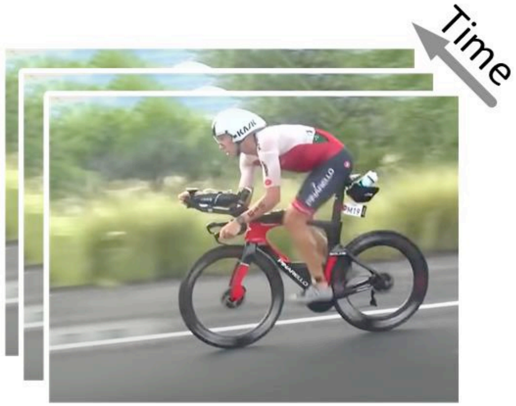
Palleon: A Runtime System for Efficient Video Processing toward Dynamic Class Skew

Boyuan Feng, Yuke Wang, Gushu Li, Yuan Xie, and Yufei Ding

UC SANTA BARBARA

Motivation



Continuous Video Streams

Mobile Platforms

CNN-based video processing system
- Analyze continuous video streams
- High accuracy
- Intensive resource consumption (e.g., energy and latency)

Mobile platforms
- Limited resource (e.g., energy and latency)

Goal: video processing system on mobile platforms with high accuracy, low latency, and low energy consumption.

Temporal Locality: Class Skew

- A small number of classes keep appearing in a large number of consecutive frames

Class Skew: Class Cardinality

- A small number of classes keep appearing in a large number of consecutive frames

Class cardinality

- The number of classes in a class skew
- Intuitive benefit:
  - From a general CNN (for recognizing thousands of classes)
  - To a specialized CNN (only recognizing a small number of classes in the current class skew)

Class Skew: Visual Separability

- A small number of classes keep appearing in a large number of consecutive frames

Easier to classify

Harder to classify

Visual separability

- Under the same class cardinality, one class skew is easy to classify and the other one is hard
- Intuitive benefit
  - Use a more compact model when the class skew is easier to classify

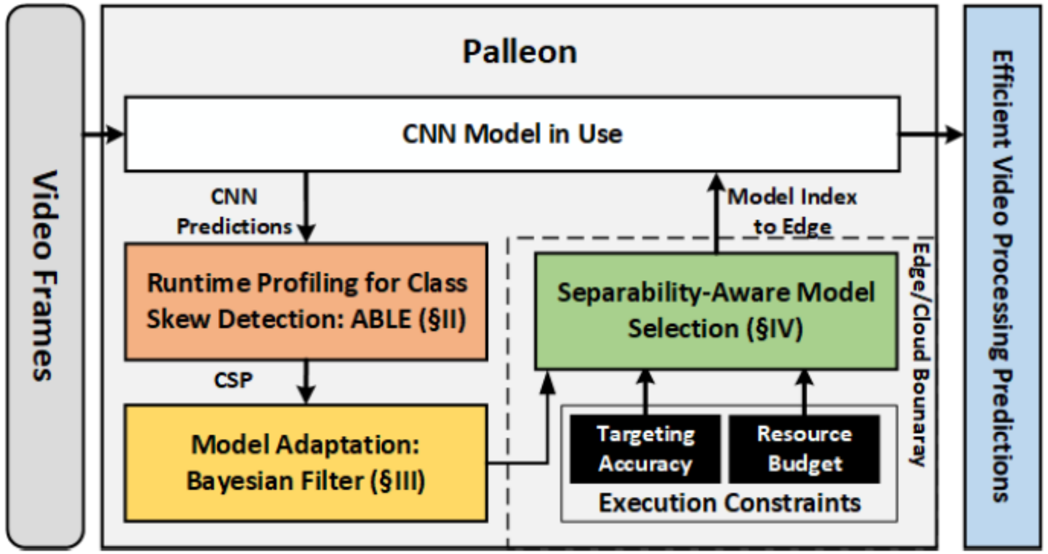
Challenges

- **How to precisely capture class skews?**
  - New class skews appear and disappear suddenly as time goes
  - A class skew may last for minutes or even hours while this lasting time varies across videos and scenarios

- **How to efficiently adapt CNNs during runtime?**
  - Do not foreknow class skews in an online video
  - Hard to offline train compact models for each class skew
  - Existing model adaptation are computation-intensive and not affordable on mobile platforms

- **How to exploit visual separability and efficiently select CNNs?**
  - A single model adapted to different class skews show significantly different accuracy
  - Model selection on mobile platforms may introduce high overhead

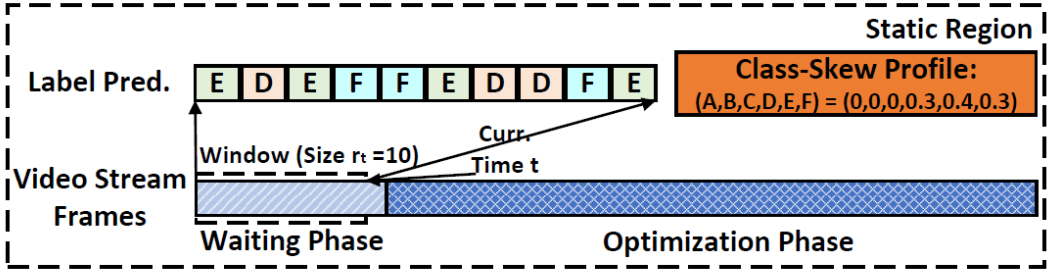
System Overview



- **ABLE** to abstract class skews from video streams
- **Bayesian Filter** to adapt CNNs toward the detected class skew during runtime
- **Separability-Aware Model Selection** to further squeeze system energy consumption

ABLE

Goal:
- Give a precise class-skew profile (CSP) in static regions between adjacent class-skew switches
- Detect when the class-skew switches occur



Static Class-Skew Profiling
- Approximate the CSP in each static region with an empirical distribution

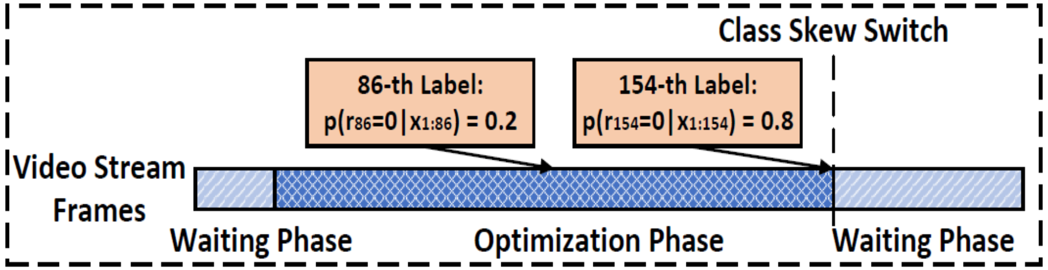
$$p(j|r_t, x_{1:t}) = \frac{1}{r_t} \sum_{i=t-r_t+1}^{t} \mathbb{1}_{x_i=j}$$

- Early optimization by adaptive waiting scheme based on asymptotic error bound

$$F_{min} = \max_{\hat{p}_j > \xi} Z_c / (\varepsilon \sqrt{\hat{p}(j|r_t, x_{1:t})})$$

ABLE

Goal:
- Give a precise class-skew profile (CSP) in static regions between adjacent class-skew switches
- Detect when the class-skew switches occur



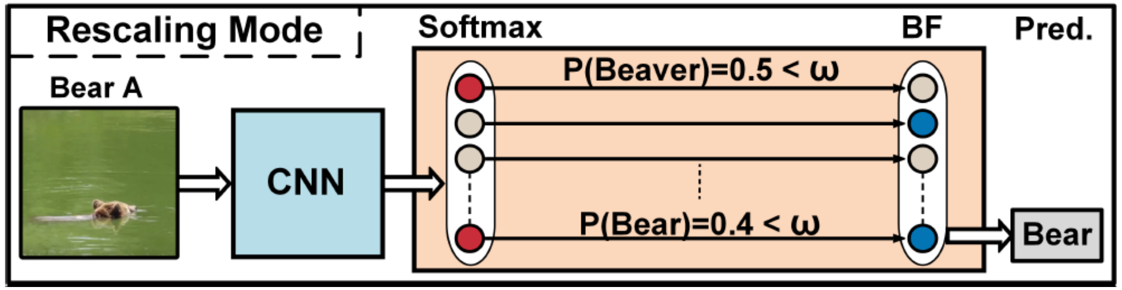
$$p(r_t|x_{1:t}) = p(r_t, x_{1:t}) / \sum_{r_t=0}^{t} p(r_t, x_{1:t})$$

$$p(r_t, x_{1:t}) = \sum_{i=1}^{\hat{n}} p(r_t|r_{t-1}=w_i) \cdot p(x_t|r_{t-1}=w_i, x_{1:t-1}) \cdot p(r_{t-1}=w_i, x_{1:t-1})$$

Dynamic Class-Skew Switch Detection
- Estimate the probability of class-skew switch at each time step t by considering:
  - Joint probability of the lasting time and the predicted label
  - The probability that a class skew of length r_{t-1} is still alive at r_t
  - Detected class skew distribution
- Reduce detection overhead by
  - Window sampling that considers a subset of time windows
  - Reuse computation in adjacent time windows

Bayesian Filter

Goal:
- Efficiently adapting CNNs toward the detected class skew during runtime
- Allowing the adapted CNNs to recognize classes out of the current CSP

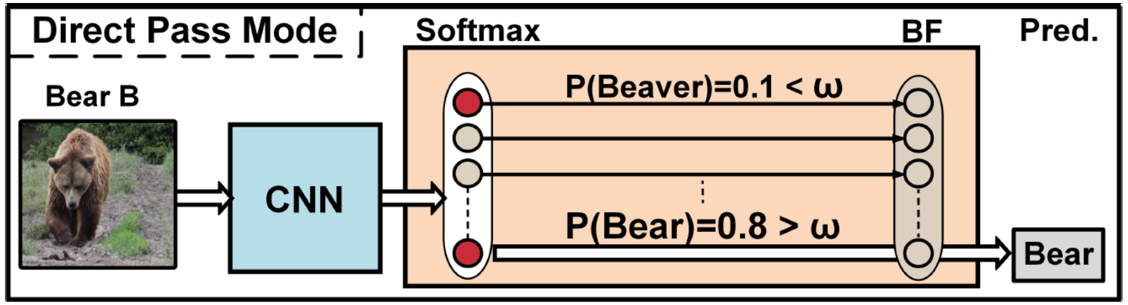


Rescaling Mode
- Intuition:
  - Update the probability of each prediction based on both the current CSP and the predicted probability
- Given:
  - P(i): profiled probability of class *i* in the current CSP
  - P(X|i): CNN predicted probability that an image X comes from class *i*
- Generate:
  - Posterior probability

$$P(i|X) \propto P(i) \cdot P(X|i)$$

Bayesian Filter

Goal:
- Efficiently adapting CNNs toward the detected class skew during runtime
- Allowing the adapted CNNs to recognize classes out of the current CSP



Direct Pass Mode
- Observation:
  - When a model makes a prediction with high probability ($> \omega$), this prediction tends to be correct
- Strategy:
  - Select the original CNN prediction without rescaling when the predicted probability is higher than a pre-selected threshold ω
- Formally:

$$P(i|X) \propto \begin{cases} P(i) \cdot P(X|i) & \text{if } P(X|i) < \omega \\ P(X|i) & \text{if } P(X|i) \geq \omega \end{cases}$$

Separability-Aware Model Selection

Key observation:
- The same model under different CSP may have significantly different accuracy
  - Even with the same number of classes

Strategy:
- Maintain a set of models with different accuracy and energy consumption
- Automatically switch to compact models for saving energy when the detected CSP is easy to classify

Easier to classify

Harder to classify

Separability-Aware Model Selection

Key observation:

- The same model under different CSP may have significantly different accuracy
  - Even with the same number of classes

Strategy1: Efficient Online Model Selection

- **Model selection on the cloud for only class skews detected during runtime**
  - On the cloud, profiling CNN accuracy for the detected CSP
  - Using binary search for acceleration
- **Cache service to avoid redundant model selection**
  - Record the model selection results along with the CSP
  - Skip model selection for a CSP that have appeared previously

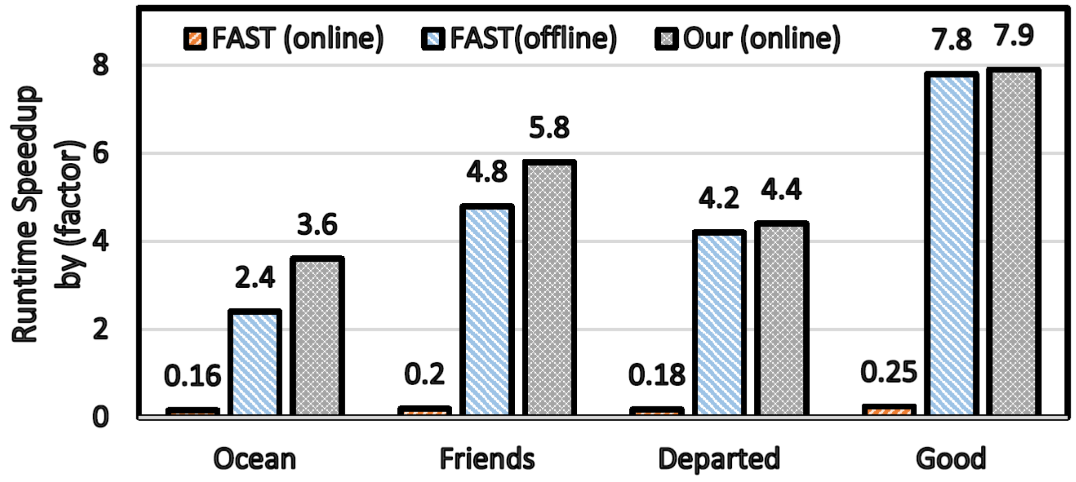
Strategy2: Edge-Cloud Duplicated Model Bank

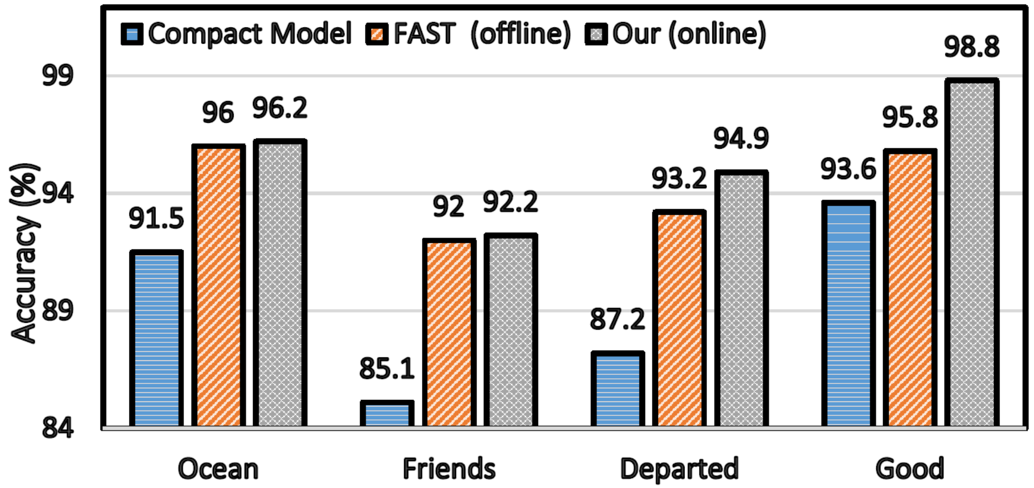
- **Model bank generation with offline profiling**
  - Offline profile CNNs and select only models on the Pareto-optimal curve
  - Conduct once on all CSPs and keep top-k best to reduce online overhead
- **Edge/cloud duplication to reduce network overhead**
  - Maintain a duplicated model bank on both the edge and the cloud
  - The cloud select a model and only send index of the selected model to the edge

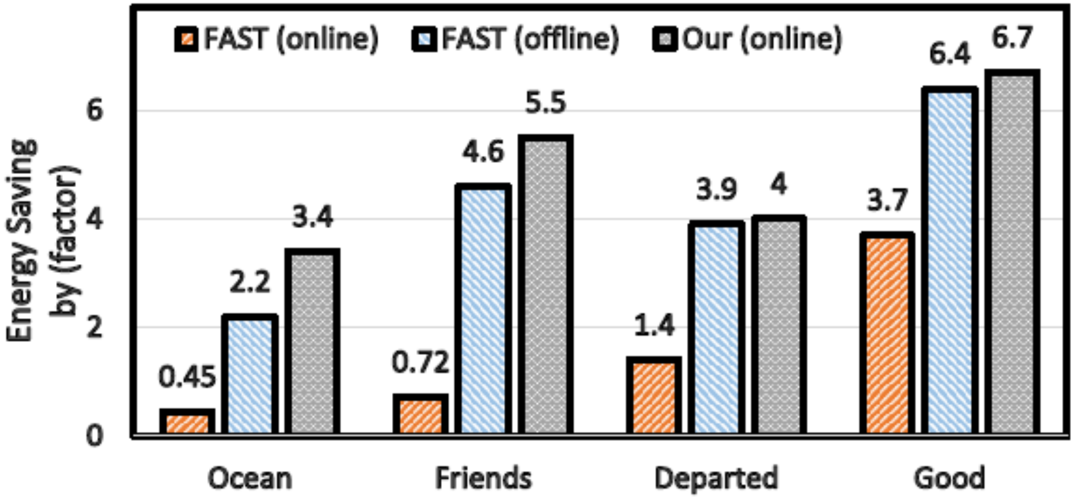
Evaluation

Dataset: Four real videos with
- 6~24 minutes
- 8~45 class skew switches
- 2.0 ~ 3.5 classes in each class skew

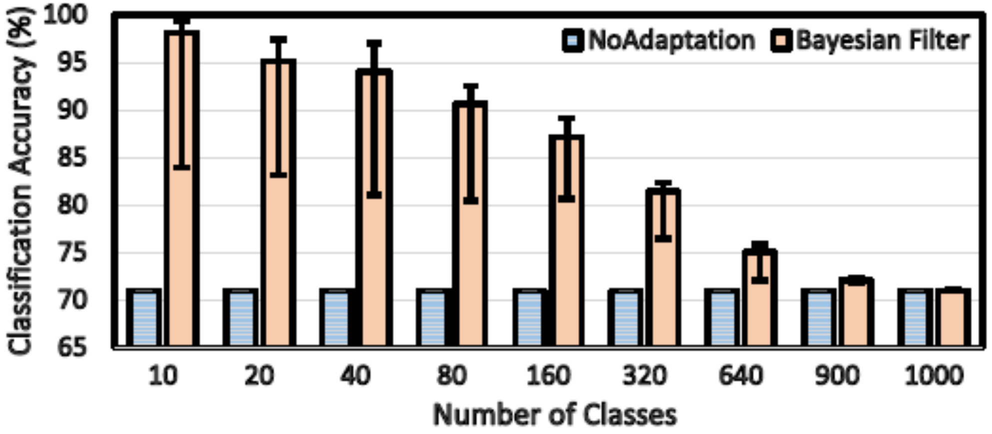
Platform:
- NVIDIA Jetson Nano as the edge device
- Dell Workstation T7910 as the cloud server



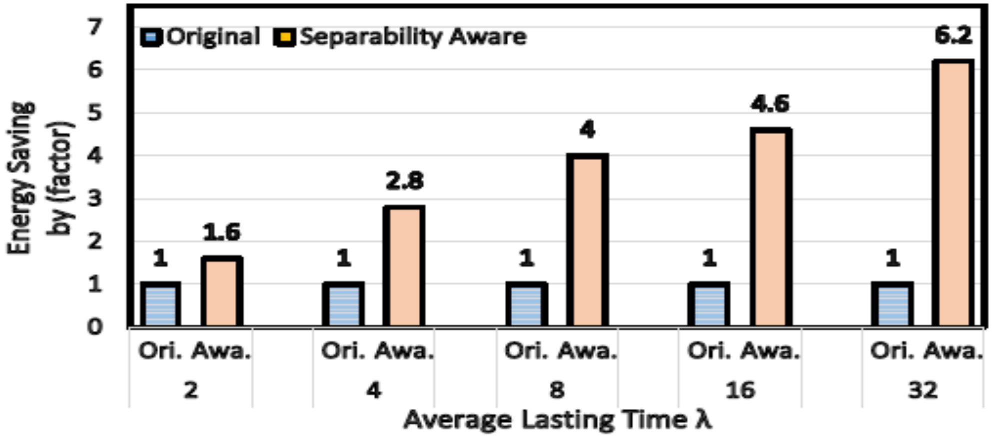




Evaluation



Accuracy improvement in an ideal case that class skew is known and fixed. Dataset: ImageNet. Model: MobileNet



Energy saving on synthesized class skews with diverse lasting time

Thanks!