Mathematical Modeling and Methodology

Section 1: Mathematical Modeling of the Problem

1.1 Problem Definition

Sequence of Frames

Let $\{I_t\}_{t=1}^{T}$ denote a sequence of image frames captured over time by a vision-enabled device, where $I_t \in \mathbb{R}^{H\times W\times C}$ with H and W representing the height and width in pixels, and C the number of color channels (e.g., $C = 3$ for RGB). Each frame is independently subject to higher-level analysis (scene classification or object detection). Since consecutive frames are often highly correlated, processing all frames with the heavy model is inefficient [1].

Defining heavy model:

Let $f_\theta : \mathbb{R}^{H\times W\times C} \to Y$

denote a high-capacity object detection or scene classification model, parameterized by θ, which maps input frame I_t to output space Y (e.g., class labels or bounding boxes). The heavy model has significant computational cost C_f [2][3][4].

The predicted output is

$$\hat{y}_t = f_\theta(I_t), \hat{y}_t \in Y \qquad (1)$$

Defining gatekeeper:

To reduce computational cost, we define a gatekeeper function:

$$g_\phi : \mathbb{R}^{H\times W\times C} \times \mathbb{R}^{H\times W\times C} \to \{0,1\}$$

where $g_\phi(I_t, I_{t-1}) = 1$ indicates that f_θ should process frame I_t, and 0 indicates it can be skipped or processed with a lightweight fallback $L(I_t)$ [5][6].

The gatekeeper decision is:

$$d_t = g_\phi(I_t, I_{t-1}), d_t \in \{0,1\} \qquad (2)$$

If $d_t = 1$, the heavy model executes:

$$\hat{y}_t = f_\theta(I_t) \qquad (3)$$

Otherwise:

$$\hat{y}_t = L(I_t) \qquad (4)$$

Costs:

We define the computational costs per frame:

- C_f: heavy model
- C_g: gatekeeper function ($C_g \ll C_f$)
- C_l: fallback or skip operation ($C_l \ll C_g$)

The expected computational cost per frame is:

$$C_{\text{avg}} = \mathbb{E}[C_g + d_t C_f + (1 - d_t) C_l] \qquad (5)$$

Loss function:

The task-specific loss measures discrepancy between predicted and ground truth outputs:

$$\ell(f_\theta(I_t), y_t) \qquad (6)$$

Depending on the task:

- Classification: cross-entropy

$$\ell_{\text{CE}}(f_\theta(I_t), y_t) = -\sum_{i=1}^{n} \mathbf{1}_{[y_t = c_i]} \log P(c_i \mid f_\theta(I_t))$$

- Regression: L2 loss

$$\ell_{\text{L2}}(f_\theta(I_t), y_t) = \| f_\theta(I_t) - y_t \|_2^2$$

Objective: Minimize expected cost while maintaining task accuracy

The system aims to minimize expected cost while maintaining task accuracy:

$$\min_{\phi} \ \mathbb{E}[C_g + d_t C_f + (1 - d_t) C_l] \text{ s.t. } \mathbb{E}[\ell(f_\theta(I_t), y_t)] \leq \epsilon \qquad (7)$$

Where:

- Expectation is over the frame distribution $I_t \in D$
- ϵ is a tolerable loss threshold
- ϕ are the gatekeeper parameters

In essence, the gatekeeper selectively forwards frames to the heavy model, reducing expected cost while ensuring accuracy [2][3][5][6].

1.2 Objective Function

Minimize expected computation while bounding loss:

Let the gate decision be $d_t = g_\phi(I_t, I_{t-1})$. The expected computational cost per frame is

$$C_{\text{avg}} = \mathbb{E}[\, C_g + d_t C_f + (1 - d_t) C_l \,]. \qquad (8)$$

We denote the expected task loss per frame by

$$L_{\text{avg}} = \mathbb{E}[\, \ell(f_\theta(I_t), y_t) \,]. \qquad (9)$$

The constrained optimization problem is therefore written as

$$\boxed{\min_{\phi} \; C_{\text{avg}} \text{s.t.} L_{\text{avg}} \leq \varepsilon} \qquad (10)$$

where $\varepsilon$is a user-specified tolerance on expected loss (or equivalently a minimum acceptable accuracy).

Equivalently, one can form a Lagrangian (cost–accuracy trade-off) and optimize a single scalar objective:

$$\boxed{\min_{\phi} \; \mathcal{J}(\phi) = \mathbb{E}[\, \ell_{\text{used}}(I_t) + \lambda(C_g + d_t C_f + (1 - d_t) C_l)]} \qquad (11)$$

where $\ell_{\text{used}}(I_t) = d_t \ell(f_\theta(I_t), y_t) + (1 - d_t)\ell(L(I_t), y_t)$, and $\lambda \geq 0$controls the trade-off between accuracy and cost. Choosing λcorresponds to moving along the Pareto front between lowest loss and lowest cost.

Useful derived expressions. Let $p = \Pr(d_t = 1) = \mathbb{E}[d_t]$be the fraction of frames forwarded to the heavy model. Then the cost in (8) can be written in terms of $p$as

$$C_{\text{avg}} = C_g + p\, C_f + (1 - p)\, C_l. \qquad (12)$$

The relative compute saving versus always-running the heavy model (baseline cost C_f) is

$$\text{Saved\_frac} = 1 - \frac{C_{\text{avg}}}{C_f} = 1 - \frac{C_g}{C_f} - p - (1 - p)\frac{C_l}{C_f}. \qquad (13)$$

1.3 Assumptions

To ensure the mathematical formulation remains tractable and analytically interpretable, several simplifying assumptions are made in this study:

1. **Temporal Correlation of Frames.**
   Consecutive video frames $I_t$and $I_{t+1}$are assumed to be strongly correlated in both spatial and semantic content. This allows the gatekeeper $g_\phi$to estimate frame novelty or change using low-rank features such as the singular values of the SVD decomposition.
   This assumption is widely adopted in video-efficiency research [1], [2].

2. **Hierarchical Computational Costs.**
   The processing cost of the gatekeeper network $C_g$is significantly smaller than that of the heavy object-detection model C_f, i.e.

$$C_l \ll C_g \ll C_f. \qquad (12)$$

This hierarchy guarantees that skipping redundant frames yields a measurable gain in energy and latency. Similar cost modeling has been adopted in adaptive inference studies [5], [6].

3. **Decision Dependence on Frame Variation.**
   The gatekeeper decision d_t is assumed to depend monotonically on frame difference or novelty metrics (e.g., the temporal correlation coefficient $\rho_k(t)$).
   Formally,

$$\text{Higher } D_t \text{ or lower } \rho_k(t) \Rightarrow \text{higher probability of running } f_\theta. \quad (13)$$

This simplifies the decision boundary and enables threshold-based gating.

4. **Accuracy Constraint.**
   The overall system must maintain an average loss L_{avg} that does not exceed an acceptable upper bound L_{max}, defined during validation.
   This ensures that computational savings do not compromise detection quality.

Section 2: Mathematical methodology

2.1 Overview of Proposed Approach

The proposed system aims to dynamically decide whether a given frame in a video sequence should be processed by a heavy model or skipped, based on the structural change between consecutive frames. The framework operates on a sequence of frames $\{I_t\}$, each representing an input image at time t.

2.1.1 Feature Extraction and Temporal Metrics

Each frame I_t is first transformed into a low-rank feature matrix $M_t \in \mathbb{R}^{H\times W\times C}$, which captures its essential spatial and channel-wise information. To efficiently summarize the internal structure of the frame, we compute its Singular Value Decomposition (SVD):

$$M_t \rightarrow U_t \Sigma_t V_t^T \quad (14)$$

where U_t and V_t represent the left and right singular vectors, and Σ_t is the diagonal matrix of singular values arranged in descending order.
This step yields a compact, low-rank representation that highlights the dominant structural and energy components in the frame.

To quantify the similarity between consecutive frames, we define the temporal correlation metric $\rho_k(t)$ using the top-k singular values from two consecutive frames:

$$\rho_k(t) = \frac{\Sigma_t^{(1:k)} \cdot \Sigma_{t-1}^{(1:k)}}{\| \Sigma_t^{(1:k)} \| \ \| \Sigma_{t-1}^{(1:k)} \|} \quad (15)$$

Here, $\Sigma_t^{(1:k)} = [\sigma_{t,1}, \dots, \sigma_{t,k}]$ denotes the first k singular values of frame t.
The metric $\rho_k(t)$ measures the cosine similarity between the singular spectra of successive frames:

- $\rho_k(t) \approx 1$: frames are highly correlated, indicating minimal structural change.

- $\rho_k(t)$ decreases: suggests motion, scene change, or novel content.

Additionally, a frame-difference measure D_t can be computed to capture pixel- or feature-level changes between frames, complementing the SVD-based similarity.

2.1.2 Gatekeeper Decision Mechanism

The computed metrics are then passed to a gatekeeper network g_ϕ, which determines whether to execute the heavy inference model or use a lightweight alternative. The gate function outputs a binary decision:

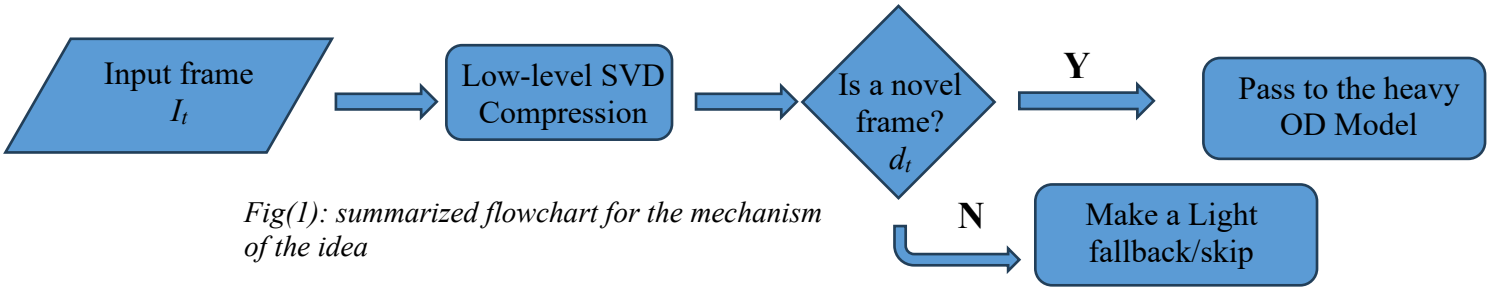
$$d_t = g_\phi(s_t), d_t \in \{0,1\} \qquad (16)$$

where $s_t = [\rho_k(t), D_t,$other temporal features$]$ represents the combined state vector describing the current frame's novelty level.

The decision logic is as follows:

- If $d_t = 1$: the heavy model f_θ is executed to ensure high accuracy on new or complex content.
- If $d_t = 0$: the system skips or falls back to a lightweight approximation (e.g., cached output, optical flow prediction, or previous inference result).

This adaptive mechanism allows the system to maintain accuracy with reduced computational cost, balancing efficiency and responsiveness to scene dynamics.



Fig(1): summarized flowchart for the mechanism of the idea

2.2 Gatekeeper Design

"The gatekeeper is rule-based: it triggers the heavy model only if the frame-difference metric D_t exceeds a threshold τ."

In the proposed framework, the *gatekeeper module* acts as the decision controller that determines whether each input frame should be processed by the heavy model or skipped. Its primary goal is to minimize computational cost while ensuring that the prediction accuracy remains within acceptable limits.

The gatekeeper design in this methodology directly operationalizes the optimization objective introduced in Section 1.2. Its goal is to minimize the expected computational cost while maintaining an acceptable level of task accuracy, formulated as $\phi_{\min} E\left[l_u sed(I_t) + \lambda\left(C_g + d_t C_f + (1 - d_t)C_l\right)\right]$

Here, ℓ_{used} (I_t) denotes the task-specific loss associated with the processed frame I_t; C_f, C_l, and C_g represent the computational costs of the heavy model, the lightweight fallback, and the gatekeeper function, respectively.

The binary decision variable d_t determines whether the heavy model is executed at time t. This formulation aligns with recent adaptive-inference frameworks that dynamically balance computational efficiency and predictive accuracy at the input level [5], [6].

2.3 Gatekeeper Decision Function (Rule-Based Mechanism)

In this stage, the gatekeeper implements a rule-based decision process that determines whether to activate the heavy model or skip processing for a given frame. The decision is based on the difference between two consecutive frames, which reflects the level of visual change or motion information. This difference metric is defined as

(13) $D_t = \|\Sigma_t - \Sigma_{t-1}\|_2$

where Σ_t represents the singular-value vector of frame I_t obtained through Singular Value Decomposition (SVD). The magnitude of D_t captures the variation between consecutive frames.

Based on this measure, the gatekeeper makes a binary decision according to a predefined threshold τ.

(14) $d_t = \{if\ D_t > \tau\ \ ,1\ otherwise\ ,0\}$

If D_t exceeds the threshold, the gatekeeper triggers the heavy model; otherwise, the lightweight module or skipping strategy is applied. The threshold τ is tuned empirically on a validation subset to balance the trade-off between computational cost and inference accuracy.

This rule-based formulation provides a lightweight and interpretable gating mechanism that requires no additional training and can be easily adapted to different datasets or models. Similar threshold-based gating approaches have recently been reported in adaptive deep learning frameworks for efficient vision inference [5], [6].

Cost-aware loss(based only on rule-based keeper) and happen just after inference

To further quantify the balance between accuracy and computational efficiency, a cost-aware metric is adopted, defined as

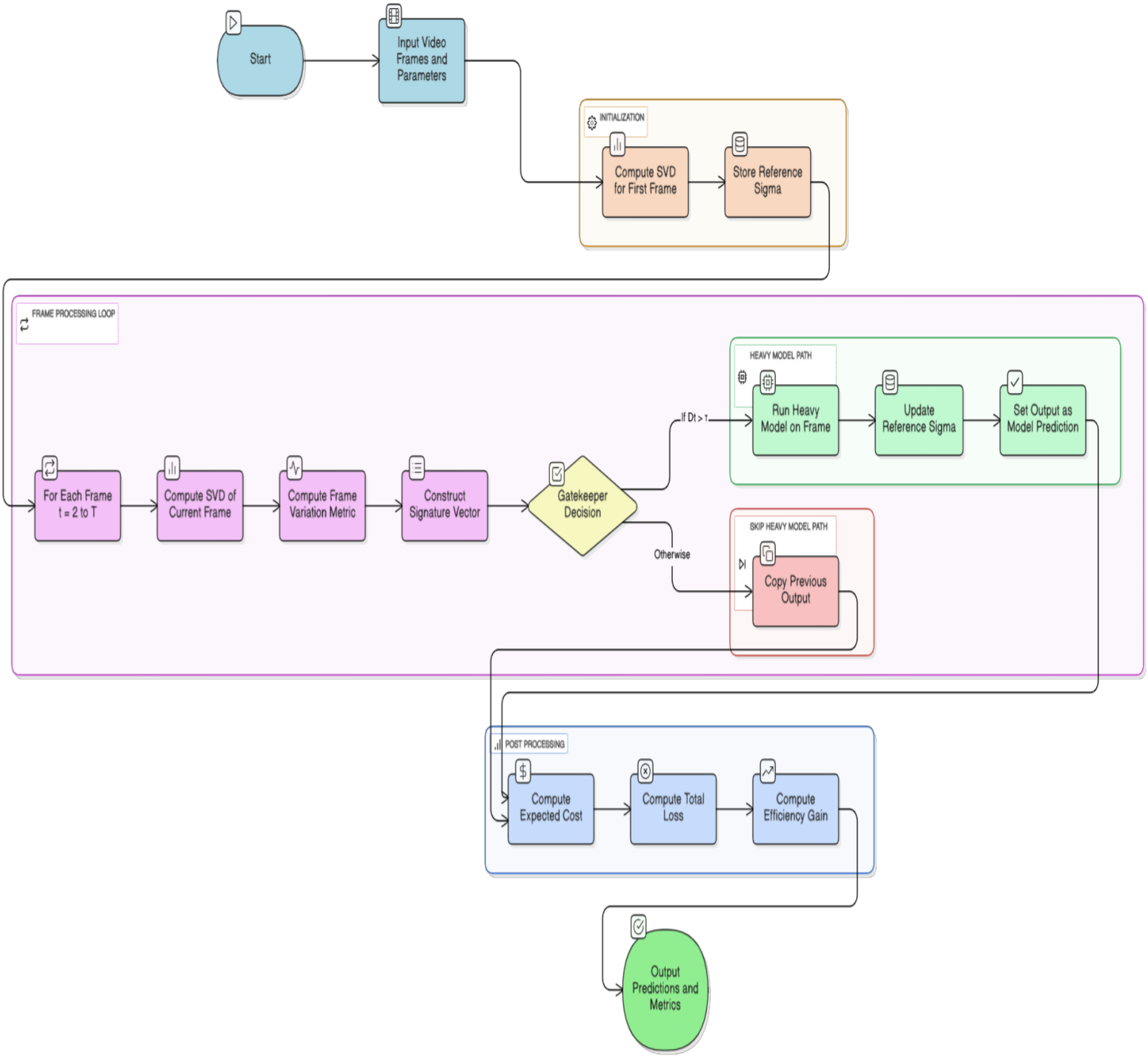
$$(15) L_{\text{total}} = L_{(accuracy)} + \beta \times C_a vg$$

L_{accuracy} denotes the task-specific performance loss (for example, classification or detection error), and $C_a vg$ represents the average computational cost per input frame. The weighting factor β controls the relative importance of computational efficiency versus accuracy. In this work, this formulation is used not as a training objective, but rather as an evaluation indicator to assess how well the rule-based gatekeeper has achieved the trade-off between speed and performance.

This type of post-inference cost analysis has been highlighted in recent research on efficient deep inference and conditional computation for edge-based systems [5], [6].

2.3 Algorithm / Pseudocode

As shown in the *fig(2)* the algorithm is about taking the input frames, compressing them using SVD and then according to the GateKeeper, where its mechanism is explained in *section 2.2* , output it decides whether the frame will pass to the heavy AI object detection model or not.



Fig(2): a detailed flowchart showing the idea mechanism

Citation

[1] ScienceDirect, “Video Sequences,” *Computer Science Topics*,[Online]. [Accessed: Nov. 11, 2025]. Available:https://www.sciencedirect.com/topics/computer-science/video-sequences.

[2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 2017. Available: https://doi.org/10.1109/TPAMI.2016.2577031

[3] J. Glenn Jocher et al., “YOLOv5: A Real-Time Object Detection Framework,” arXiv preprint arXiv:2006.02675, 2020. Available: https://arxiv.org/abs/2006.02675

[4] A. Dosovitskiy et al., “An Image Is Worth 16×16 Words: Transformers for Image Recognition at Scale,” arXiv preprint arXiv:2010.11929, 2020. Available: https://arxiv.org/abs/2010.11929

[5] S. Teerapittayanon, B. McDanel, and H. T. Kung, “BranchyNet: Fast Inference Via Early Exiting from Deep Neural Networks,” in 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 2464-2469, 2016. Available: https://doi.org/10.1109/ICPR.2016.7890059

[6] M. Bolukbasi, J. Wang, D. Dekel, and V. Saligrama, “Adaptive Neural Networks for Efficient Inference,” arXiv preprint arXiv:1702.07811, 2017. Available: https://arxiv.org/abs/1702.07811