Couper: DNN Model Slicing for Video Analytics Containers at the Edge

Ke-Jou (Carol) Hsu
Ketan Bhardwaj
Ada Gavrilovska
Video analytics applications are in high demand
Video analytics applications are in high demand.
Video analytics applications are in high demand.

Video analytics application may face great performance degradation because of its data-intensive and latency-sensitive workload.
Edge’s proximity benefit can help!

Edge computing brings benefits:

- Higher computing resource than client
- Reduce communication cost, lower processing latencies, higher processing rates, …
- Flexible service deployment
How does video analytics application work with edge?
How does video analytics application work with edge?

Deep neural network (DNN)
How does video analytics application work with edge?

Deep neural network (DNN)

High accuracy and famous
How does video analytics application work with edge?

Deep neural network (DNN)

High accuracy and famous

Computation-intensive workload
How does video analytics application work with edge?

Deep neural network (DNN)

High accuracy and famous

Computation-intensive workload

<table>
<thead>
<tr>
<th>Model</th>
<th>VGG 16</th>
<th>MobileNet V2 1.4</th>
<th>ResNet V2 50</th>
<th>Inception V3</th>
<th>Inception ResNet V2</th>
<th>NASNet 331</th>
<th>PNASNet 331</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top-1 Accuracy</td>
<td>71.5</td>
<td>74.9</td>
<td>75.6</td>
<td>78.0</td>
<td>80.4</td>
<td>82.7</td>
<td>82.9</td>
</tr>
<tr>
<td># Operators</td>
<td>54</td>
<td>155</td>
<td>205</td>
<td>788</td>
<td>871</td>
<td>1265</td>
<td>939</td>
</tr>
</tbody>
</table>

Accuracy increases, so does model complexity
How does video analytics application work with edge?

**Deep neural network (DNN)**

- Google, Cliff Young (Linley processor conference 2018)
How does video analytics application work with edge?

**Deep neural network (DNN)**

- Google, Cliff Young (Linley processor conference 2018)

Single type of device cannot fit **every DNN**, more accurate DNNs require more resource.
How does video analytics application work with edge?

Deep neural network (DNN)

Client -> Edge -> Cloud
How does video analytics application work with edge?

Deep neural network (DNN)

Client -> Edge -> Cloud

Diverse specification and network distance
Bringing out edge’s benefit is not easy
Bringing out edge’s benefit is not easy

If edge cannot run whole DNN:
Bringing out edge’s benefit is not easy

DNN

Client -> Edge -> Cloud

If edge cannot run whole DNN:

Optimize DNN for edge
Bringing out edge’s benefit is not easy

If edge cannot run whole DNN:

Optimize DNN for edge
Bringing out edge’s benefit is not easy

DNN

Client -> Edge -> Cloud

If edge cannot run whole DNN:

Optimize DNN for edge

or

Bring specific edge for DNN
Bringing out edge’s benefit is not easy

DNN

Client -> Edge -> Cloud

If edge cannot run whole DNN:

Optimize DNN for edge

or

Bring specific edge for DNN
Bringing out edge’s benefit is not easy

DNN ? Client -> Edge -> Cloud

If edge cannot run whole DNN:
- Optimize DNN for edge
- Bring specific edge for DNN

These two methods are relatively **time- and money-consuming** and turns to be **impractical** for **rapid growth** of DNNs and **diverse** and **shared** edge environment.
Problem Statement

This is a multi-dimensional problem:

1. **Heterogeneous computing resource** on client-edge-cloud.
2. **Various** compute-intensive **DNN** models
3. **No single deployment** meets users’ expectation **forever**
Problem Statement

This is a multi-dimensional problem:

1. Heterogeneous computing resource on client-edge-cloud.
2. Various compute-intensive DNN models
3. No single deployment meets users’ expectation forever

Given a DNN and an edge,

How can we deploy the model with good performance?
Problem Statement

This is a multi-dimensional problem:

1. **Heterogeneous computing resource** on client-edge-cloud.
2. **Various** compute-intensive **DNN** models
3. **No single deployment** meets users’ expectation **forever**

Given a **DNN** and an edge,

How can we deploy the model with good performance?

**Couper**: a general edge system finding (and deploying) a good DNN deployment for you!
Share load across edge and cloud by DNN partitioning
Share load across edge and cloud by DNN partitioning
Share load across edge and cloud by DNN partitioning
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?

LeNet (1998)

http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?

LeNet (1998)

AlexNet (2012)

http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?

LeNet (1998)

AlexNet (2012)

GoogLeNet (2014)

http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?

LeNet (1998)

AlexNet (2012)

GoogLeNet (2014)

Inception V3 (2015)

http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/
Share load across edge and cloud by DNN partitioning

How do we decide the slicing point?

LeNet (1998)

AlexNet (2012)

GoogLeNet (2014)

Inception V3 (2015)


http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/
Share load across edge and cloud by DNN partitioning

How do we decide the partition point?

1. Filter out splittable candidates in DNN
2. Pick up a right one among the candidates
Listing splicing candidates

Placeholder ➔ Pad ➔ MaxPool | Conv2D ➔ ReLu ➔ Squeeze ➔ Reshape

paddings ➔ weights
Listing splicing candidates

- Placeholder
- Pad
- MaxPool
- Conv2D
- ReLu
- Squeeze
- Reshape

Multi-parallel path
Listing splicing candidates

- Placeholder
- Pad
  - paddings
- MaxPool
- Conv2D
  - weights
- ReLu
- Squeeze
- Reshape

- Multi-parallel path
- Constant or reading operator
Listing splicing candidates

- Placeholder
- Pad
  - paddings
- MaxPool
- Conv2D
  - weights
- ReLu
- Squeeze
- Reshape

- Multi-parallel path
- Constant or reading operator
- Last operator
Listing splicing candidates

- Placeholder
- Pad
  - paddings
- MaxPool
- Conv2D
  - weights
- ReLu
- Squeeze
- Reshape

- Multi-parallel path
- Constant or reading operator
- Last operator
Listing splicing candidates

- Placeholder → Pad
  - Paddings
- MaxPool
- Conv2D
  - Weights
- ReLu → Squeeze → Reshape

Take “Pad” as slicing point!

Running on edge
Running on cloud

- Multi-parallel path
- Constant or reading operator
- Last operator
Evaluating splicing candidates
Evaluating splicing candidates

**Strongman**

Evaluate every candidate

```
1 2 3
```
Evaluating splicing candidates

**Strongman**
Evaluate every candidate

**Comm-slim**
Bypass candidates with high networking cost

1. Evaluate every candidate
2. Bypass candidates with high networking cost
Evaluating splicing candidates

Strongman
Evaluate every candidate

Comm-slim
Bypass candidates with high networking cost

Early-stop
Stop evaluation when edge is overload
Evaluating splicing candidates

Strongman
Evaluate every candidate

Comm-slim
Bypass candidates with high networking cost

Early-stop
Stop evaluation when edge is overload

Hybrid
Combination of comm-slim and early-stop
Couper Overview

Staging Environment

Model -> Model Slicer
  Slicing algorithm
  List of split points

App -> Application Wrapper
  Container images

Resource configuration -> Slice Evaluator
  Method
  Best split point

Publisher

Containers

Production Environment
Staging Environment

Model Slicer
- Slicing algorithm
- List of split points

Application Wrapper
- Container images

Slice Evaluator
- Method
- Best split point

Publisher
- Containers

Production Environment

Couper Overview

Model

App

Resource configuration
Couper Overview

Staging Environment

- Model
- App
- Resource configuration

Model Slicer
- Slicing algorithm
- List of split points

Application Wrapper
- Container images

Slice Evaluator
- Method
- Best split point

Publisher
- Containers

Production Environment

- Kubernetes icon

Resources configuration
- Model
- App

Couper Overview
Couper Overview

Staging Environment

- Model
- App
- Resource configuration

Model Slice
- Slicing algorithm
- List of split points

Application Wrapper
- Container images

Slice Evaluator
- Method
- Best split point

Publisher
- Containers

Production Environment
Results for different SLAs

Normalized processing latency per frame (%)

VGG 16
MobileNet V2
ResNet V2
Inception V3
Inception ResNet
NASNet
PNASNet

Normalized frame drop rate (%)

VGG 16
MobileNet V2
ResNet V2
Inception V3
Inception ResNet
NASNet
PNASNet
Results for different SLAs

placing all DNN inference on cloud

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet
Results for different SLAs

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet
Results for different SLAs

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

40-90% reduction

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet
Results for different SLAs

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Reduce network latency

40-90% reduction

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet
Results for different SLAs

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Reduce network latency

40-90% reduction

65-100% reduction

Reduce network latency
Results for different SLAs

Normalized processing latency per frame (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

Reduce network latency
40-90% reduction

Normalized frame drop rate (%)

- VGG 16
- MobileNet V2
- ResNet V2
- Inception V3
- Inception ResNet
- NASNet
- PNASNet

More balanced pipeline
65-100% reduction
<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Strongman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hybrid</td>
</tr>
<tr>
<td>Model</td>
<td># Operator</td>
<td>Method</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>--------</td>
</tr>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>Strongman: 34</td>
</tr>
</tbody>
</table>

99% reduction
Strongman method tests 34 slicing candidates

<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>Strongman 34</td>
</tr>
</tbody>
</table>

99% reduction
<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>Strongman: 34</td>
</tr>
<tr>
<td>Model</td>
<td># Operator</td>
<td>Method</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Inception V3</td>
<td>788</td>
<td></td>
</tr>
</tbody>
</table>

**Diagram:**
- **ML inference on cloud** (purple)
- **Data transmission** (yellow)
- **ML inference on edge** (blue)

**Legend:**
- **Shortest latency** (circle)

---

12
<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>Strongman</td>
</tr>
</tbody>
</table>

The Hybrid method can find the same slicing deployment with much smaller problem space.
<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Strongman</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>34</td>
<td>2</td>
</tr>
</tbody>
</table>

**Frame drop rate**

- **Model**: Inception V3
- **Method**: Strongman, Hybrid
- **# Operator**: 788
- **Values**: 34, 2

**Graph Details**

- **X-axis**: Second(s)
- **Y-axis**: ML inference on cloud, Data transmission, ML inference on edge, Shortest latency
- **Legend**:
  - Purple: ML inference on cloud
  - Orange: Data transmission
  - Blue: ML inference on edge
  - Green circle: Shortest latency
  - Black line: Frame drop rate

**Graph Description**

- The graph illustrates the latency and data transmission for different layers in a model.
- The y-axis represents the percentage of the total latency contributed by each layer.
- The x-axis represents the layers of the model.
- The graph includes a green circle indicating the shortest latency for each model variant.
- The black line indicates the frame drop rate across different layers.
<table>
<thead>
<tr>
<th>Model</th>
<th># Operator</th>
<th>Method</th>
<th>Strongman</th>
<th>Hybrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>34</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td># Operator</td>
<td>Method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>------------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>Strongman</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hybrid</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Frame drop rate

Strongman: 788
Hybrid: 34

Shortest frame drop rate

Not single slicing deployment for all SLAs

Hybrid is not found the best one, but a relatively good one

ML inference on cloud
Data transmission
ML inference on edge
Shortest latency
Shortest frame drop rate
Frame drop rate

Not single slicing deployment for all SLAs
Couper contribution

• **Improve DNN inference on various metrics:**
  Achieved up to 90% improvement on processing latency and 100% improvement on processing quality.

• **Rapid to find solution:**
  Reduced 99% problem space for searching best deployment.

• **Flexible to different DNN inference service:**
  Supported pluggable slicing algorithm and evaluating method.

• **Compatible with contemporary software stack:**
  Deployed with container orchestration, Kubernetes.
Thanks for your attention!
Running PNASNet on different edge

Spec level

- Low
- Mid
- High
- Extra-high
- Extra-high+GPU

- 10^0
- 10^1
- 10^2
- 10^3
- 10^4

Legend:
- camera
- transformer
- DNN evaluation
- classifier
This is a multi-dimensional problem:

1. Heterogeneous computing resource between client, edge and cloud.
2. Various compute-intensive DNN models

=> slicing the DNN to fit the edge resource
Here comes Couper!

This is a multi-dimensional problem:

1. Heterogeneous computing resource between client, edge and cloud.
2. Various compute-intensive DNN models

=> **slicing the DNN to fit the edge resource**

<table>
<thead>
<tr>
<th></th>
<th>Neurosurgeon (ASPLOS’17)</th>
<th>DDNN (ICDCS’17)</th>
<th>Edge-host partitioning of DNN (AVSS’18)</th>
<th>Couper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edge involved?</td>
<td></td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Generic slicing method?</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Verified by production DNN?</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Supporting different tenancies?</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Couper Introduction

DNN Model

Video Analytics Application

Resource specification

Couper

Container image on edge

Container image on cloud
Couper Introduction

DNN Model

Video Analytics Application

Resource specification

Couper

Container image on edge

Container image on cloud

Camera processor

ML inference processor 1st part of DNN

ML inference processor 2nd(rest) part of DNN

The best slicing deployment
Goals:
- How Couper **improves performance**?
- How Couper **reduces problem space** and saves evaluation time?
- Why Couper **supports different evaluating methods**?

**Hardware specification of experiments:**

<table>
<thead>
<tr>
<th>Device</th>
<th>CPU Freq (GHz)</th>
<th>CPU proc</th>
<th>RAM (GB)</th>
<th>GPU</th>
<th>RTT (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client device</td>
<td>2.0</td>
<td>2</td>
<td>1</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Low-end edge</td>
<td>2.0</td>
<td>4</td>
<td>16</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Mid-end edge</td>
<td>3.1</td>
<td>8</td>
<td>32</td>
<td>N/A</td>
<td>15</td>
</tr>
<tr>
<td>High-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td>N/A</td>
<td>25</td>
</tr>
<tr>
<td>Cloud server</td>
<td>3.1</td>
<td>48</td>
<td>96</td>
<td>2 Nvidia P100</td>
<td></td>
</tr>
</tbody>
</table>
Experiment

Goals:
- How Couper **improves performance**?
- How Couper **reduces problem space** and saves evaluation time?
- Why Couper **supports different evaluating methods**?

Hardware specification of experiments:

<table>
<thead>
<tr>
<th>Device</th>
<th>CPU Freq (GHz)</th>
<th>CPU proc</th>
<th>RAM (GB)</th>
<th>GPU</th>
<th>RTT (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>client</td>
</tr>
<tr>
<td>Client device</td>
<td>2.0</td>
<td>2</td>
<td>1</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Low-end edge</td>
<td>2.0</td>
<td>4</td>
<td>16</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Mid-end edge</td>
<td>3.1</td>
<td>8</td>
<td>32</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>High-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td>2 Nvidia P100</td>
<td></td>
</tr>
<tr>
<td>Cloud server</td>
<td>3.1</td>
<td>48</td>
<td>96</td>
<td>2 Nvidia P100</td>
<td></td>
</tr>
</tbody>
</table>

More powerful edge is further from client
## Experiment

The original layers of DNN and the # evaluation candidates

<table>
<thead>
<tr>
<th>Model</th>
<th># Layer</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Strongman</td>
</tr>
<tr>
<td>VGG 16</td>
<td>54</td>
<td>52</td>
</tr>
<tr>
<td>MobileNet V2 1.4</td>
<td>158</td>
<td>155</td>
</tr>
<tr>
<td>ResNet V2 50</td>
<td>205</td>
<td>34</td>
</tr>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>34</td>
</tr>
<tr>
<td>Inception ResNet V2</td>
<td>871</td>
<td>106</td>
</tr>
<tr>
<td>NASNet 331</td>
<td>1265</td>
<td>7</td>
</tr>
<tr>
<td>PNASNet 331</td>
<td>939</td>
<td>7</td>
</tr>
</tbody>
</table>
## Experiment

The original layers of DNN and the # evaluation candidates

<table>
<thead>
<tr>
<th>Model</th>
<th># Layer</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Strongman</td>
</tr>
<tr>
<td>VGG 16</td>
<td>54</td>
<td>52</td>
</tr>
<tr>
<td>MobileNet V2 1.4</td>
<td>158</td>
<td>155</td>
</tr>
<tr>
<td>ResNet V2 50</td>
<td>205</td>
<td>34</td>
</tr>
<tr>
<td>Inception V3</td>
<td>788</td>
<td>34</td>
</tr>
<tr>
<td>Inception ResNet V2</td>
<td>871</td>
<td>106</td>
</tr>
<tr>
<td>NASNet 331</td>
<td>1265</td>
<td>7</td>
</tr>
<tr>
<td>PNASNet 331</td>
<td>939</td>
<td>7</td>
</tr>
</tbody>
</table>

Up to 98% evaluation time reduction

Hugely reduce problem space(split point candidates) by methods
Next Step

• **Couper Enhancement:**
  Working with different DNN model, application, and framework (i.e. Yolov3 with object detection)

• **Collaborate with edge software stack:**
  Evaluating 5G environment, edge infrastructure (i.e. Akraino), and supporting software (i.e. NFV techniques)

• **Multi-tenancy with different workloads:**
  Evaluating on the compute and network interference/overhead while sharing resource with other services
Edge resources are diverse and target to support multi-tenancy.

- Linux Foundation Edge, Akraino — emerging technology and edge coverage
Edge resources are diverse and target to support multi-tenancy.

- Linux Foundation Edge, Akraino – emerging technology and edge coverage.
Edge resources are diverse and target to support multi-tenancy.

- Linux Foundation Edge, Akraino – emerging technology and edge coverage
Edge resources are diverse and target to support multi-tenancy

Even in specific edge device owned by certain company, need to support multiple services

- Chick-fil-A, Edge computing architecture overview
Edge resources are diverse and target to support multi-tenancy

Even in specific edge device owned by certain company, need to support multiple services

- Chick-fil-A, Edge computing architecture overview
Couper Introduction

Hardware Spec

Container images

Method Adaptor

Method

Next iteration

Profiler

kubeconfig Creator

Stop verification

Referee

The best split point
Experiment

Goals:
- How Couper **improves performance**?
- How Couper **reduces problem space** and saves evaluation time?
- Why Couper **supports different evaluating methods**?

Hardware specification of experiments:

<table>
<thead>
<tr>
<th>Device</th>
<th>CPU Freq (GHz)</th>
<th>CPU proc</th>
<th>RAM (GB)</th>
<th>GPU</th>
<th>RTT (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>client</td>
</tr>
<tr>
<td>Client device</td>
<td>2.0</td>
<td>2</td>
<td>1</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Low-end edge</td>
<td>2.0</td>
<td>4</td>
<td>16</td>
<td>1 Nvidia P100</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>cloud</td>
</tr>
<tr>
<td>Mid-end edge</td>
<td>3.1</td>
<td>8</td>
<td>32</td>
<td>N/A</td>
<td>15</td>
</tr>
<tr>
<td>High-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td>1 Nvidia P100</td>
<td>25</td>
</tr>
<tr>
<td>Super-high-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td>1 Nvidia P100</td>
<td>25</td>
</tr>
<tr>
<td>Cloud server</td>
<td>3.1</td>
<td>48</td>
<td>96</td>
<td>2 Nvidia P100</td>
<td>42</td>
</tr>
</tbody>
</table>
Experiment

Goals:
- How Couper improves performance?
- How Couper reduces problem space and saves evaluation time?
- Why Couper supports different evaluating methods?

Hardware specification of experiments:

<table>
<thead>
<tr>
<th>Device</th>
<th>CPU Freq (GHz)</th>
<th>CPU proc</th>
<th>RAM (GB)</th>
<th>GPU</th>
<th>RTT (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>client</td>
</tr>
<tr>
<td>Client device</td>
<td>2.0</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low-end edge</td>
<td>2.0</td>
<td>4</td>
<td>16</td>
<td>N/A</td>
<td>1</td>
</tr>
<tr>
<td>Mid-end edge</td>
<td>3.1</td>
<td>8</td>
<td>32</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>High-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>Super-high-end edge</td>
<td>3.1</td>
<td>16</td>
<td>64</td>
<td>1 Nvidia P100</td>
<td>25</td>
</tr>
<tr>
<td>Cloud server</td>
<td>3.1</td>
<td>48</td>
<td>96</td>
<td>2 Nvidia P100</td>
<td></td>
</tr>
</tbody>
</table>

More powerful edge is further from client
Real evaluation time in minutes across models and edge devices, the hybrid method comes out decision more faster than strongman.

<table>
<thead>
<tr>
<th>Model</th>
<th>Inception V3</th>
<th>Inception ResNet V2</th>
<th>PNASNet 331</th>
</tr>
</thead>
<tbody>
<tr>
<td>The evaluation time of Strongman</td>
<td>&gt; 30</td>
<td>≈ 120</td>
<td>≈ 10</td>
</tr>
<tr>
<td>Low-end edge</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Mid-end edge</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>High-end edge</td>
<td>10</td>
<td>16</td>
<td>1</td>
</tr>
</tbody>
</table>