs NVDEC [30], optimized for GPU, with the same inference backend. We also compare to two state-of-the-art VAPs, both with pre-decoding filtering to reduce decoding workload: (3) COVA+ [17] uses oracle object tracks to select minimal anchor frames for successive NVDEC decoding and inference, and (4) PacketGame+ [57] leverages ground-truth labels to guide packet filtering and decodes only the selected packets on the CPU. Note that both methods are enhanced with oracle filtering to isolate the impact of over-decoding, excluding filtering errors.

Table 4: Comparison of decoded frames (Sel: Selected to Decode, Over: Over-Decoding, Act: Actual Decoding).

System	Traffic			Downtown			Glenwood			Tilton			Wales		
	Sel.	Over	Act.	Sel.	Over	Act.	Sel.	Over	Act.	Sel.	Over	Act.	Sel.	Over	Act.
PacketGame+	0.735	0.126	0.861	0.734	0.148	0.882	0.646	0.121	0.767	0.759	0.149	0.908	0.351	0.169	0.520
COVA+	0.014	0.576	0.590	0.007	0.292	0.299	0.013	0.527	0.541	0.007	0.307	0.314	0.007	0.320	0.327
ParaDeco	0.064	0	**0.064**	0.078	0	**0.078**	0.077	0	**0.077**	0.064	0	**0.064**	0.075	0	**0.075**

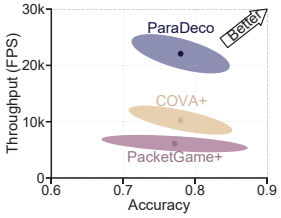


Figure 12: ParaDeco v.s. Decoding Accelerated VAPs.

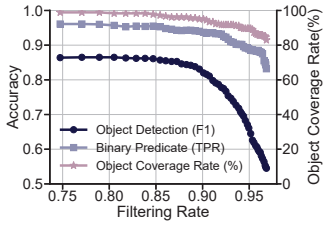


Figure 13: Performance of Contribution-based Filtering.

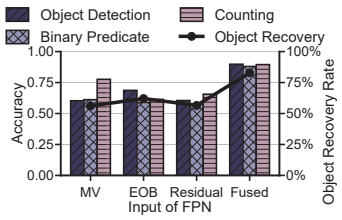


Figure 14: Comparison of analysis results with/without FFN.

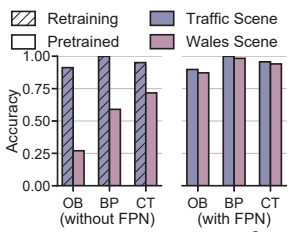


Figure 15: Comparison of analysis results with/without FPN.

4.2 Performance of ParaDeco

Throughput Improvement. Fig. 11 presents the throughput of ParaDeco and baseline VAPs across five datasets under the same accuracy level. ParaDeco achieves an average speedup of 4.41× and 5.68× over C-VAP and N-VAP, respectively, and outperforms decoding-optimized COVA+ and PacketGame+ by 1.59×–4.58×. This significant improvement highlights ParaDeco's ability to eliminate the over-decoding bottleneck by generating high-quality necessary pseudo frames in parallel. COVA+ still incurs substantial over-decoding overhead, as decoding anchor frames requires processing many redundant frames. PacketGame+ filters at the packet level, but decoding a selected packet still requires its preceding packets within the GOP. The performance gap is amplified in the Traffic scene, where high object occupancy demands more anchor frames and packets to maintain accuracy, further limiting throughput.

Accuracy Analysis. Since the baselines leverage strengthened filtering (with oracle knowledge), a direct accuracy comparison would be biased. Hence, we first present ParaDeco's accuracy and then evaluate the accuracy-throughput trade-offs with baselines. Table 4 reports the detailed accuracy results of ParaDeco on four tasks for all datasets, keeping at most 10% frames selected. For BP and CT tasks, ParaDeco achieves a high average accuracy exceeding 0.93, indicating that pseudo-frame detection and result completion are effective. For more complicated tasks such as OD and VD, metadata-based analytics incurs larger accuracy drops of 0.175–0.238. Empirically, we observe roughly half of the loss is from result completion, which is inevitable in filtering-based VAPs [1, 19, 24, 26] (accuracy of pseudo frame is depicted in § 4.4). Further, as suggested by prior work on large-scale video analytics [17, 34], a modest accuracy loss of about 0.1–0.2 is often acceptable, especially when considering specific application purposes. Figure 12 compares the accuracy-throughput trade-offs on the OD task. The ellipse centers indicate average performance, with ParaDeco achieving 3.62× and 2.15× higher throughput than PacketGame+ and COVA+, respectively, while maintaining comparable accuracy. Overall, ParaDeco offers the best balance between accuracy and throughput.

Over-decoding Analysis. To understand why ParaDeco outperforms other decoding-optimized VAPs with pre-decoding filters, we analyze the proportion of decoded frames. As illustrated in Table 4, ParaDeco achieves *zero over-decoding* by enabling independent and parallel reconstruction of selected frames. However, PacketGame+ requires decoding additional referenced packets to decode the selected one. Moreover, as it conducts coarse-grained packet-level filtering, it has to filter out fewer packets in high object occupancy scenes to maintain accuracy. Although COVA+ selects fewer than 2% of anchor frames by oracle tracks, it still over-decodes 30%-60% of frames due to the constraints of traditional serial decoding.

4.3 Component-wise Analysis

Frame Filter Mechanism. First, we evaluate the effectiveness of our frame filter at varying filtering rates, as illustrated in Fig. 13. The results show that ParaDeco's selection mechanism filters out 85% of frames while maintaining object detection and binary predicate accuracy with little degradation. The effectiveness is attributed to the high object coverage rate, defined by the proportion of objects covered by the selected frames. By using the maximum contribution of frames as the priority for selection, ParaDeco tends to select frames containing object changes, ensuring that new objects are captured by at least one selected frame.

Skeleton Feature Fusion (FFN). To measure the contribution of the skeleton feature fusion network, we compare the analysis results using fused frame skeletons and individual metadata as inputs for the frame predictor. Fig. 14 indicates that the pseudo-frames generated from fused frame skeletons achieve average accuracy improvements of 0.23, 0.26, and 0.28 across tasks compared to the three metadata types, respectively. Moreover, the fused skeleton contributes to a higher object recovery rate, defined as the proportion of successfully detected objects relative to the total number of objects within all pseudo frames (i.e., recall). This improvement is attributed to the FFN's ability to merge multi-level features and consolidate information from all objects, whereas individual metadata lacks sufficient features to predict all objects.

Frame Predictor (FPN). Fig. 15 further evaluates the necessity of the frame predictor (FPN) by comparing detection accuracy with and without it. A natural question is whether the fused skeleton alone can serve as a sufficient pseudo-frame. Hence, we retrain the detection model on fused skeletons in Traffic scene, as it is not directly compatible with skeleton features. While high accuracy is

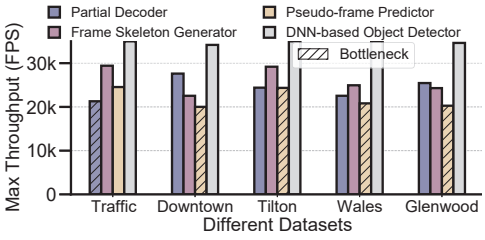


Figure 16: Throughput of each component in ParaDeco.

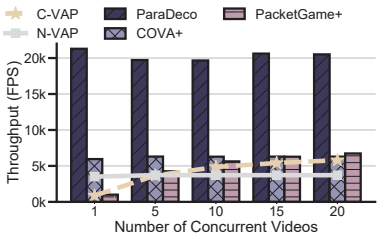


Figure 17: Throughput in different video concurrency.

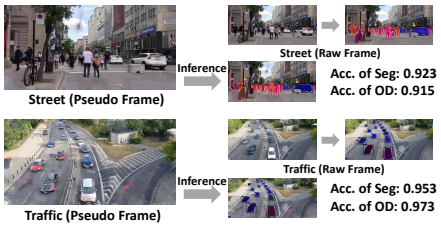


Figure 18: A visualization example of pseudo frames and inference results.

achieved in the Traffic scene, performance drops significantly when transferred to the Wales scene, which features a lower viewing angle. In contrast, pseudo-frames generated by the FPN maintain high accuracy across both scenes without retraining, benefiting from machine-centric prediction and the ability to capture long-term content similarity via scene-specific training. These results confirm that the FPN is essential for generalizable pseudo-frame analysis, as retraining inference models for each scene is impractical.

Throughput of Component. We analyze the maximal throughput of each component across different datasets, as shown in Fig. 16. Specifically, the maximal throughput is the component throughput (i.e., frames processed per second), divided by the selected frame proportion. Benefiting from the acceleration techniques and small input size, the DNN-based object detector never becomes a bottleneck for all datasets. And since the networks in pseudo-frame predictor are more computing-intensive, although with a throughput of more than 20,000 FPS, it still bounds the system throughput in most cases. The observation reveals the optimization potential of the system by improving the efficiency of pseudo-frame generation.

4.4 Compatibility Study

Concurrent Videos. Fig. 17 shows the throughput of ParaDeco and baselines under different video processing concurrency. Due to multi-core, C-VAP scales with available compute, whereas N-VAP saturates early due to full decoder utilization. ParaDeco's throughput, bound by pseudo-frame generation, remains unaffected by concurrency levels. However, PacketGame+ and CoVA+ exhibit limited throughput due to over-decoding, especially under low concurrency. When processing a single video, ParaDeco achieves 3.57× and 24.6× speedup compared to CoVA+ and PacketGame+.

Semantic Consistency. Since downstream tasks in video analytics basically depend on semantic and categorical information, it is essential that pseudo frames preserve these key semantics to support diverse applications. We assess semantic consistency through pixel-level accuracy on a segmentation task [31] and category-level consistency on an object detection task. Fig. 18 visualizes segmentation results across two representative scenes in large-scale video analytics [51, 54], street and traffic, showing that pseudo frames effectively retain critical semantic information while discarding some fine-grained texture and color details. As a result, it achieves a relative accuracy of over 0.9 compared to raw frames (normalized to 1) on both tasks. Specifically, ParaDeco reconstructs object categories and contours (e.g., people and vehicles) while omitting fine-grained textures such as facial features or license plates.

Discussion. Under extreme cases like very low bitrate or poor visibility, pseudo-frame quality may degrade due to sparse or noisy metadata. Though rare in real-world deployments where moderate compression and structured scenes are common, such cases remain an open challenge for both pixel- and metadata-based analysis.

5 Related Work

Video Decoding Acceleration. Efforts to accelerate decoding span both hardware and software. NVDEC [30] leverages NVIDIA GPUs for fast decoding via the graphics engine, while Kiloview [22] offers dedicated decoders (e.g., D350, D260) for up to 16 streams. However, scaling to thousands of streams remains costly, and modern codecs still suffer from inefficient over-decoding even with hardware support. On the software side, LiFteR [8] boosts frame rates via loose referencing but relies on offline re-encoding, complicating deployment. Selective decoding methods [10, 17, 45, 57] reduce overhead but still over-decode redundant frames, offering limited gains. In contrast, ParaDeco enables frame-level, generative decoding that restores only selected frames in parallel, maximizing throughput.

Filtering-Based Video Analytics. Frame filtering improves video analytics efficiency by removing redundancy and can be categorized as post-decoding or pre-decoding. Post-decoding methods like FFS-VA [59] use multi-stage filtering and global feedback to accelerate processing, while InFi [56] employs neural networks to skip or reuse frames based on prior results. Pre-decoding approaches such as PacketGame [57] apply packet gating to discard less informative data before decoding but risk missing critical frames and cause over-decoding. Camera-side solutions like Reducto [24] detect content changes to eliminate redundancy but are impractical due to limited programmability and incompatibility with pre-encoded videos. Unlike these, ParaDeco decouples inter-frame dependencies and adopts a "decode-what-matters" strategy, decoding and analyzing only selected frames to maximize cascading benefits.

6 Conclusion

In this paper, we identified the over-decoding trap as a major bottleneck in VAPs, as codec dependencies force the decoding of many redundant frames. To address the issue, we presented ParaDeco, a video analytics framework that employs a decode-what-matters approach for high-throughput, large-scale analytics. Unlike existing VAPs that rely on the traditional decoder, ParaDeco performs frame-level parallel generative decoding only on selected frames, preventing the over-decoding of redundant content. Thus, through decoding and analyzing a few critical frames, it delivers extremely fast and semantically accurate analytics of video data. Experiments on real-world video datasets demonstrate that ParaDeco achieves up to 4.58× speedup over state-of-the-art decoding-optimized VAPs.

Acknowledgments

This work was supported in part by Nanjing University-China Mobile Communications Group Co.,Ltd. Joint Institute, Nanjing Key S&T Special Projects (202309006), NSFC (62202233), and Grant from State Key Laboratory for Novel Software Technology, Nanjing University (KFKT2024B18).

References

[1] Neil Agarwal and Ravi Netravali. 2023. Boggart: Towards General-Purpose acceleration of retrospective video analytics. In *Proceedings of USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. 933–951.

[2] Michael R Anderson, Michael Cafarella, German Ros, and Thomas F Wenisch. 2019. Physical representation-based predicate optimization for a visual analytics database. In *Proceedings of IEEE International Conference on Data Engineering (ICDE)*. 1466–1477.

[3] AV1 Video Codec. 2025. https://aomedia.org/specifications/av1/.

[4] R Venkatesh Babu, Manu Tom, and Paras Wadekar. 2016. A survey on compressed domain video analysis techniques. *Multimedia Tools and Applications* 75 (2016), 1043–1078.

[5] A Balasundaram and C Chellappan. 2020. An intelligent video analytics model for abnormal event detection in online surveillance video. *Journal of Real-Time Image Processing* 17, 4 (2020), 915–930.

[6] Bilel Benjdira, Anis Koubaa, Ahmad Taher Azar, Zahid Khan, Adel Ammar, and Wadii Boulila. 2022. TAU: A framework for video-based traffic analytics leveraging artificial intelligence and unmanned aerial systems. *Engineering Applications of Artificial Intelligence* 114 (2022), 105095.

[7] Christopher Canel, Thomas Kim, Giulio Zhou, Conglong Li, Hyeontaek Lim, David G Andersen, Michael Kaminsky, and Subramanya Dulloor. 2019. Scaling video analytics on constrained edge nodes. In *Proceedings of the Second Conference on Machine Learning and Systems (MLSys)*. 406–417.

[8] Bo Chen, Zhisheng Yan, Yinjie Zhang, Zhe Yang, and Klara Nahrstedt. 2024. LiFteR: Unleash Learned Codecs in Video Streaming with Loose Frame Referencing. In *Proceedings of USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. 533–548.

[9] Tiffany Yu-Han Chen, Lenin Ravindranath, Shuo Deng, Paramvir Bahl, and Hari Balakrishnan. 2015. Glimpse: Continuous, real-time object recognition on mobile devices. In *Proceedings of ACM International Conference on Embedded Networked Sensor Systems (SenSys)*. 155–168.

[10] Xiang Fang, Daizong Liu, Pan Zhou, and Guoshun Nan. 2023. You can ground earlier than see: An effective and efficient pipeline for temporal sentence grounding in compressed videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2448–2460.

[11] FFmpeg. 2025. https://ffmpeg.org/.

[12] Google's libvpx Official Github Repository. 2025. https://github.com/webmproject/libvpx/.

[13] H.265 Specification. 2025. https://www.itu.int/rec/T-REC-H.265.

[14] Mengxi Hanyao, Yibo Jin, Zhuzhong Qian, Sheng Zhang, and Sanglu Lu. 2021. Edge-assisted online on-device object detection for real-time video analytics. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*. 1–10.

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

[16] Yihui He, Xiangyu Zhang, and Jian Sun. 2017. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision (ICCV)*. 1389–1397.

[17] Jinwoo Hwang, Minsu Kim, Daeun Kim, Seungho Nam, Yoonsung Kim, Dohee Kim, Hardik Sharma, and Jongse Park. 2022. CoVA: Exploiting Compressed-Domain analysis to accelerate video analytics. In *Proceedings of USENIX Annual Technical Conference (ATC)*. 707–722.

[18] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. 1125–1134.

[19] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. 2018. Chameleon: scalable adaptation of video analytics. In *Proceedings of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*. 253–266.

[20] Daniel Kang, Peter Bailis, and Matei Zaharia. 2018. Blazeit: Optimizing declarative aggregation and limit queries for neural network-based video analytics. *arXiv preprint arXiv:1805.01046* (2018).

[21] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529* (2017).

[22] Kiloview. 2025. Kiloview D350/D260 Multi-Channel Decoder. https://www.kiloview.com/en/decoder/.

[23] Yuxin Kong, Peng Yang, and Yan Cheng. 2023. Edge-assisted on-device model update for video analytics in adverse environments. In *Proceedings of the 31st ACM International Conference on Multimedia (MM)*. 9051–9060.

[24] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. 2020. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *Proceedings of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*. 359–376.

[25] Fengyin Lin, Mingkang Li, Da Li, Timothy Hospedales, Yi-Zhe Song, and Yonggang Qi. 2023. Zero-shot everything sketch-based image retrieval, and in explainable style. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 23349–23358.

[26] Luyang Liu, Hongyu Li, and Marco Gruteser. 2019. Edge assisted real-time object detection for mobile augmented reality. In *Proceedings of ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*. 1–16.

[27] Jian-Hao Luo and Jianxin Wu. 2020. Neural network pruning with residual-connections and limited-data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition(CVPR)*. 1458–1467.

[28] Chengqi Lyu, Wenwei Zhang, Haian Huang, Yue Zhou, Yudong Wang, Yanyi Liu, Shilong Zhang, and Kai Chen. 2022. Rtmdet: An empirical study of designing real-time object detectors. *arXiv preprint arXiv:2212.07784* (2022).

[29] Anup Mohan, Kent Gauen, Yung-Hsiang Lu, Wei Wayne Li, and Xuemin Chen. 2017. Internet of video things in 2030: A world with many cameras. In *Proceedings of IEEE international symposium on circuits and systems (ISCAS)*. 1–4.

[30] NVIDIA. 2025. NVIDIA Video Codec SDK. https://developer.nvidia.com/video-codec-sdk.

[31] OpenMMLab. 2025. MMSegmentation. https://github.com/open-mmlab/mmsegmentation.

[32] OpenVINO. 2025. https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/overview.html.

[33] Paddle. 2025. PaddleDetection. https://github.com/PaddlePaddle.

[34] Jongse Park, Emmanuel Amaro, Divya Mahajan, Bradley Thwaites, and Hadi Esmaeilzadeh. 2016. AxGames: Towards crowdsourcing quality target determination in approximate computing. In *Proceedings of ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*. 623–636.

[35] Pytorch. 2025. https://pytorch.org/.

[36] Aluizio Rocha Neto, Thiago P Silva, Thais Batista, Flávia C Delicato, Paulo F Pires, and Frederico Lopes. 2020. Leveraging edge intelligence for video analytics in smart city applications. *Information* 12, 1 (2020), 14.

[37] Weibin Rong, Zhanjing Li, Wei Zhang, and Lining Sun. 2014. An improved CANNY edge detection algorithm. In *Proceedings of IEEE international conference on mechatronics and automation (ICMA)*. 577–582.

[38] Aneeshan Sain, Ayan Kumar Bhunia, Pinaki Nath Chowdhury, Subhadeep Koley, Tao Xiang, and Yi-Zhe Song. 2023. Clip for all things zero-shot sketch-based image retrieval, fine-grained or not. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2765–2775.

[39] Haichen Shen, Seungyeop Han, Matthai Philipose, and Arvind Krishnamurthy. 2017. Fast video classification via adaptive cascading of deep models. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. 3646–3654.

[40] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[41] Lin Sun, Weijun Wang, Tingting Yuan, Liang Mi, Haipeng Dai, Yunxin Liu, and Xiaoming Fu. 2024. BiSwift: Bandwidth orchestrator for multi-stream video analytics on edge. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*. 1181–1190.

[42] TensorRT. 2025. https://developer.nvidia.com/tensorrt.

[43] TVM. 2025. https://tvm.apache.org/.

[44] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. 2023. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. 7464–7475.

[45] Yongchen Wang, Ying Wang, Huawei Li, and Xiaowei Li. 2021. An efficient deep learning accelerator architecture for compressed video analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 9 (2021), 2808–2820.

[46] Webm Official Website. 2025. https://www.webmproject.org/.

[47] Kai Xu and Angela Yao. 2022. Accelerating video object segmentation with compressed video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1342–1351.

[48] Zheng Yang, Xu Wang, Jiahang Wu, Yi Zhao, Qiang Ma, Xin Miao, Li Zhang, and Zimu Zhou. 2022. Edgeduet: Tiling small object detection for edge assisted autonomous mobile vision. *IEEE/ACM Transactions on Networking (TON)* (2022).

[49] Hyunho Yeo, Youngmok Jung, Jaehong Kim, Jinwoo Shin, and Dongsu Han. 2018. Neural adaptive content-aware internet video delivery. In *Proceedings of USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 645–661.

[50] YouTube. 2025. 4 Corners Camera Downtown. https://www.youtube.com/watch?v=ByED80IKdIU.
[51] YouTube. 2025. 4K Road traffic video for object detection and tracking. https://www.youtube.com/watch?v=QuUxHIVUoaY.
[52] YouTube. 2025. Ammanford Cam Wales. https://www.youtube.com/watch?v=Uczhfukba_k.
[53] YouTube. 2025. Grand Avenue Bridge in Glenwood Springs Live Camera. https://www.youtube.com/watch?v=B0YjuKbVZ5w.
[54] YouTube. 2025. People Walking Stock Footage. https://www.youtube.com/watch?v=bwJ-TNu0hGM.
[55] YouTube. 2025. Village of Tilton. https://www.youtube.com/watch?v=5_XSYlAfJZM.
[56] Mu Yuan, Lan Zhang, Fengxiang He, Xueting Tong, and Xiang-Yang Li. 2022. Infi: End-to-end learnable input filter for resource-efficient mobile-centric inference. In *Proceedings of ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*. 228–241.
[57] Mu Yuan, Lan Zhang, Xuanke You, and Xiang-Yang Li. 2023. PacketGame: Multi-Stream Packet Gating for Concurrent Video Inference at Scale. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*. 724–737.
[58] Donghai Zhai, Xiaobo Zhang, Xun Li, Xichen Xing, Yuxin Zhou, and Changyou Ma. 2023. Object detection methods on compressed domain videos: An overview, comparative analysis, and new directions. *Measurement* 207 (2023), 112371.
[59] Chen Zhang, Qiang Cao, Hong Jiang, Wenhui Zhang, Jingjun Li, and Jie Yao. 2020. A fast filtering mechanism to improve efficiency of large-scale video analytics. *IEEE Transactions on Computers (TC)* 69, 6 (2020), 914–928.
[60] Wuyang Zhang, Zhezhi He, Luyang Liu, Zhenhua Jia, Yunxin Liu, Marco Gruteser, Dipankar Raychaudhuri, and Yanyong Zhang. 2021. Elf: accelerate high-resolution mobile deep vision with content-aware parallel offloading. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 201–214.
[61] Yu Zhang, Yu Liu, Peng Sun, Han Yan, Xiaolin Zhao, and Li Zhang. 2020. IFCNN: A general image fusion framework based on convolutional neural network. *Information Fusion* 54 (2020), 99–118.
[62] Jia-Xing Zhao, Jiang-Jiang Liu, Deng-Ping Fan, Yang Cao, Jufeng Yang, and Ming-Ming Cheng. 2019. EGNet: Edge guidance network for salient object detection. In *Proceedings of the IEEE/CVF international conference on computer vision (CVPR)*. 8779–8788.