FlexDNN: Input-Adaptive On-Device Deep Learning for Efficient Mobile Vision

ACM/IEEE Symposium on Edge Computing (SEC)

Biyi Fang, Xiao Zeng, Faen Zhang,
Hui Xu and Mi Zhang

Mobile Vision Systems are Revolutionizing Our Lives Now

Smartphones

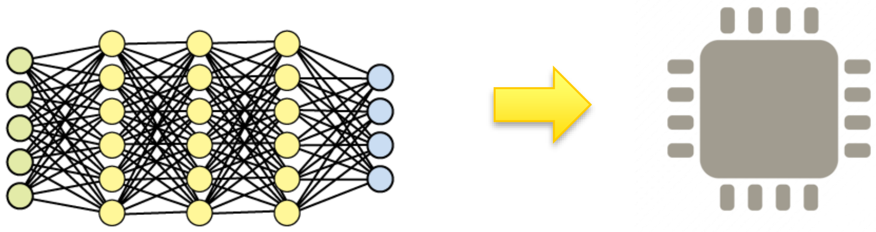
Drones

AR/VR Headset

Robots

Challenge

- Challenge: Each application (DNN) is resource demanding.



- A typical image recognition DNN designed for server/cloud takes up to **hundreds of milliseconds** to compute in mobile devices.
- This is **unacceptable** for video processing pipeline that requires high frame rate.

Typical Solutions

- **Model Compression Techniques**
  - Quantization, Pruning, Knowledge Distillation, Efficient Convolution Block.
- **Do not Take Advantage of the Dynamics of Mobile Video Inputs.**
  - Not all images are created equal.
  - Some images are ‘easy’ and some are ‘hard’ to recognize.
- **FlexDNN Leverages these Dynamics to further reduce resource demand.**
  - Complementary technique to model compression technique.

Dynamics of Mobile Video Inputs

Videos taken in real-world mobile settings show substantial dynamics in terms of difficulty level across frames over time.

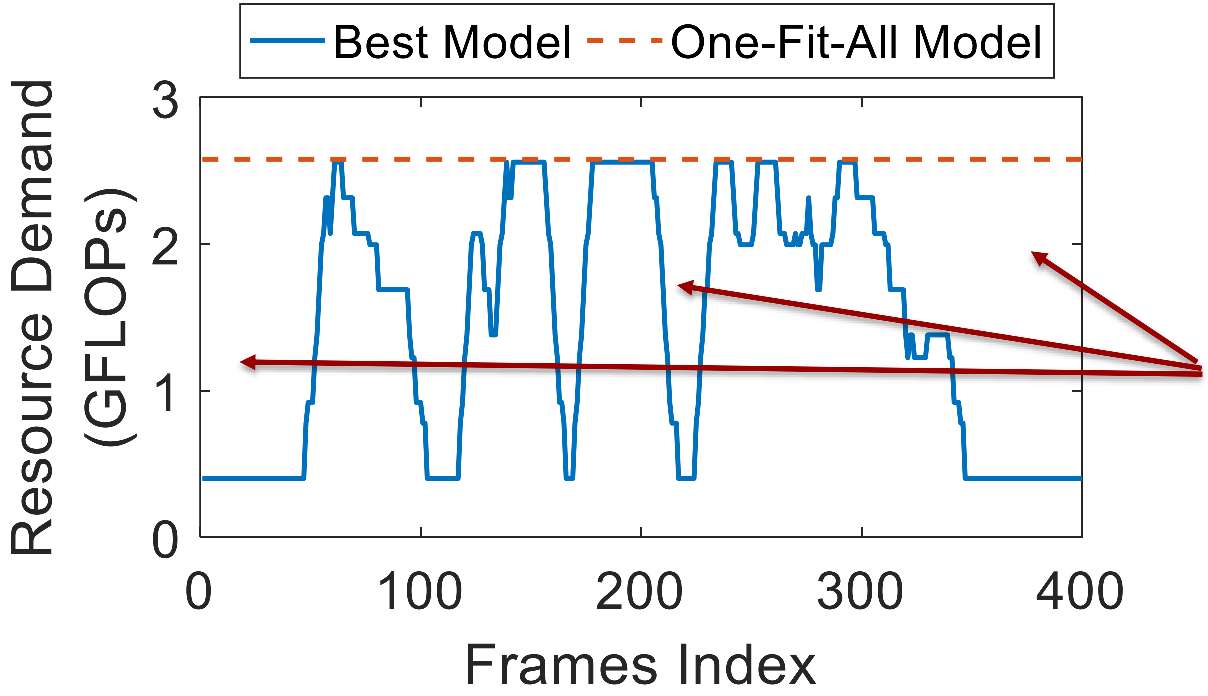
Relatively easier to be recognized as biking activity
Require less complex model

(a) (b) (c) (d)

Relatively harder to be recognized as biking activity
Require more complex model

Pilot Study: Dynamics of Resource Demand

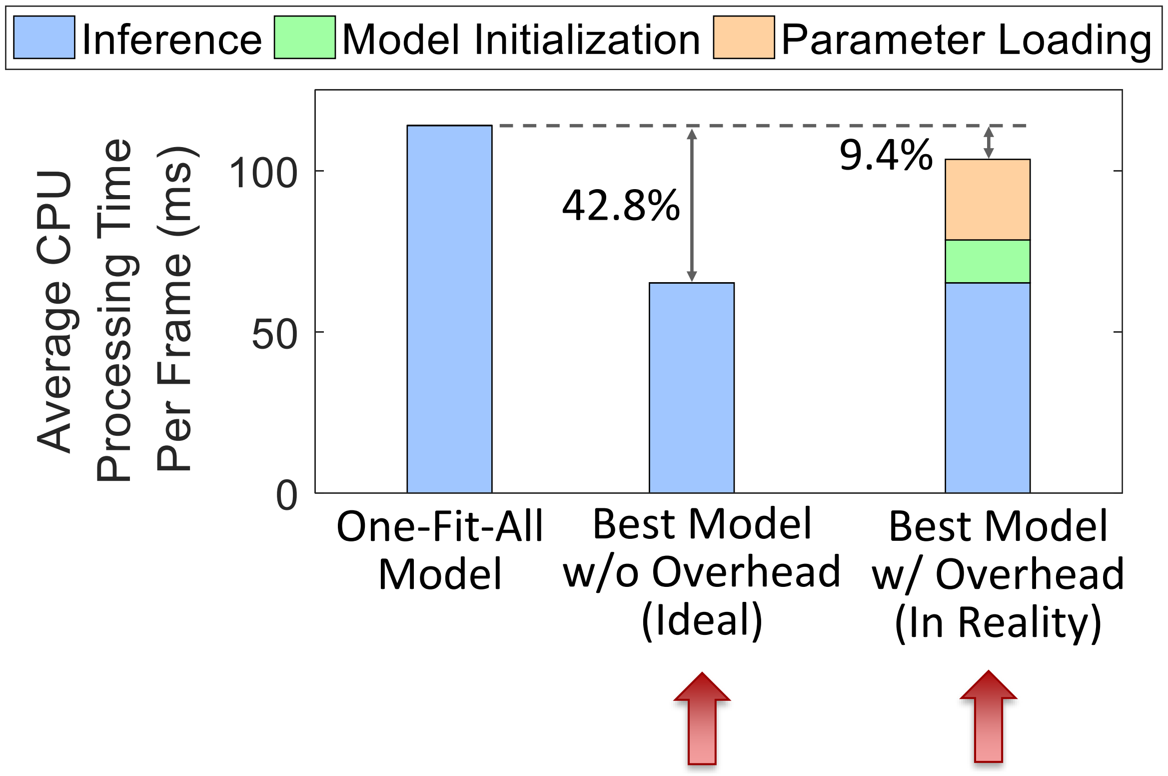
- Ten model variants with different complexities for a 400-frame video.
- Model with lowest complexity that correctly recognizes the activity (Best Model).
- Compare to the model that correctly recognizes all the frames (One-Fit-All Model).



- Best Model changes frequently.
- The difference area between curves indicate considerable resource demand that can be reduced.

Pilot Study: Quantify the Benefit of Leveraging the Dynamics

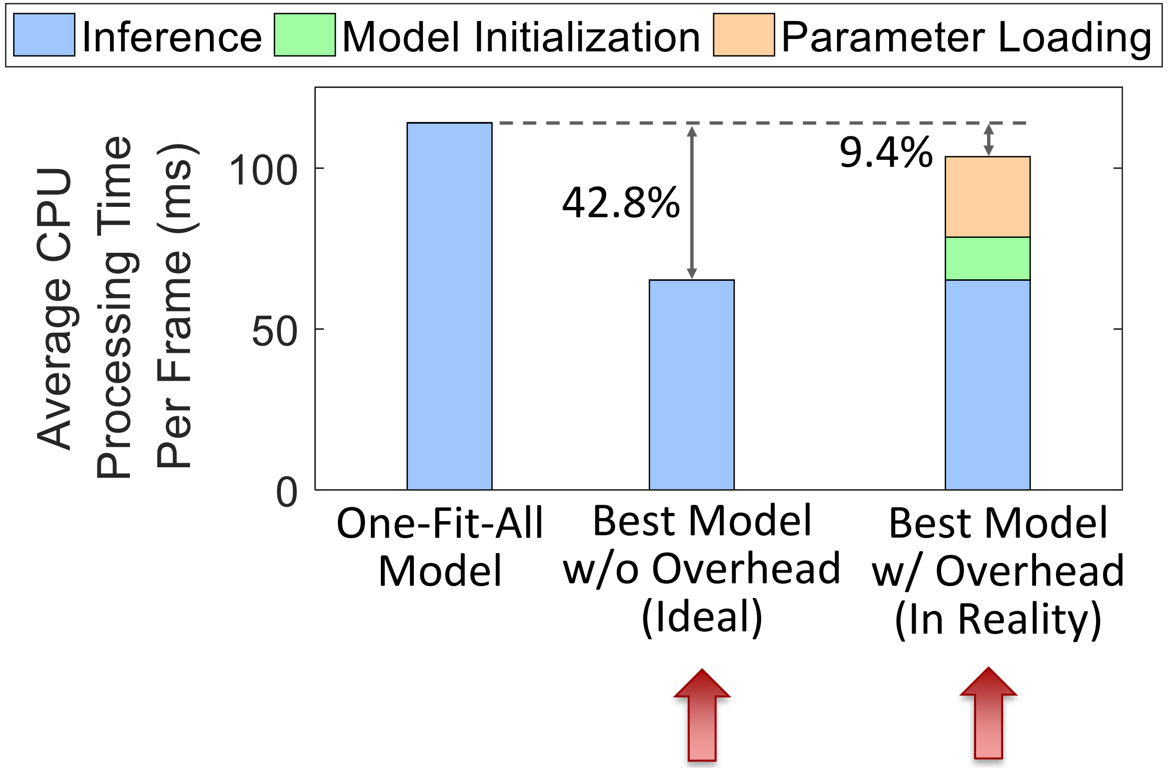
- Quantify the benefit in average CPU processing time of each frame (Samsung S8).
- Compare One-Fit-All Model and Best Model.
- In reality, model switching causes extra overhead.



- We can reduce resource demand in terms of inference time by 42.8%.
- Parameter loading and model initialization time take away the benefit by 21.8% and 11.6%.
- Actual gain is only 9.4%

Input-Adaptative On-Device Deep Learning

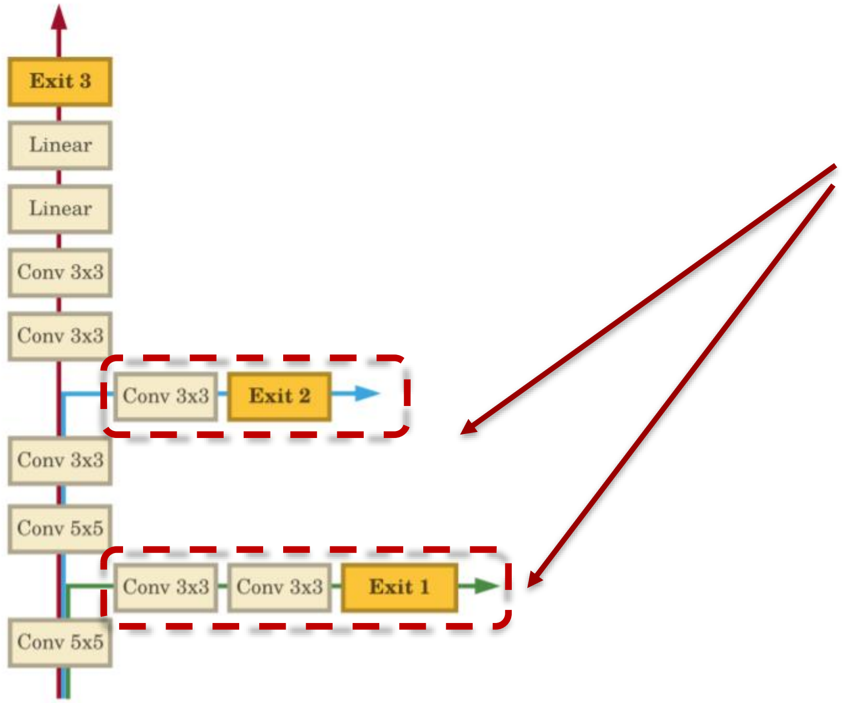
- No model switching overhead (Ideal).



State-of-the-art Input-Adaptive Works

- BranchyNet

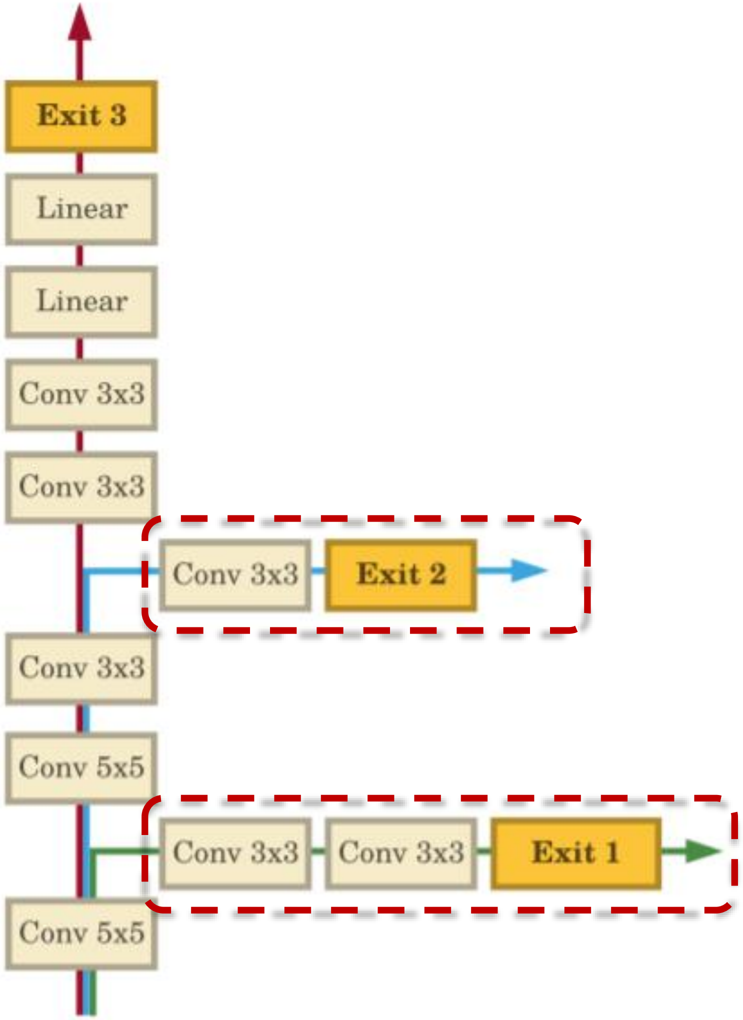
  [Teerapittayanon et al.*ICPR'16*]



Insert early exit branches into a backbone model and hence is not limited to certain types of model. FlexDNN follows this line of input-adaptive works.

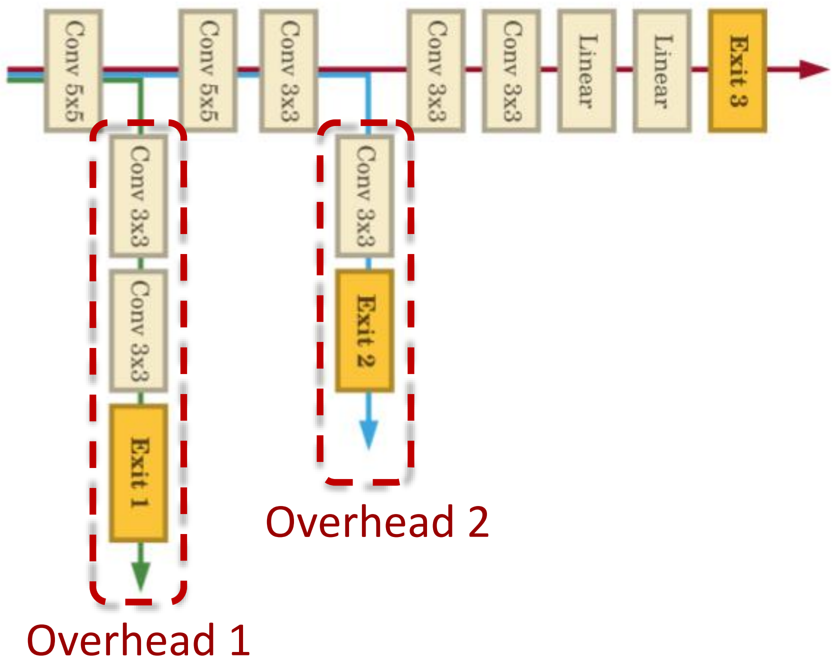
Early Exit Technique

- Early exit is a classifier with convolutional layer(s) and linear layer(s) that are inserted at the early layers of a backbone DNN.
- Able to identify and exit easy inputs without causing further computation.
- In doing so, the average computational consumption can be lower than the backbone DNN without inserting any early exit.



Drawback (BranchyNet)

- The way BranchyNet design their early exit branches brings two drawbacks:
  - Early exit itself consumes computation. Without careful design, it leads to suboptimal performance of the input-adaptive model.
  - Inserting larger amount of early exit will make the model less efficient by latency cumulation.

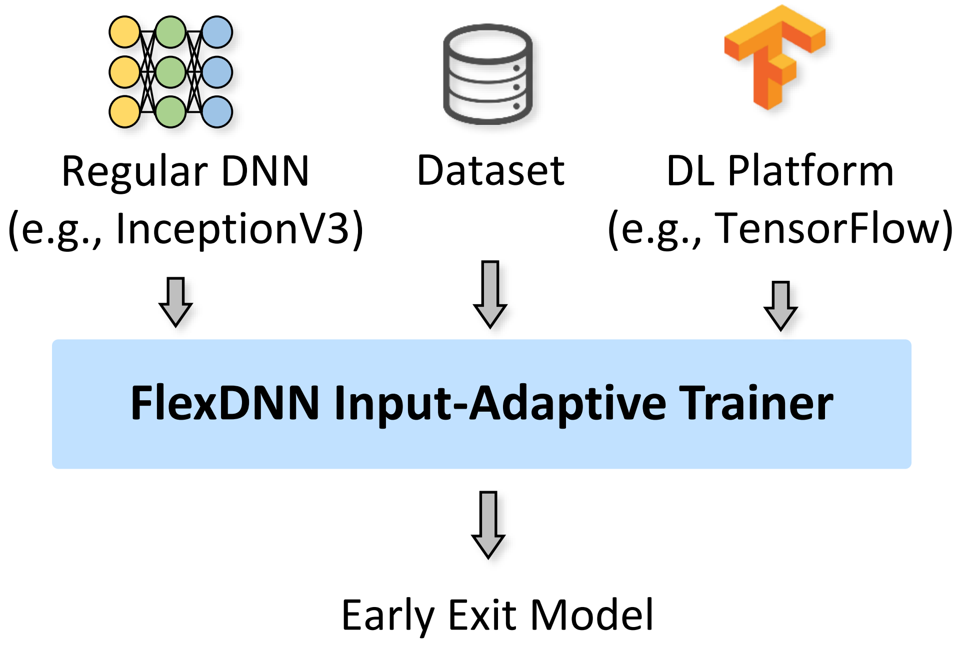


Hard Inputs:
Overhead 1 + Overhead 2

Overview of FlexDNN

- A novel input-adaptive framework that enables computation-efficient DNN-based on-device DL based on early exit mechanism.

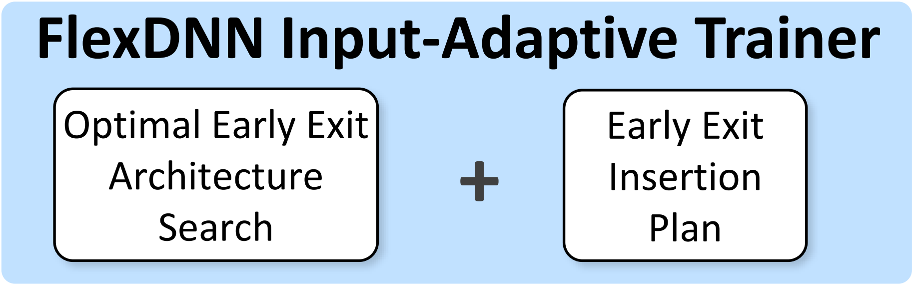
- As an overview, FlexDNN is a technique that inserts early exits with optimal architecture at optimal locations of a backbone DNN.



FlexDNN Input-Adaptive Trainer

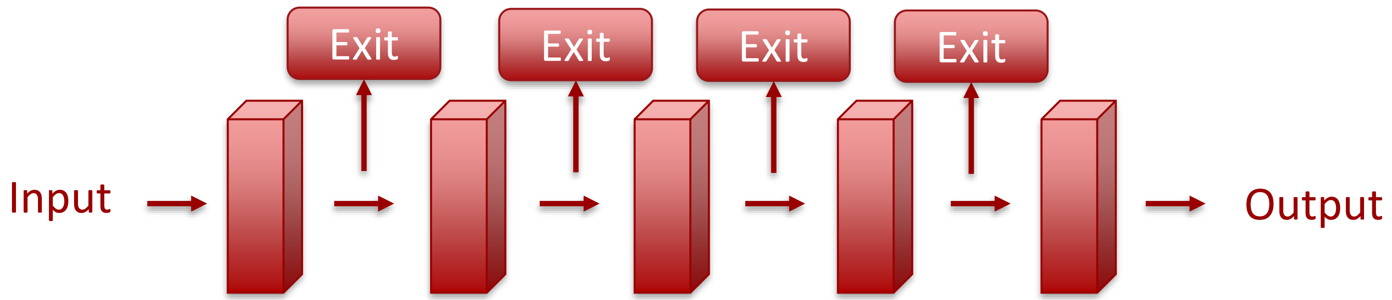
- Component #1: Optimal Early Exit Architecture Search

- Component #2: Early Exit Insertion Plan



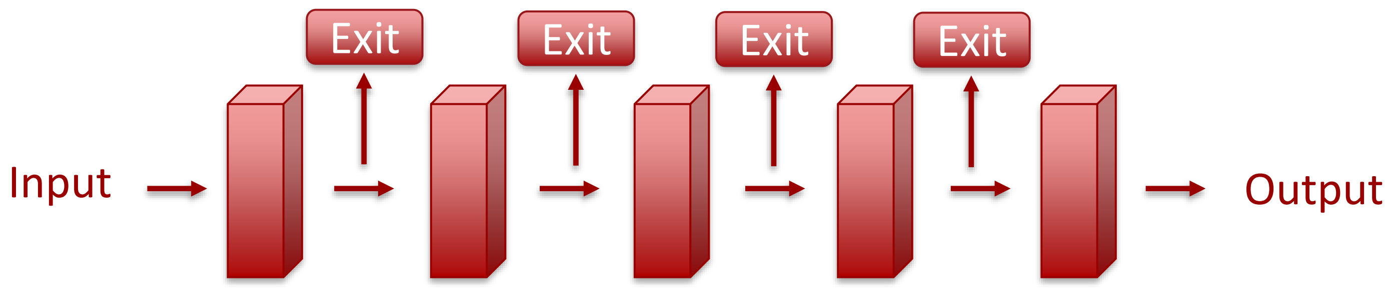
#1 Optimal Early Exit Architecture Search

- Motivation: early exits consume overhead. Hence, a lightweight early exit is preferred. However, an extremely lightweight early exit could exit much less easy frames, which diminishes the benefit of early exit.
- FlexDNN inserts over-parameterized early exit branches at each possible location and prune the filters and layers until the accuracy of the early exit starts to drop.
- As a result, the architecture of each inserted early exit achieves optimal trade-off between early exit rate and computational overhead.



#2 Early Exit Insertion Plan

- Motivation: by far early exits have been inserted at each possible location throughout the DNN model and hence accumulate immense overhead altogether.
- FlexDNN adopts a systematic approach to derive an optimal insertion plan of early exits.
- We prune the most inefficient early exits.



#2 Early Exit Insertion Plan

- To identify the most inefficient early exits, we define a metric *R* that quantifies the quality of the trade-off between early exit rate and computational overhead of a particular early exit.
- We remove early exits whose *R* values are less than or equal to 1.

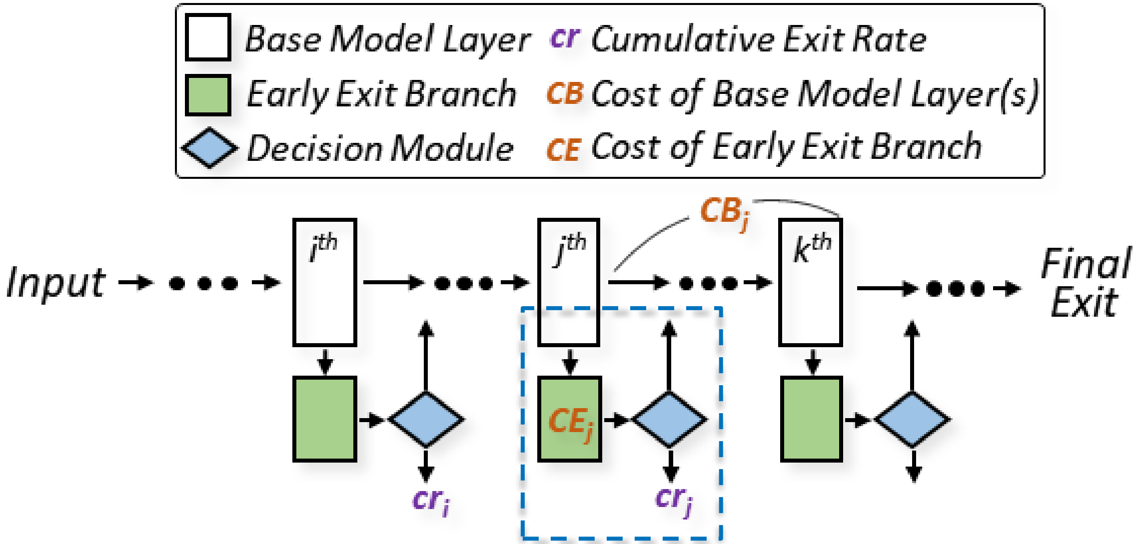
$$R_j = G_j / C_j$$

$$C_j = \boxed{N * (1 - cr_i)} * CE_j$$

Number of frames cannot exit before this exit

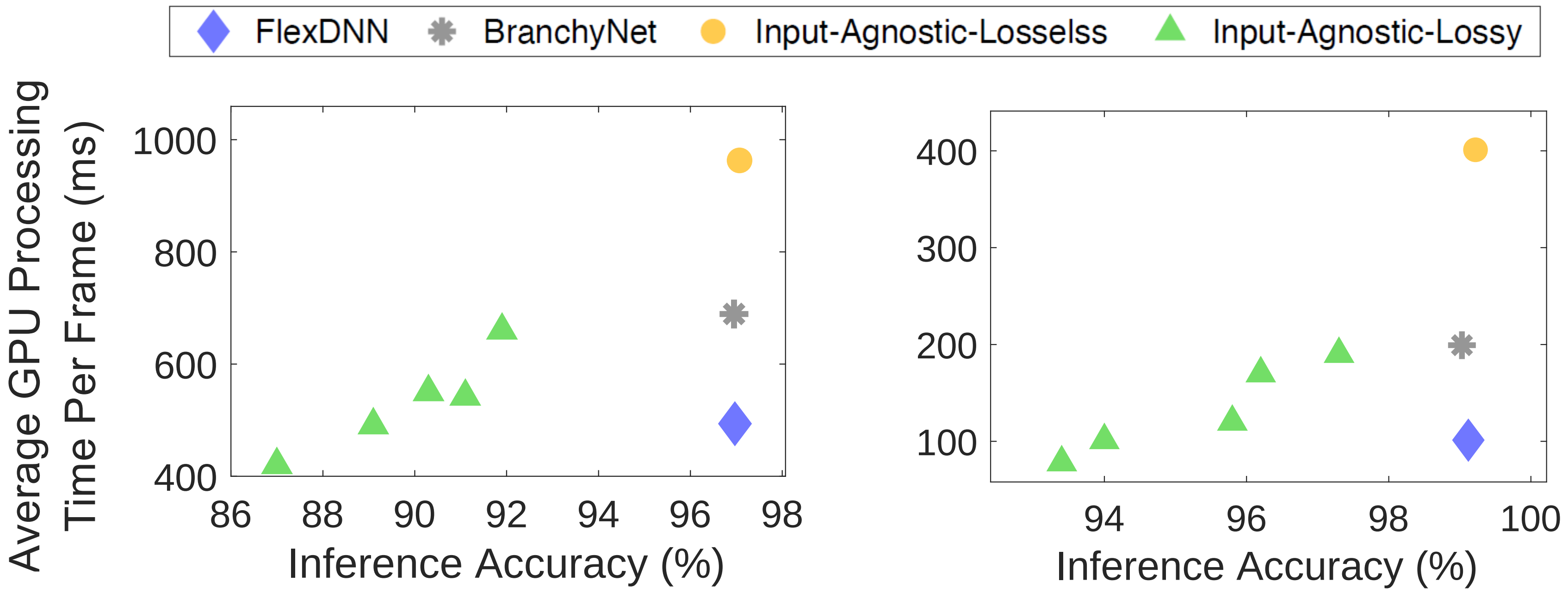
$$G_j = \boxed{N * (cr_j - cr_i)} * CB_j$$

Number of frames successfully exit at this exit



Evaluation

- Evaluation is on UCF-101 derived dataset.
- Backbone: VGG-16 and Inception-V3.
- Experiments are conducted on Samsung S8.

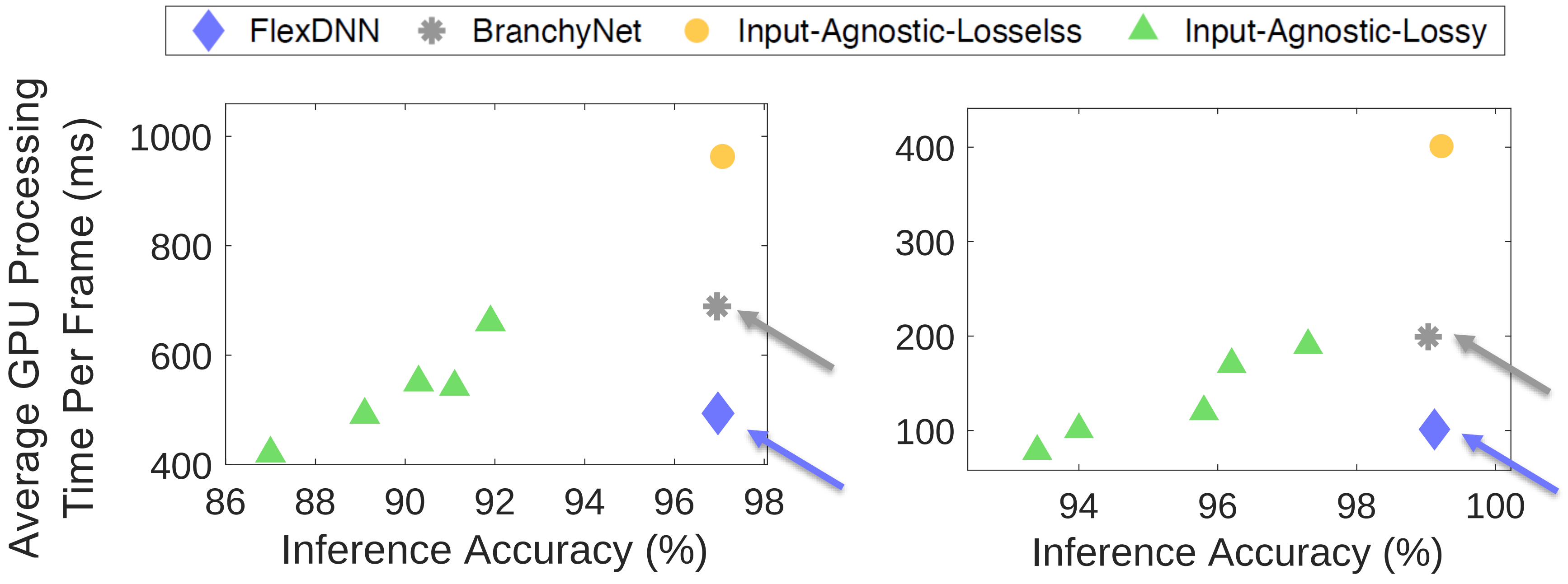


VGG-16

Inception-V3

Evaluation: Compared to BranchyNet

- Baselines: 1) BranchyNet; 2) Input-Agnostic-Lossless; 3) Input-Agnostic-Lossy
- Results: Compared to BranchyNet, FlexDNN reduces 28.4% and 49.3% on VGG and Inception-V3, respectively.



VGG-16

Inception-V3

Contribution of FlexDNN

- An input-adaptive framework for computation-efficient DNN-based mobile video stream analytics that achieves better performance compared to state-of-the-art counterparts.

- FlexDNN addresses the limitations of existing solutions and pushes the state-of-the-art forward through the approach for generating the optimal architecture based on early exits for input adaptation.

- We experimentally demonstrate the effectiveness of input-adaptive for on-device DL.

Thank You

Biyi Fang

fangbiyi@msu.edu

Mi Zhang

mizhang@egr.msu.edu