Adapting Swarm Application
A Systematic and Quantitative Approach

Ben Zhang
July 6, 2018

TerraSwarm Research Center
https://github.com/nebgnahz/dissertation-talk
# Table of contents

1. Introduction to the Swarm

2. Network Resource Adaptation

3. Compute Resource Adaptation

4. Conclusion and Acknowledgement
Introduction to the Swarm
The Swarm at the Edge of the Cloud

J. Rabaey, ASPDAC’08
Swarm, or

- Internet of Things (IoT)
- Internet of Everything (IoE)
- Industry 4.0
- The Industrial Internet
- TSensors (Trillion Sensors)
- Machine to Machine (M2M)
- Smarter Planet

J. Rabaey, ASPDAC’08
The Swarm at the Edge of the Cloud

Swarm, or
- Internet of Things (IoT)
- Internet of Everything (IoE)
- Industry 4.0
- The Industrial Internet
- TSensors (Trillion Sensors)
- Machine to Machine (M2M)
- Smarter Planet

J. Rabaey, ASPDAC’08
The Swarm at the Edge of the Cloud

Swarm, or

- Internet of Things (IoT)
- Internet of Everything (IoE)
- Industry 4.0
- The Industrial Internet
- TSensors (Trillion Sensors)
- Machine to Machine (M2M)
- Smarter Planet

J. Rabaey, ASPDAC’08
A Cloud-centric Approach

(a) Electric Imp

(b) Samsung SAMI

(c) Ninja Sphere
“The Cloud”: Model vs. Reality
“The Cloud”: Model vs. Reality
## The Cloud is Not Enough

<table>
<thead>
<tr>
<th></th>
<th>Web/IT</th>
<th>Swarm/IoT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy &amp; Security</td>
<td>Open for access</td>
<td>Sensitive data</td>
</tr>
<tr>
<td>Scalability</td>
<td>Power law</td>
<td>Billion devices</td>
</tr>
<tr>
<td>Interaction Model</td>
<td>Human</td>
<td>Machine</td>
</tr>
<tr>
<td>Latency</td>
<td>Variable</td>
<td>Bounded</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Downstream</td>
<td>Upstream</td>
</tr>
<tr>
<td>Availability (QoS)</td>
<td>No guarantee</td>
<td>Requirement</td>
</tr>
<tr>
<td>Durability Management</td>
<td>Cloud controls</td>
<td>Users control</td>
</tr>
</tbody>
</table>

Pitfalls with Today’s Approach to IoT [Zhang et al., 2015]
<table>
<thead>
<tr>
<th></th>
<th>Web/IT</th>
<th>Swarm/IoT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy &amp; Security</td>
<td>Open for access</td>
<td>Sensitive data</td>
</tr>
<tr>
<td>Scalability</td>
<td>Power law</td>
<td>Billion devices</td>
</tr>
<tr>
<td>Interaction Model</td>
<td>Human</td>
<td>Machine</td>
</tr>
<tr>
<td>Latency</td>
<td>Variable</td>
<td>Bounded</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>Downstream</td>
<td>Upstream</td>
</tr>
<tr>
<td>Availability (QoS)</td>
<td>No guarantee</td>
<td>Requirement</td>
</tr>
<tr>
<td>Durability Management</td>
<td>Cloud controls</td>
<td>Users control</td>
</tr>
</tbody>
</table>

Pitfalls with Today’s Approach to IoT [Zhang et al., 2015]
Bandwidth: Downstream vs. Upstream

Web/IT

Users

Swarm/IoT

Users
Network Resource Adaptation
Limited Network Resource

<table>
<thead>
<tr>
<th>Demand</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge Data Volume at the Edge</td>
<td>Insufficient WAN Bandwidth</td>
</tr>
</tbody>
</table>

What about edge processing? (I will cover in the second half of this talk).

But communication is not avoidable.

- Large performance gap between the cloud and the edge (GPU/TPU/ASIC).
- Aggregation is sometimes necessary in applications.
- Last-hop wireless may become the bottleneck.
### Limited Network Resource

<table>
<thead>
<tr>
<th>Demand</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge Data Volume at the Edge</td>
<td>Insufficient WAN Bandwidth</td>
</tr>
</tbody>
</table>

- Video surveillance, 3 mbps per camera [Amerasinghe, 2009]
- Electrical grid monitoring, 1.4 million data points per second [Andersen and Culler, 2016]
Limited Network Resource

<table>
<thead>
<tr>
<th>Demand</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge Data Volume at the Edge</td>
<td>Insufficient WAN Bandwidth</td>
</tr>
</tbody>
</table>

... *Dropcam*, a WiFi video-streaming camera and associated cloud backend service for storing and watching the resulting video. *Dropcam has the fewest clients* (2,940) .... Yet, each client uses roughly *2.8 GB* a week and uploads *nearly 19 times more* than they download, implying that *Dropcam users do not often watch what they record.*

Large-scale Measurements of Wireless Network Behavior

[Biswas et al., 2015]
Limited Network Resource

<table>
<thead>
<tr>
<th>Demand</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge Data Volume at the Edge</td>
<td>Insufficient WAN Bandwidth</td>
</tr>
</tbody>
</table>

Bandwidth variations throughout the day between Amazon EC2 sites. Similar scarcity and variation for wireless networks, broadband access networks and cellular networks (backup slides).

What about edge processing? (I will cover in the second half of this talk).

But communication is not avoidable.

- Large performance gap between the cloud and the edge (GPU/TPU/ASIC).
- Aggregation is sometimes necessary in applications.
- Last-hop wireless may become the bottleneck.
## Limited Network Resource

<table>
<thead>
<tr>
<th>Demand</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Huge Data Volume at the Edge</td>
<td>Insufficient WAN Bandwidth</td>
</tr>
</tbody>
</table>

What about edge processing? (I will cover in the second half of this talk).

But communication is not avoidable.

- Large performance gap between the cloud and the edge (GPU/TPU/ASIC).
- Aggregation is sometimes necessary in applications.
- Last-hop wireless may become the bottleneck.
When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
  - JetStream [Rabkin et al., 2014] uses manual policy
  - “if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if still not enough”
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
- Application-specific optimizations don’t generalize
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
- Application-specific optimizations don't generalize
  - Video streaming often aims at Quality of Experience (limited degradation dimension, e.g. maintain 25FPS)
  - For object detection, resolution matters more than FPS
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
- Application-specific optimizations don’t generalize
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
- Application-specific optimizations don’t generalize
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
- Application-specific optimizations don’t generalize
Application-specific Optimizations Don’t Generalize

$t=0s$, small target in far-field views

$t=1s$, small difference
Positive if intersection over union (IOU) larger than 0.5.

\[
\text{IOU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}
\]

- (a) \(\text{IOU} = 0.14\)
- (b) \(\text{IOU} = 0.57\)
- (c) \(\text{IOU} = 0.82\)
Application-specific Optimizations Don’t Generalize

F1 score is the harmonic mean of precision and recall, **ranging from 0 to 1**:

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>True Positive</td>
<td>False Positive</td>
</tr>
<tr>
<td>N</td>
<td>False Negative</td>
<td>True Negative</td>
</tr>
</tbody>
</table>

\[
\text{Precision} = \frac{\text{true positive}}{\text{all positive}}
\]

\[
\text{Recall} = \frac{\text{true positive}}{\text{all detection}}
\]

\[
\text{F1} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}
\]

`t=0s, small target in far-field views`

`t=1s, small difference`
Application-specific Optimizations Don’t Generalize

$t=0s$, small target in far-field views

$t=1s$, small difference
Application-specific Optimizations Don’t Generalize

- $t=0s$, small target in far-field views
- $t=1s$, small difference

![Chart showing Bandwidth (normalized) vs. Frame Rate and Accuracy]
Application-specific Optimizations Don’t Generalize

$t=0s$, small target in far-field views

$t=1s$, small difference
Application-specific Optimizations Don’t Generalize

t=0s, small target in far-field views

t=1s, small difference
Application-specific Optimizations Don’t Generalize

t=0s, nearby and large targets

t=1s, large difference
Making Adaptation Practical is Challenging

<table>
<thead>
<tr>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimize bandwidth while maximizing application accuracy</td>
</tr>
</tbody>
</table>
Making Adaptation Practical is Challenging

Goal
Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.
Goal

Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.

2. It requires expertise and manual work to explore multidimensional adaptation.
Goal

Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.

2. It requires expertise and manual work to explore multidimensional adaptation.

3. The adaptation happens at the runtime.
Making Adaptation Practical is Challenging

Goal
Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.
   - APIs: maybe operators to express adaptation.

2. It requires expertise and manual work to explore multidimensional adaptation.

3. The adaptation happens at the runtime.
Making Adaptation Practical is Challenging

Goal
Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.
   - APIs: maybe operators to express adaptation.
2. It requires expertise and manual work to explore multidimensional adaptation.
   - Profiling: automatically learn Pareto-optimal strategy with multi-dimensional exploration.
3. The adaptation happens at the runtime.
Goal
Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don’t generalize.
   - APIs: maybe operators to express adaptation.

2. It requires expertise and manual work to explore multidimensional adaptation.
   - Profiling: automatically learn Pareto-optimal strategy with multi-dimensional exploration.

3. The adaptation happens at the runtime.
   - Engineering an adaptation system to balance latency and accuracy.
Goal
Minimize bandwidth while maximizing application accuracy
(1) Stream Processing APIs

Data Stream → Operator → Data Stream
(1) Stream Processing APIs

![Diagram of Stream Processing]

- **Data Stream** → **Operator** → **Data Stream**
- $\{x_1, x_2, x_3, x_4, \ldots \}$ → **map($f$)** → $\{f(x_1), f(x_2), f(x_3), f(x_4), \ldots \}$
- $\{x_1, x_2, x_3, x_4, \ldots \}$ → **window(2, f)** → $\{f(x_1, x_2), f(x_3, x_4), \ldots \}$
(1) Stream Processing APIs

![Diagram of stream processing]

- **Data Stream** → **Operator** → **Data Stream**

  - {\(x_1, x_2, x_3, x_4, \ldots\)} → **map**(\(f\)) → {\(f(x_1), f(x_2), f(x_3), f(x_4), \ldots\)}

  - {\(x_1, x_2, x_3, x_4, \ldots\)} → **window**(2, \(f\)) → {\(f(x_1, x_2), f(x_3, x_4), \ldots\)}

### Normal

- **map** (\(f: I \Rightarrow O\)) → **Stream**\(<I>\) → **Stream**\(<O>\)
- **skip** (\(i: Integer\)) → **Stream**\(<I>\) → **Stream**\(<I>\)
- **window** (count: Integer, \(f: Vec\(<I>\) \Rightarrow O\)) → **Stream**\(<I>\) → **Stream**\(<O>\)

... ...
(1) Stream Processing APIs

**Data Stream** → **Operator** → **Data Stream**

- **map** \( f \) : \( \{x_1, x_2, x_3, x_4, \ldots \} \) → \( \{f(x_1), f(x_2), f(x_3), f(x_4), \ldots \} \)
- **window** \( (2, f) \) : \( \{x_1, x_2, x_3, x_4, \ldots \} \) → \( \{f(x_1, x_2), f(x_3, x_4), \ldots \} \)
- **maybe** \( (k, f) \) : \( \{x_1, x_2, x_3, x_4, \ldots \} \) → \( \{f(x_1, k_i_1), f(x_2, k_i_2), f(x_3, k_i_3), f(x_4, k_i_4), \ldots \} \)

<table>
<thead>
<tr>
<th>Normal</th>
<th><strong>map</strong> ( f : I \Rightarrow O )</th>
<th>Stream( &lt;I&gt; ) → Stream( &lt;O&gt; )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>skip</strong> ( i : \text{Integer} )</td>
<td>Stream( &lt;I&gt; ) → Stream( &lt;I&gt; )</td>
</tr>
<tr>
<td></td>
<td><strong>window</strong> ( \text{count} : \text{Integer}, f : \text{Vec}&lt;I&gt; \Rightarrow O )</td>
<td>Stream( &lt;I&gt; ) → Stream( &lt;O&gt; )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

## 1. Stream Processing APIs

### Normal

- **map** ($f: I \Rightarrow O$)
  - $\text{Stream}<I> \Rightarrow \text{Stream}<O>$
- **skip** ($i: \text{Integer}$)
  - $\text{Stream}<I> \Rightarrow \text{Stream}<I>$
- **window** (count: Integer, $f: \text{Vec}<I> \Rightarrow O$)
  - $\text{Stream}<I> \Rightarrow \text{Stream}<O>$
  - ...
  - ...

### Adaptation

- **maybe** (knobs: Vec<T>, $f: (T, I) \Rightarrow I$)
  - $\text{Stream}<I> \Rightarrow \text{Stream}<I>$
- **maybe_skip** (knobs: Vec<Integer>)
  - $\text{Stream}<I> \Rightarrow \text{Stream}<I>$
- **maybe_head** (knobs: Vec<Integer>)
  - $\text{Stream}<\text{Vec}<I>> \Rightarrow \text{Stream}<\text{Vec}<I>>$
  - ...
  - ...

---

**Diagram:**

![Diagram of Stream Processing APIs](image-url)
```rust
maybe(knobs: Vec<T>, f: (T, I) => I)

let quantized_stream = vec![1, 2, 3, 4].into_stream()
    .maybe(vec![2, 4], |k, val| val / k)
    .collect();

[1, 2, 3, 4]

no degradation

k = 2

[0, 1, 1, 2]

k = 4

[0, 0, 0, 1]
```

We rewrite the video streaming application as follows,
```rust
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![(1600, 900), (1280, 720)])
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();
```

Example code in Rust, simplified for presentation.
maybe(knobs: Vec<T>, f: (T, I) => I)

```rust
let quantized_stream = vec![1, 2, 3, 4].into_stream()
    .maybe(vec![2, 4], |k, val| val / k)
    .collect();
```

We rewrite the video streaming application as follows,

```rust
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![1600, 900], (1280, 720))
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();
```

Example code in Rust, simplified for presentation.
let app = Camera::new((1920, 1080), 30)
  .maybe_downsample(vec![1600, 900], (1280, 720))
  .maybe_skip(vec![2, 5])
  .map(|frame| pedestrian_detect(frame))
  .compose();
let app = Camera::new((1920, 1080), 30)
  .maybe_downsample(vec![1600, 900, 1280, 720])
  .maybe_skip(vec![2, 5])
  .map(|frame| pedestrian_detect(frame))
  .compose();
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![1600, 900], 1280, 720)
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();

<table>
<thead>
<tr>
<th>Training Data</th>
<th>Accuracy Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1920, 1080)</td>
<td>0 10.7 1.0</td>
</tr>
<tr>
<td>(1600, 900)</td>
<td>0 8.3 0.88</td>
</tr>
<tr>
<td>(1280, 720)</td>
<td>0 6.3 0.87</td>
</tr>
<tr>
<td>(1920, 1080)</td>
<td>2 9.3 0.90</td>
</tr>
</tbody>
</table>
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![(1600, 900), (1280, 720)])
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();

<table>
<thead>
<tr>
<th>downsample</th>
<th>skip</th>
<th>bandwidth</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1920, 1080)</td>
<td>0</td>
<td>10.7</td>
<td>1.0</td>
</tr>
<tr>
<td>(1600, 900)</td>
<td>0</td>
<td>8.3</td>
<td>0.88</td>
</tr>
<tr>
<td>(1280, 720)</td>
<td>0</td>
<td>6.3</td>
<td>0.87</td>
</tr>
<tr>
<td>(1920, 1080)</td>
<td>2</td>
<td>9.3</td>
<td>0.90</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Profile: Pareto-optimal Strategy

Symbol Description

- $n$: number of degradation operations
- $k_i$: the $i$-th degradation knob
- $c = [k_1, k_2, \ldots k_n]$: one specific configuration
- $\mathcal{C}$: the set of all configurations
- $B(c)$: bandwidth requirement for $c$
- $A(c)$: accuracy measure for $c$
- $\mathbb{P}$: Pareto-optimal set
Profile: Pareto-optimal Strategy

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of degradation operations</td>
</tr>
<tr>
<td>$k_i$</td>
<td>the $i$-th degradation knob</td>
</tr>
<tr>
<td>$c = [k_1, k_2, \ldots k_n]$</td>
<td>one specific configuration</td>
</tr>
<tr>
<td>$C$</td>
<td>the set of all configurations</td>
</tr>
</tbody>
</table>

- $B(c)$ bandwidth requirement for $c$
- $A(c)$ accuracy measure for $c$
- $\mathbb{P}$ Pareto-optimal set

See red markers in the figure.
Profile: Pareto-optimal Strategy

Symbol | Description
---|---
\(n\) | number of degradation operations
\(k_i\) | the \(i\)-th degradation knob
\(c = [k_1, k_2, \ldots, k_n]\) | one specific configuration
\(\mathcal{C}\) | the set of all configurations

\(B(c)\) | bandwidth requirement for \(c\)
\(A(c)\) | accuracy measure for \(c\)
\(\mathbb{P}\) | Pareto-optimal set
Profile: Pareto-optimal Strategy

### Symbol Description

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>number of degradation operations</td>
</tr>
<tr>
<td>( k_i )</td>
<td>the ( i )-th degradation knob</td>
</tr>
<tr>
<td>( c = [k_1, k_2, \ldots, k_n] )</td>
<td>one specific configuration</td>
</tr>
<tr>
<td>( C )</td>
<td>the set of all configurations</td>
</tr>
</tbody>
</table>

\[ B(c) \] bandwidth requirement for \( c \)

\[ A(c) \] accuracy measure for \( c \)

\( \mathbb{P} \) Pareto-optimal set

---

See red markers in the figure.
Profile: Pareto-optimal Strategy

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of degradation operations</td>
</tr>
<tr>
<td>$k_i$</td>
<td>the $i$-th degradation knob</td>
</tr>
<tr>
<td>$c = [k_1, k_2, \ldots k_n]$</td>
<td>one specific configuration</td>
</tr>
<tr>
<td>$\mathcal{C}$</td>
<td>the set of all configurations</td>
</tr>
<tr>
<td>$B(c)$</td>
<td>bandwidth requirement for $c$</td>
</tr>
<tr>
<td>$A(c)$</td>
<td>accuracy measure for $c$</td>
</tr>
<tr>
<td>$\mathbb{P}$</td>
<td>Pareto-optimal set</td>
</tr>
</tbody>
</table>

See red markers in the figure.
Profile: Pareto-optimal Strategy

- **Symbol**
  - $n$: number of degradation operations
  - $k_i$: the $i$-th degradation knob
  - $c = [k_1, k_2, \ldots k_n]$: one specific configuration
  - $\mathcal{C}$: the set of all configurations
  - $B(c)$: bandwidth requirement for $c$
  - $A(c)$: accuracy measure for $c$
  - $\mathbb{P}$: Pareto-optimal set

- **Description**
  - number of degradation operations
  - the $i$-th degradation knob
  - one specific configuration
  - the set of all configurations
  - bandwidth requirement for $c$
  - accuracy measure for $c$
  - Pareto-optimal set

---

[Graph showing a scatter plot with marked points on the x-axis and y-axis labeled as Accuracy and Bandwidth (normalize).]
# Profile: Pareto-optimal Strategy

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of degradation operations</td>
</tr>
<tr>
<td>$k_i$</td>
<td>the $i$-th degradation knob</td>
</tr>
<tr>
<td>$c = [k_1, k_2, ... k_n]$</td>
<td>one specific configuration</td>
</tr>
<tr>
<td>$\mathcal{C}$</td>
<td>the set of all configurations</td>
</tr>
<tr>
<td>$B(c)$</td>
<td>bandwidth requirement for $c$</td>
</tr>
<tr>
<td>$A(c)$</td>
<td>accuracy measure for $c$</td>
</tr>
<tr>
<td>$\mathbb{P}$</td>
<td>Pareto-optimal set</td>
</tr>
</tbody>
</table>

---

![Graph showing Accuracy vs. Bandwidth (normalize)](image)
Profile: Pareto-optimal Strategy

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>number of degradation operations</td>
</tr>
<tr>
<td>$k_i$</td>
<td>the $i$-th degradation knob</td>
</tr>
<tr>
<td>$c = [k_1, k_2, \ldots k_n]$</td>
<td>one specific configuration</td>
</tr>
<tr>
<td>$\mathbb{C}$</td>
<td>the set of all configurations</td>
</tr>
<tr>
<td>$B(c)$</td>
<td>bandwidth requirement for $c$</td>
</tr>
<tr>
<td>$A(c)$</td>
<td>accuracy measure for $c$</td>
</tr>
<tr>
<td>$\mathbb{P}$</td>
<td>Pareto-optimal set</td>
</tr>
</tbody>
</table>

\[
\mathbb{P} = \{ c \in \mathbb{C} : \{ c' \in \mathbb{C} : B(c') < B(c), A(c') > A(c) \} = \emptyset \} \]

See red markers in the figure.
(3) Runtime Adaptation

Adaptation Controller State Machine. We introduce the Probe phase to conservatively change the adaptation level. For details, please see the paper/thesis.
(3) Runtime Adaptation

Adaptation Controller State Machine. We introduce the Probe phase to conservatively change the adaptation level. For details, please see the paper/thesis.
(3) Runtime Adaptation

Adaptation Controller State Machine. We introduce Probe phase to conservatively change adaptation level. For details, please see the paper/thesis.
## Applications

<table>
<thead>
<tr>
<th>Application</th>
<th>Knobs</th>
<th>Accuracy</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Augmented Reality</td>
<td>resolution, frame rate, quantization</td>
<td>F1 score</td>
<td>iPhone video clips</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[Rijsbergen, 1979]</td>
<td>training: office (24s) testing: home (246s)</td>
</tr>
<tr>
<td>Pedestrian Detection</td>
<td>resolution, frame rate, quantization</td>
<td>F1 score</td>
<td>MOT16 [Milan et al., 2016]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>training: MOT16-04 testing: MOT16-03</td>
</tr>
<tr>
<td>Log Analysis (Top-K, K=50)</td>
<td>head (N), threshold (T)</td>
<td>Kendall’s $\tau$</td>
<td>SEC.gov logs [DERA, 2016]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[Abdi, 2007]</td>
<td>training: 4 days testing: 16 days</td>
</tr>
</tbody>
</table>
A distributed Top-K application with two degradation operations: head and threshold. In this example, f2, which is not in Top-1 for either client, becomes the global Top-1 after the merge. It would have been purged if the clients use threshold $T=3$, demonstrating degradation that reduces data sizes affects fidelity.
Evaluation: Generated Profiles

(a) Augmented Reality (AR)

(b) Pedestrian Detection (PD)

Optimal strategy is achieved with multiple dimensions; tuning one dimension leads to suboptimal performance.

For the same application, different dimensions have different impact.

For different applications, the same dimension has different impact.
Evaluation: Generated Profiles

(a) Augmented Reality (AR)

(b) Pedestrian Detection (PD)

- Optimal strategy is achieved with multiple dimensions; tuning one dimension leads to suboptimal performance.
Evaluation: Generated Profiles

(a) Augmented Reality (AR)

(b) Pedestrian Detection (PD)

- Optimal strategy is achieved with multiple dimensions; tuning one dimension leads to suboptimal performance.
- For the same application, different dimensions have different impact.
Optimal strategy is achieved with multiple dimensions; tuning one dimension leads to suboptimal performance.

- For the same application, different dimensions have different impact.
- For different applications, the same dimension has different impact.
The effect of each dimension is not significantly different.
The profile offers quantified effects of degradation.
The effect of each dimension is not significantly different.
The effect of each dimension is not significantly different.

The profile offers quantified effects of degradation.
<table>
<thead>
<tr>
<th>Baseline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streaming over TCP</td>
<td>A non-adaptive approach</td>
</tr>
<tr>
<td>Streaming over UDP</td>
<td>A non-adaptive approach, represents RTP/UDP/P/RTSP video streaming</td>
</tr>
</tbody>
</table>
## Evaluation: Runtime Experiment Baselines

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streaming over TCP</td>
<td>A non-adaptive approach</td>
</tr>
<tr>
<td>Streaming over UDP</td>
<td>A non-adaptive approach, represents RTP/UDP/RTSP video streaming</td>
</tr>
<tr>
<td>JetStream [Rabkin et al., 2014]</td>
<td>Manual Policy: “if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if there still isn’t enough bandwidth. Beyond that point, reduce the frame rate, but keep the image fidelity.”</td>
</tr>
<tr>
<td>Baseline</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Streaming over TCP</td>
<td>A non-adaptive approach</td>
</tr>
<tr>
<td>Streaming over UDP</td>
<td>A non-adaptive approach, represents RTP/UDP/P/RTSP video streaming</td>
</tr>
<tr>
<td>JetStream [Rabkin et al., 2014]</td>
<td>Manual Policy: “if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if there still isn’t enough bandwidth. Beyond that point, reduce the frame rate, but keep the image fidelity.”</td>
</tr>
<tr>
<td>JetStream++</td>
<td>Uses adaptation policy generated by AWStream. JetStream runtime does not probe (hence may oscillate between policies).</td>
</tr>
</tbody>
</table>
## Evaluation: Runtime Experiment Baselines

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Streaming over TCP</td>
<td>A non-adaptive approach</td>
</tr>
<tr>
<td>Streaming over UDP</td>
<td>A non-adaptive approach, represents RTP/UDP/RTSP video streaming</td>
</tr>
<tr>
<td>JetStream</td>
<td>Manual Policy: &quot;if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if there still isn’t enough bandwidth. Beyond that point, reduce the frame rate, but keep the image fidelity.&quot;</td>
</tr>
<tr>
<td>JetStream++</td>
<td>Uses adaptation policy generated by AWStream. JetStream runtime does not probe (hence may oscillate between policies).</td>
</tr>
<tr>
<td>HLS</td>
<td>HTTP Live Streaming represents popular adaptive video streaming techniques; used for Periscope video stream [Wang et al., 2016].</td>
</tr>
</tbody>
</table>
HTTP Live Streaming (HLS) architecture: designed for live video viewing and relying on buffering at the viewing side.
Evaluation: Runtime Performance

- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

**Throughput (mbps)**

- Time (seconds)

**Latency (seconds)**

- Time (seconds)

**Accuracy (F1 Score)**

- Time (seconds)
Evaluation: Runtime Performance

- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

Throughput (mbps):

Latency (seconds):

Accuracy (F1 Score):

Time (seconds):

Graphs showing the performance metrics over time for different stream systems and protocols.
Evaluation: Runtime Performance

- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

Throughput (Mbps)

Latency (seconds)

Accuracy (F1 Score)
Evaluation: Runtime Performance

- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

Throughput (Mbps)

Latency (seconds)

Accuracy (F1 Score)

Time (seconds)
Evaluation: Runtime Performance

Throughput (mbps):
- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

Latency (seconds):
- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP

Accuracy (F1 Score):
- AWStream
- JetStream++
- JetStream
- HLS
- Streaming over TCP
- Streaming over UDP
Evaluation: Runtime Performance

- Latency (ms)
- Accuracy (F1 score)

- Streaming over UDP
- Streaming over TCP
- HLS
- JetStream
- JetStream++
- AWStream

- Manual Policies
- App-specific

- Better

- Freshness, latency reversed (seconds)
- Fidelity, accuracy (%)

23/37
Evaluation: Runtime Performance

- Latency (ms)
  - AWStream
  - JetStream++
  - JetStream
  - HLS
  - Streaming over TCP
  - Streaming over UDP

- Accuracy (F1 score)

- Streaming over UDP
- Streaming over TCP
- HLS
- JetStream
- JetStream++
- AWStream

- Manual Policies
  - Better

- Fidelity, accuracy (%)
  - Streaming over TCP
  - Manual Policies
  - App-specific
  - Streaming over UDP

- Freshness, latency reversed (seconds)
Compute Resource Adaptation
Edge Computing: Fog/Cloudlet/Swarmbox & New Infrastructure
Edge Computing: Fog/Cloudlet/Swarmbox & New Infrastructure

Cisco Fog Computing [Bonomi et al., 2012]

Cloudlet [Satyanarayanan et al., 2009]
Edge Computing: Fog/Cloudlet/Swarmbox & New Infrastructure

Cisco Fog Computing [Bonomi et al., 2012]

Cloudlet [Satyanarayanan et al., 2009]

Philips Hue Hub  SmartThings  Smartphones  Google onHub  SwarmBox

Intel NUC

Many Gateways
Heterogeneous Environment

Characteristics of IoT/mobile, edge and cloud:

- **IoT / Mobile Devices**
  - Limited Resources
  - Less workload
  - Lower latency
  - Compute Power

- **Edge Computers (Cloudlet)**
  - More Available Resources
  - More workload
  - Higher latency
  - Workload

- **Cloud**
  - More available resources
  - Less resource guarantee
  - Less stable connections
  - Latency

In summary, IoT/mobile devices, edge computers, and cloud each have distinct characteristics in terms of resources, workload, latency, and reliability.
Heterogeneous Environment

IoT / Mobile Devices

Edge Computers (Cloudlet)

Cloud

More Available Resources

Limited Resources

Less workload

More workload

Less resource guarantee

Higher latency

Less stable connections

Latency

Lower latency

<table>
<thead>
<tr>
<th>Platform</th>
<th>Processing times (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPi Model B</td>
<td>4105</td>
</tr>
<tr>
<td>Macbook Model A1502</td>
<td>544</td>
</tr>
<tr>
<td>Workstation Xeon E5-1620</td>
<td>346</td>
</tr>
</tbody>
</table>

Processing times (ms) on different platforms.
Accuracy and Processing Times Tradeoff

**Adaptation**

Different algorithm and parameters affect the accuracy and processing times.

Within the tradeoff space, select appropriate algorithm and parameters to meet bounded response time goal.
Accuracy and Processing Times Tradeoff


(b) Benchmarks for Viola Jones face detection when changing different parameters (see explanation on the next slide).
Accuracy and Processing Times Tradeoff

Accuracy and Processing Times Tradeoff


(b) Benchmarks for Viola Jones face detection when changing different parameters (see explanation on the next slide).
The OpenCV implementation of Viola-Jones [Viola and Jones, 2001] has three parameters,

- **scale**: how much the image size is reduced at each image scale.
- **min_size**: minimum detectable object size.
- **min_neighbors**: how many neighbors each candidate rectangle should have to retain it.

Image Source: pyimagesearch.
The OpenCV implementation of Viola-Jones [Viola and Jones, 2001] has three parameters,

- **scale**: how much the image size is reduced at each image scale.
- **min_size**: minimum detectable object size.

Image Source: pyimagesearch.
The OpenCV implementation of Viola-Jones [Viola and Jones, 2001] has three parameters,

- **scale**: how much the image size is reduced at each image scale.
- **min_size**: minimum detectable object size.
- **min_neighbors**: how many neighbors each candidate rectangle should have to retain it.

Image Source: Stack Overflow.
Exhaustive Search is Too Expensive

- **scale**: how much image size is reduced at each image scale.
- **min_size**: minimum detectable object size.
- **min_neighbors**: how many neighbors each candidate rectangle should have to retain it.
pub struct HogParams {
    pub win_size: Size2i,
    pub block_size: Size2i,
    pub block_stride: Size2i,
    pub cell_size: Size2i,
    pub nbins: c_int,
    pub win_sigma: f64,
    pub l2hys_threshold: f64,
    pub gamma_correction: bool,
    pub nlevels: usize,
    pub hit_threshold: f64,
    pub win_stride: Size2i,
    pub padding: Size2i,
    pub scale: f64,
    pub group_threshold: c_int,
    pub use_meanshift_grouping: bool,
    pub final_threshold: f64,
}
Challenges in Adapting Computation

<table>
<thead>
<tr>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapt computation to different platforms</td>
</tr>
</tbody>
</table>
Challenges in Adapting Computation

Goal

Adapt computation to different platforms

Challenges:

1. Large parameter space

2. Heterogeneous capabilities (and not available when profiling)
Challenges in Adapting Computation

Goal
Adapt computation to different platforms

Challenges:

1. Large parameter space
   - Previous approaches use random search or coordinate/greedy approach
   - We propose Bayesian Optimization (BO) for profiling

2. Heterogeneous capabilities (and not available when profiling)
Challenges in Adapting Computation

Goal
Adapt computation to different platforms

Challenges:

1. Large parameter space
   - Previous approaches use random search or coordinate/greedy approach
   - We propose Bayesian Optimization (BO) for profiling

2. Heterogeneous capabilities (and not available when profiling)
   - Profile transfer: refine existing Pareto-optimal points
Bayesian optimization approximate black-box functions with proxy functions and iteratively proposes new sample point in the large parameter space. Effective for,
Bayesian Optimization 101

Bayesian optimization approximate black-box functions with proxy functions and iteratively proposes new sample point in the large parameter space. Effective for,

- Evaluating each sample is expensive.
- The objective is a black-box.
- The evaluation can be noisy.

Gaining attraction beyond ML scope:

- CherryPick [Alipourfard et al., 2017] finds the best cloud configurations for big data analytics.
- Google optimize chocolate chip cookies recipes [Solnik et al., 2017].
Bayesian Optimization 101

Bayesian optimization approximate black-box functions with proxy functions and iteratively proposes new sample point in the large parameter space. Effective for,

- Evaluating each sample is expensive.
- The objective is a black-box.
- The evaluation can be noisy.

Gaining attraction beyond ML scope:

- CherryPick [Alipourfard et al., 2017] finds the best cloud configurations for big data analytics.
- Google optimize chocolate chip cookies recipes [Solnik et al., 2017].
Bayesian Optimization 101

Bayesian optimization approximate black-box functions with proxy functions and iteratively proposes new sample point in the large parameter space. Effective for,

- Evaluating each sample is expensive.
- The objective is a black-box.
- The evaluation can be noisy.

Gaining attraction beyond ML scope:

- CherryPick [Alipourfard et al., 2017] finds the best cloud configurations for big data analytics.
- Google optimize chocolate chip cookies recipes [Solnik et al., 2017].
Acquisition function evaluates the utility of candidate points for the next evaluation of \( f \), balancing a high objective (exploitation) and high uncertainty (exploration) [Shahriari et al., 2016]
Acquisition function evaluates the utility of candidate points for the next evaluation of $f$, balancing a high objective (exploitation) and high uncertainty (exploration) [Shahriari et al., 2016]

For two-objective optimization, utility gain is based on additive-epsilon (top) or hypervolume (bottom) [Binois and Picheny, 2018]
We use PESMO\textsuperscript{2} [Hernández-Lobato et al., 2016] and compare it with two baselines: (1) greedy/coordinate search; (2) random search.

\textsuperscript{2}A Python package based on \textit{Spearmint}. It chooses evaluation points to maximally reduce the entropy of the posterior distribution over the Pareto set.
We use PESMO\(^2\) [Hernández-Lobato et al., 2016] and compare it with two baselines: (1) greedy/coordinate search; (2) random search.

BO evaluates 50 configurations and recommends 29 configurations as the Pareto-optimal boundary (the blue line). Greedy and Random find sub-optimal Pareto configurations with a budget of 80 evaluations (the yellow line in each figure).

\(^2\)A Python package based on Spearmint. It chooses evaluation points to maximally reduce the entropy of the posterior distribution over the Pareto set.
Profile Transfer (without re-running the entire BO)

We make the following observations:
We make the following observations:

- Accuracy remains for a given algorithm/parameter.
We make the following observations:

- Accuracy remains for a given algorithm/parameter.
- Processing time order is preserved
  - More expensive algorithms/parameters remain the same across platforms.
Profile Transfer (without re-running the entire BO)

We make the following observations:

- Accuracy remains for a given algorithm/parameter.
- Processing time order is preserved
  - More expensive algorithms/parameters remain the same across platforms.
- The “Pareto-optimal” is horizontally stretched/compressed.
We make the following observations:

- Accuracy remains for a given algorithm/parameter.
- Processing time order is preserved
  - More expensive algorithms/parameters remain the same across platforms.
- The “Pareto-optimal” is horizontally stretched/compressed.

(Left) Empirically, processing times follows a linear approximation. (Right) Stretched/compressed profile. See paper for details.
Conclusion and Acknowledgement
Summary and Contributions

• Swarm/IoT has huge potentials but also challenges

• Network resource adaptation
  - Addresses scarce and variable WAN bandwidth

• Tradeoff between application accuracy and data size demand

• Compute resource adaptation
  - Addresses heterogeneous platforms and large parameter space

• Tradeoff between application accuracy and processing times

• Overall, a systematic and quantitative approach for adaptation
Summary and Contributions

- Swarm/IoT has huge potentials but also challenges
Summary and Contributions

- Swarm/IoT has huge potentials but also challenges
- Network resource adaptation
  - Addresses scarce and variable WAN bandwidth
  - Tradeoff between application accuracy and data size demand
- Compute resource adaptation
  - Addresses heterogeneous platforms and large parameter space
  - Tradeoff between application accuracy and processing times
- Overall, a systematic and quantitative approach for adaptation
Summary and Contributions

- Swarm/IoT has huge potentials but also challenges
- Network resource adaptation
  - Addresses scarce and variable WAN bandwidth
  - Tradeoff between application accuracy and data size demand
- Compute resource adaptation
  - Addresses heterogeneous platforms and large parameter space
  - Tradeoff between application accuracy and processing times
Summary and Contributions

- Swarm/IoT has huge potentials but also challenges
- Network resource adaptation
  - Addresses scarce and variable WAN bandwidth
  - Tradeoff between application accuracy and data size demand
- Compute resource adaptation
  - Addresses heterogeneous platforms and large parameter space
  - Tradeoff between application accuracy and processing times
- Overall, a systematic and quantitative approach for adaptation
TerraSwarm Vision

TerraSwarm applications are characterized by their ability to dynamically recruit resources such as sensors and data from the cloud, aggregate and use that information to make or aid decisions.
**TerraSwarm Vision**

TerraSwarm applications are characterized by their ability to **dynamically recruit** resources such as sensors and data from the cloud, aggregate and use that information to make or aid decisions.

(a) Accessor in a network of actors.  

(b) Instantiate accessors on another host.
**TerraSwarm Vision**

TerraSwarm applications are characterized by their ability to **dynamically recruit** resources such as sensors and data from the cloud, aggregate and use that information to make or aid decisions.

(a) Accessor in a network of actors.  
(b) Instantiate accessors on another host.

Work in progress with Marten and Andrés. Maybe checkout Marten’s dissertation talk in the future :)
Acknowledgment

Prof. Edward Lee
Prof. John Wawrzynek
Prof. John Chuang
Prof. Sylvia Ratnasamy
Prof. Xin Jin
Prof. John Kubiatowicz

Dr. Ilge Akkaya
Dr. Hokeun Kim
Dr. Chris Shaver
Antonio Iannopollo
Dr. David Mellis

The entire EAL research group (with many visiting scholars)

Kaifei Chen
Prof. John Kubiatowicz
Acknowledgment

Prof. Edward Lee
Prof. John Wawrzynek
Prof. John Chuang
Prof. Sylvia Ratnasamy
Prof. Björn Hartmann
Prof. Xin Jin
Prof. John Kubiatowicz
Dr. Ilge Akkaya
Dr. Hokeun Kim
Dr. Chris Shaver
Antonio Iannopollo
Dr. David Mellis
Kaifei Chen
The entire EAL research group (with many visiting scholars)
Acknowledgement

Edward Lee, John Wawrzynek,
Syliva Ratnasamy, John Chuang,
Björn Hartmann, John Kubiatowicz,
Xiaofan Fred Jiang, Lin Zhang,
David Mellis, Xin Jin,
Mary Stewart, Christopher Brooks,
Eunsuk Kang, Ilge Akkaya,
Hokeun Kim, Marten Lohstroh,
Matt Weber, Antonio Iannopollo,
Mehrdad Niknami, Chris Shaver (Yvan Vivid),
Michael Zimmer, Christos Stergiou,
Dai Bui, Ben Lickly,
Eleftherios Matsikoudis,
Joseph Ng, Chadlia Jerad Ep Ben Haj Hmida,
Moez Ben Haj Hmida,
Maryam Bagheri, Victor Nouvellet,
Ankush Desai, Nitesh Mor,
Yu-Hsiang Sean Chen, Claire Tuna,
Achal Dave, Jack Kolb,
Eric Allman, Roy Wang,
Bill N. Schilit, Jin Liang,
Chao Mei, Kaifei Chen,
Qifan Pu, Xiang Gao,
Peihan Miao, Zhuo Chen,
Yuting Wei, Chaoran Guo,
Qian Zhong, Tianshi Wang,
Meng Wei,
Limin Chen
1N73LL1G3NC3
15 7H3
4B1L17Y
70 4D4P7 70
CH4NG3
1N73LL1G3NC3
15 7H3
4B1L17Y
70 4D4P7 70
CH4NG3
BwE: Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing [Kumar et al., 2015]
Google Network Infrastructure

![Bandwidth Sharing on a Bottleneck Link](image)

**Figure 4:** Bandwidth Sharing on a Bottleneck Link.

**Figure 5:** BwE Architecture.

**BwE:** Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing [Kumar et al., 2015]

Move from Lagrangian to Eulerian (ask Edward if you don’t know what these words refer to).
Backup Slides.
- **I-frames** are the least compressible but don’t require other video frames to decode. I-frames are further compressed with quantization.

- **P-frames** can use data from previous frames to decompress and are more compressible than I-frames.

- **B-frames** can use both previous and forward frames for data reference to get the highest amount of data compression (not an option in live streaming).

PC: https://en.wikipedia.org/wiki/Video_compression_picture_types
Evaluation: Resource Allocation for Multiple Applications

(a) Resource Fairness

(b) Utility Fairness
Bandwidth Fluctuations (WiFi)

Figure 4: Delivery ratio variation over a week for two randomly chosen 2.4 GHz links.

Figure 5: Delivery ratio variation over a week for two randomly chosen 5 GHz links.


Continue with the main slides.
Augmented Reality

- Training and testing data characteristics
  - 1920x1080 resolution with 30 FPS
  - training: 707 frames (23.5 seconds), testing: 1384 frames (46 seconds)

- Object Recognition
  - Darknet: Open Source Neural Networks in C
  - Developed by Joseph Redmon, "Do whatever you want with it" license
  - It supports CPU/GPU
  - In this work, I am using a pre-trained model with Coco dataset

- Other systems such as TensorFlow, Caffe would also work
IOU and F1

Positive if intersection over union (IOU) larger than 0.5.

\[
\text{IOU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}
\]

(a) IOU = 0.14  
(b) IOU = 0.57  
(c) IOU = 0.82

F1 is the harmonic mean of precision and recall:

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>True Positive</td>
<td>False Positive</td>
</tr>
<tr>
<td>N</td>
<td>True Positive</td>
<td>False Positive</td>
</tr>
</tbody>
</table>

Precision = \( \frac{\text{true positive}}{\text{all positive}} \)

Recall = \( \frac{\text{true positive}}{\text{all detection}} \)

\[
F1 = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}
\]


