Caffe-SSD Inference on Edge Device Using TVM and Hybrid Script

Masahiro Hiramori

Hiramori.Masahiro@ct.MitsubishiElectric.co.jp

MITSUBISHI ELECTRIC CORPORATION

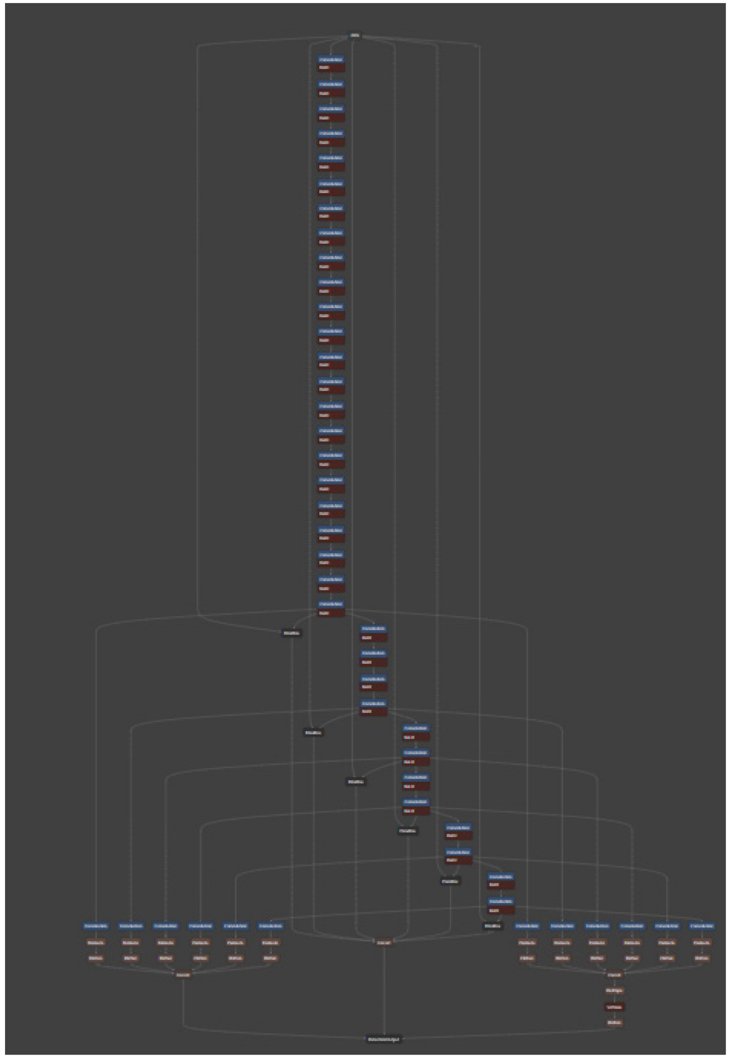
Motivation and Problem

Motivation

- Object Detection (OD) is a computationally expensive task
  - Needs performance optimization to run on the edge devices
- Our internal OD model is based on models created by the Caffe-SSD
  - Caffe-SSD: Caffe’s implementation of Single Shot Multibox Detector

Problem

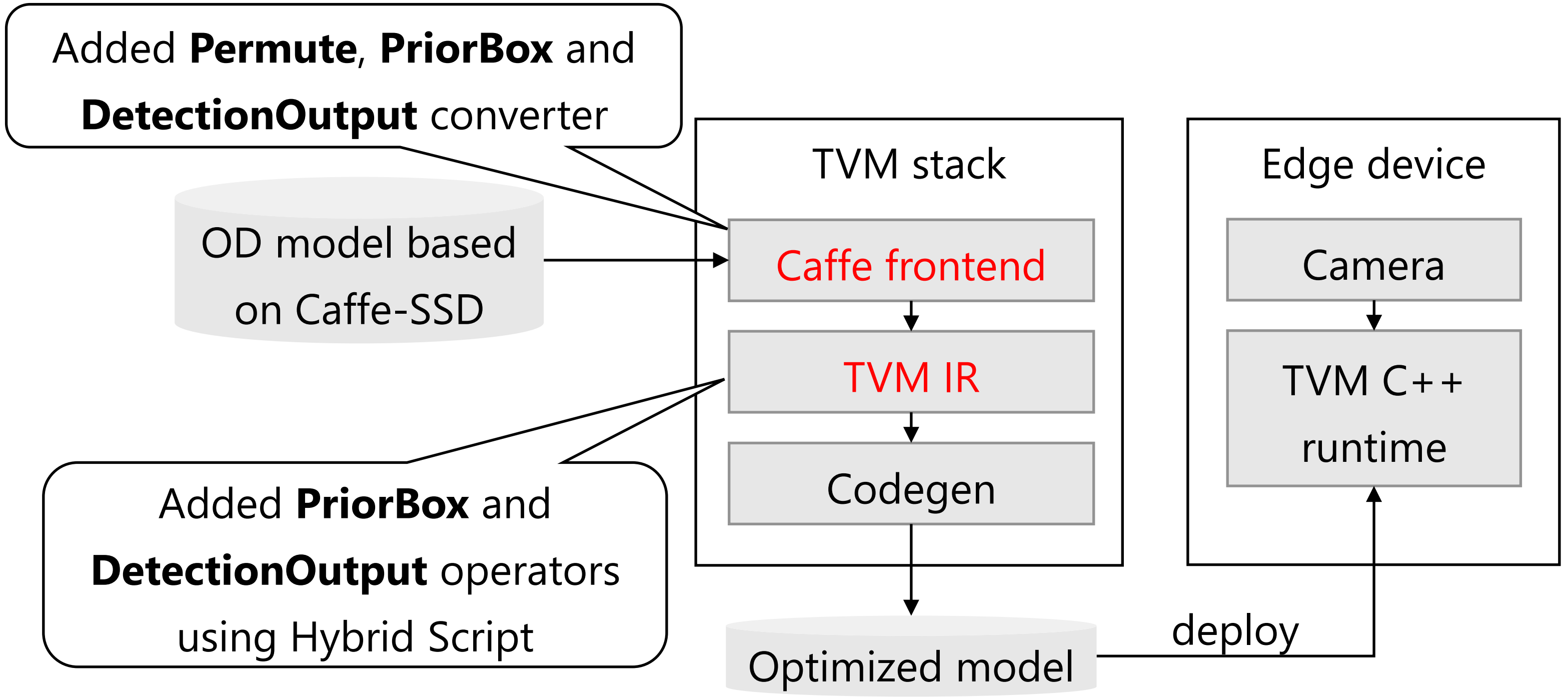
- TVM **cannot** import models created by the Caffe-SSD
  - **Permute** layer, **PriorBox** layer, and **DetectionOutput** layer are not supported by the Caffe Frontend in TVM



OD Model Architecture

Implementation

- Add **Permute**, **PriorBox**, and **DetectionOutput** layer converters to existing **Caffe frontend**
  - **Permute** can be converted to **tvm.relay.transpose** operator
- Add **PriorBox** and **DetectionOutput** operators to **TVM IR** using Hybrid Script
  - Why?-> There exists equivalent Relay operators (e.g. vision.multibox_prior, vision.non_max_suppression). However, none of them are 100% compatible with Caffe-SSD's **PriorBox** and **DetectionOutput** layers

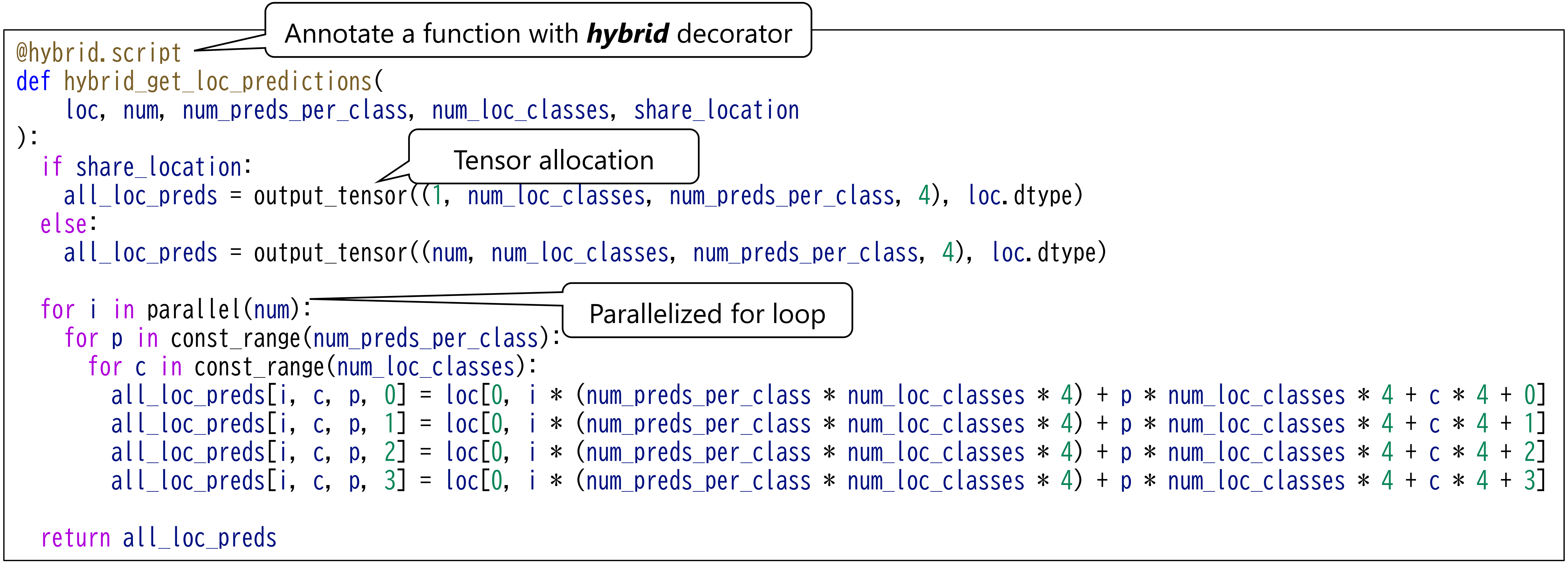


Added **Permute**, **PriorBox** and **DetectionOutput** converter

Added **PriorBox** and **DetectionOutput** operators using Hybrid Script

Hybrid Script

- Hybrid Script is a DSL for constructing TVM IR in Python
  - Subset of Python language with some extensions



```
@hybrid.script
def hybrid_get_loc_predictions(
    loc, num, num_preds_per_class, num_loc_classes, share_location
):
  if share_location:
    all_loc_preds = output_tensor((1, num_loc_classes, num_preds_per_class, 4), loc.dtype)
  else:
    all_loc_preds = output_tensor((num, num_loc_classes, num_preds_per_class, 4), loc.dtype)

  for i in parallel(num):
    for p in const_range(num_preds_per_class):
      for c in const_range(num_loc_classes):
        all_loc_preds[i, c, p, 0] = loc[0, i * (num_preds_per_class * num_loc_classes * 4) + p * num_loc_classes * 4 + c * 4 + 0]
        all_loc_preds[i, c, p, 1] = loc[0, i * (num_preds_per_class * num_loc_classes * 4) + p * num_loc_classes * 4 + c * 4 + 1]
        all_loc_preds[i, c, p, 2] = loc[0, i * (num_preds_per_class * num_loc_classes * 4) + p * num_loc_classes * 4 + c * 4 + 2]
        all_loc_preds[i, c, p, 3] = loc[0, i * (num_preds_per_class * num_loc_classes * 4) + p * num_loc_classes * 4 + c * 4 + 3]

  return all_loc_preds
```

Annotate a function with ***hybrid*** decorator

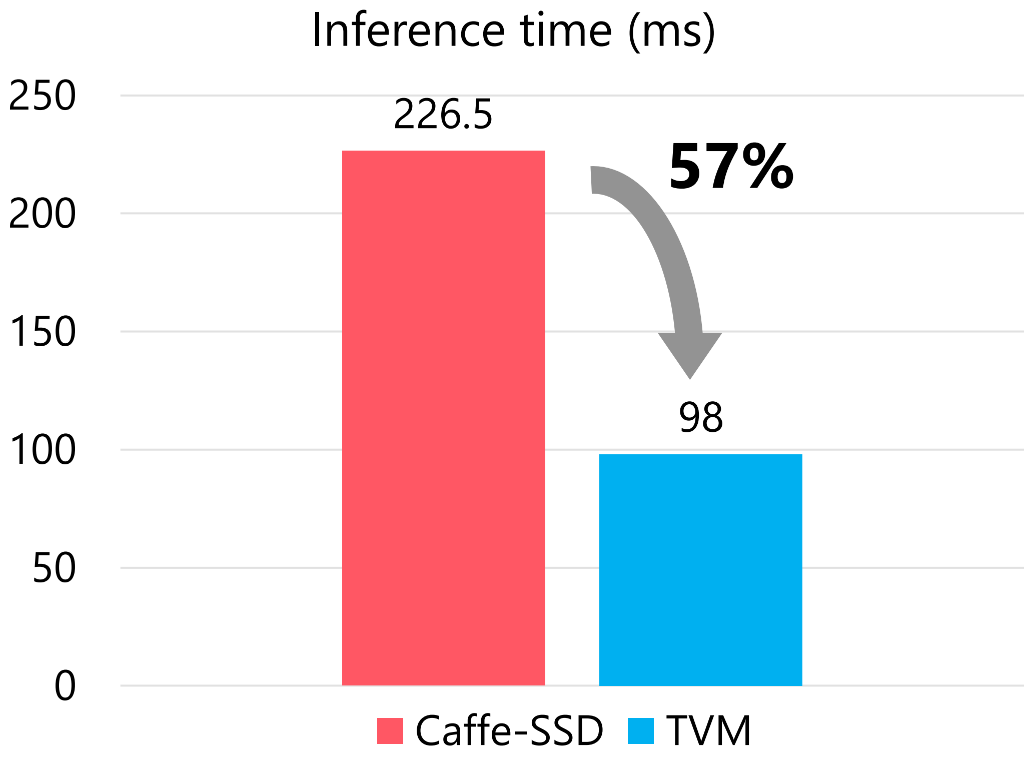
Tensor allocation

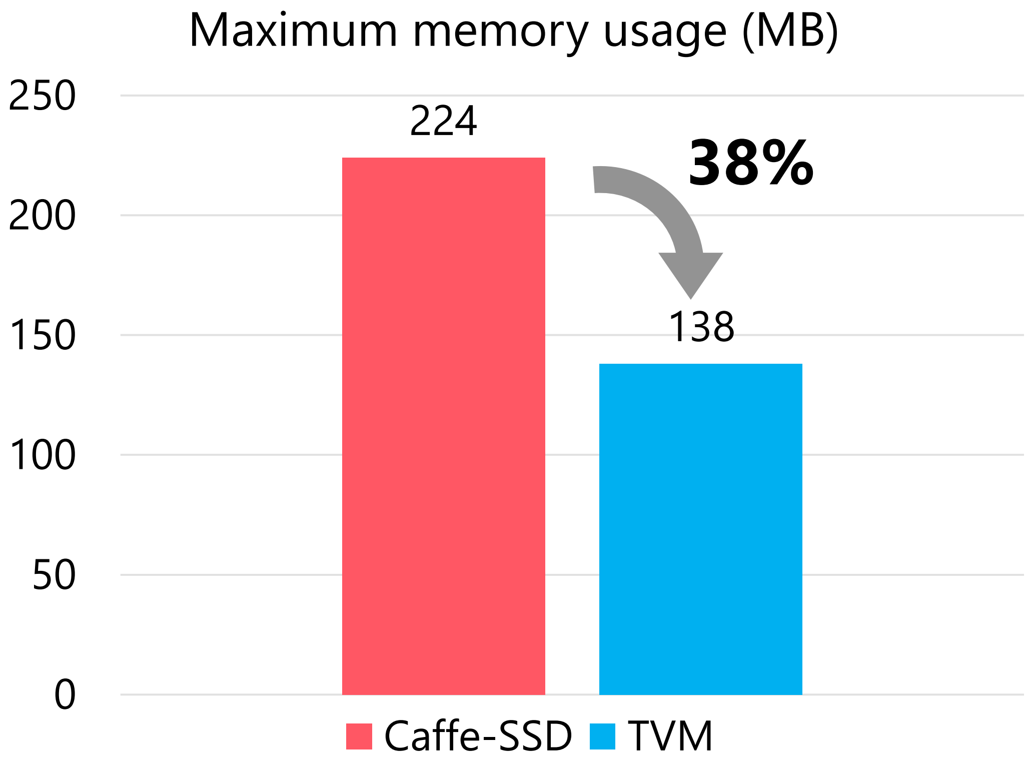
Parallelized for loop

Implementation of the DetectionOutput operator using Hybrid Script (code fragment)

Experimental results

- Evaluated on Raspberry Pi 4 with Debian 10 buster
- Single image inference time of TVM optimized model is **57% shorter** than that of Caffe-SSD
- Maximum memory usage of TVM is **38% lower** than that of Caffe-SSD





Conclusions and Future works

Conclusions

- **Motivation**: want to optimize our internal OD model for the edge devices
- **Problem**: TVM couldn't compile models created by the Caffe-SSD
- **Idea**: added support for missing operators to TVM's Caffe frontend using **Hybrid Script**
- **Results**: inference time is **57%** faster and maximum memory usage is **38%** lower than Caffe-SSD

Future works

- Apply auto-tuning (AutoTVM / AutoScheduler)
- Contribute our implementation to the upstream
  - [CI][Caffe Frontend] Change the caffe deps into SSD distribution #9060
  - [Caffe Frontend] Add support for Permute layer #9157