Vessels: Efficient and Scalable Deep Learning Prediction on Trusted Processors

Kyungtae Kim*, Chung Hwan Kim°, Junghwan John Rhee¶, Xiao Yu§, Haifeng Chen§, Dave (Jing) Tian*, Byoungyoung Leeᵀ

*Purdue University, °University of Texas at Dallas, ¶University of Central Oklahoma, §NEC Labs America, ᵃSeoul National University
Deep Learning Systems in the Cloud

• Deep Learning (DL) systems are widely used
  • Face recognition, intelligent personal assistants, object detection, etc

• Cloud platforms are popular for running DL services
  • Cost reduction, scalability, flexibility
  • MLaaS competition: AWS, Google Cloud, MS Azure, etc
Data Breaches and Untrusted Environment in the cloud

• Sophisticated data breaches in the cloud

• Emergence of cyber attacks stealing/manipulating ML data
  • Model inversion attacks
  • Neural net. Trojan’ing

• Untrusted computing environment
  • Cloud provider and tenants
  • Compromised VM/container instances
Protecting Deep Learning using Intel SGX

- Intel Software Guard Extensions (SGX)
  - *Enclave*: a hardware-protected memory region
  - Memory protection against privileged software (e.g., Hypervisor, OS)
  - Can protect ML program, model, and data from attacks
  - Availability in the cloud: IBM Cloud Computing Shield, MS Azure Confidential Computing
Limitation of SGX

• Runtime overhead remains to be a problem
  • EPC (Enclave Page Cache): 128 MB (~92MB after metadata for all enclaves)
  • DL with SGX: Large memory, 4-23x prediction time in Linux
  • Frequent EPC page swapping leads to significant performance degradation
  • EPC thrashing: EPC memory is shared by all enclaves

<table>
<thead>
<tr>
<th>Models</th>
<th># Layers</th>
<th>Peak mem size (bytes)</th>
<th>Execution time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Non-SGX</td>
</tr>
<tr>
<td>AlexNet</td>
<td>13</td>
<td>274 M</td>
<td>1.03</td>
</tr>
<tr>
<td>ResNext152</td>
<td>204</td>
<td>566 M</td>
<td>6.88</td>
</tr>
<tr>
<td>DenseNet201</td>
<td>304</td>
<td>376 M</td>
<td>2.54</td>
</tr>
<tr>
<td>InceptionV3</td>
<td>145</td>
<td>337 M</td>
<td>8.34</td>
</tr>
<tr>
<td>VGG16</td>
<td>24</td>
<td>1,121 M</td>
<td>7.43</td>
</tr>
</tbody>
</table>
Vessels: Efficient and Scalable DL with SGX

• Goal: *Minimize memory footprint of DL*
  • Reduce the memory size of DL prediction enclave
  • Efficiency and scalability close to non-SGX prediction
  • Target: CPU-only SGX computation, DL prediction system
  • No accuracy loss (vs. compression and pruning)

• Our approach
  • 1. Profile memory usage of DL: Redundancy discovered
  • 2. DL framework with SGX: Optimize based on profiling
    • *Memory usage planning*
    • *On-demand parameter loading*
    • *EPC-aware scheduling*
Memory Usage Profiling of DL Prediction

Before execution

unused by $L_3$

During execution

current memory usage

*IA = Intermediate Activations
Memory Usage Planning

- **Optimized memory pool**
  - One single memory buffer shared by all layers
  - Recycled for high memory reusability
  - Reduces page swapping significantly (minimal changes to working set)
On-demand Parameter Loading

- Model params are *partially loaded* in an on-demand fashion
  - Model parameters are assigned to a specific layer
  - Identify the *in-file* location of the model parameters for each layer

**Baseline DL**

**Vessels**

- Initial time
- Run time

- memory pool

- execute execute execute execute
EPC-aware Prediction Scheduling

• Concurrent enclaves with multiple cores?
  • A production DL system: receives a large number of prediction requests
  • EPC is not scalable

• Enclave scheduling with an *EPC-commit upper bound*
  • Create a new enclave *only if* the memory usage *will not* exceed the bound
  • Estimate the EPC usage *before* launching it
  • Avoid EPC thrashing
  • Requests are added into a FIFO queue if it has to wait
Evaluation

• Implementation
  • Darknet DL framework
  • Docker-based

• Environmental setup:
  • 9 pre-trained DL models (AlexNet, VGG16, etc)
  • Prediction dataset from ImageNet

• Experiments
  • Single prediction
  • Concurrent predictions
## Evaluation – Single Prediction

- Compared to the baseline SGX system

<table>
<thead>
<tr>
<th>Models</th>
<th>Peak EPC Reduction</th>
<th>Secure Paging Reduction</th>
<th>Latency Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>89.5%</td>
<td>100%</td>
<td>94.01%</td>
</tr>
<tr>
<td>ResNet101</td>
<td>88.1%</td>
<td>100%</td>
<td>69.81%</td>
</tr>
<tr>
<td>ResNet152</td>
<td>91.2%</td>
<td>100%</td>
<td>66.90%</td>
</tr>
<tr>
<td>DenseNet201</td>
<td>88.8%</td>
<td>100%</td>
<td>63.02%</td>
</tr>
<tr>
<td>ResNext152</td>
<td>89.6%</td>
<td>100%</td>
<td>69.47%</td>
</tr>
<tr>
<td>Darknet53</td>
<td>80.6%</td>
<td>100%</td>
<td>69.37%</td>
</tr>
<tr>
<td>InceptionV3</td>
<td>85.4%</td>
<td>100%</td>
<td>57.44%</td>
</tr>
<tr>
<td>VGG16</td>
<td>86.1%</td>
<td>50.53%</td>
<td>57.44%</td>
</tr>
<tr>
<td>YoloV3</td>
<td>73.2%</td>
<td>17.92%</td>
<td>18.89%</td>
</tr>
</tbody>
</table>
Evaluation – Concurrent Predictions

• Based on # of processed requests for 100 minutes compared to baseline
  • Vessels-S: with EPC-aware scheduling, Vessels-M: EPC usage optimization only
  • Improvement: **131%** for Vessels-M, **195%** for Vessels-S (on average)
Conclusion

• Systematic study on EPC usage of current DL prediction systems and Discovery of the inefficiency

• **Vessels**: Efficient and scalable DL prediction with full SGX protection

• 90% EPC footprint reduction per enclave and 195% higher throughput with concurrent enclaves

• No functionality or accuracy loss
Thank you