All-You-Can-Inference
: Serverless DNN Model Inference Suite

Subin Park*  Jaeghang Choi*  Kyungyong Lee
Dept. of Computer Science, Kookmin University, Seoul, South Korea

*Both authors contributed equally to this work
DNN inference task with Serverless Computing

DNN Inference task

- Latency constraints
- Dynamically requests

Challenges with serverless computing

- Limited file storage
- Unstable performance
- Large search space

Not A Major DNN Inference Platform Yet
Opportunity to Enhance Serverless DNN Inference

**ARM Hardware support**
- new hardware type of AWS Lambda
  AWS Gravition2 processors

**Larger Memory Size Support (upto 10GB)**
- higher memory allocations
  - higher **performance**
  - higher **price**

---


https://docs.aws.amazon.com/lambda/latest/operatorguide/computing-power.html
Opportunity to Enhance Serverless DNN Inference

**ONNX (Open Neural Network Exchange)**
- graph optimizer

**Apache TVM**
- operator level compiler

https://github.com/microsoft/onnxruntime-openenclave/blob/openenclave-public/docs/InferenceHighLevelDesign.md
DNN inference task with Serverless Computing

In summary of opportunities for the performance optimization,

Limitations

No prior work using the serverless computing with large search space
Proposed Method

All-You-Can-Inference

estimate the performance of inference tasks on various configurations of FaaS
All-You-Can-Inference

Frontend request with API Gateway

- Web frontend request api gateway url to perform AYCI inference task
All-You-Can-Inference

Data Storage with AWS Services

- Data storage stores the results of inference task metrics
- AWS ECR saves images of lambda environments
All-You-Can-Inference implementation using AWS

Runtime Arbitrator with AWS Step Functions

- Sequentially proceed converter, inference executor and archiver consisting of aws lambda
- Collect lambda metric saved from AWS CloudWatch
All-You-Can-Inference implementation using AWS

Output Report with SES

- Reports the results of inference task metrics to user via an e-mail
All-You-Can-Inference

Demo Web Video

- https://youtu.be/J9fhEb7jEVA
Evaluation Setting

Hardware Type
- ARM
- X86

Optimizer
- Vanilla
- ONNX RUNTIME
- tvm

DNN Models

CNN
- SqueezeNet
- ShuffleNet
- MobileNetV2
- MNasNet
- EfficientNetB0
- ResNet18
- ResNet50
- InceptionV3
- AlexNet
- VGG16

NLP
- BERT

Inference Elements

- Memory Allocation
  - batch size of 1, 2, 4, 8, 16, 32
  - lambda memory size of 0.5, 1, 2, 4, 8, 10GB
Compare between CNN and NLP model

Observation 1: Similar performance patterns with CNN and NLP

Setting

- Models: VGG16 and BERT
- Memory size: 10GB

Best Performance

- Intel hardware: Vanilla
- ARM hardware: TVM
Performance benefit with batch processing

Observation 2: Batch processing improves performance higher in vanilla

Setting
- Intel hardware and 10 CNN models
- Batch sizes: 1 ~ 32
- Memory size: 0.5 ~ 10GB

Evaluation
- Vanilla model shows performance benefit especially about 1.58x with batch size of 16
Efficient memory allocation

Observation 3: memory size of 2GB results in the best performance in most cases

Denotation

- Relative cost-performance normalization

$$\frac{M}{2} \times \frac{\text{Latency}(M)}{\text{Latency}(2)}$$

Evaluation

- Performance peak point: 2GB
- As the configured memory size becomes larger, cost-performance metric drops
The number of most efficient cases

Observation 4: Performance of ARM hardware is not as good as Intel hardware

Denotation
- count the number of best performing cases

Evaluation
- Best performance: Intel-ONNX
- ARM-Vanilla, ARM-TVM, and Intel-Vanilla often perform the best
Conclusion

- Proposed system uncovering challenges in the Faas environment setup and performance variations for distinct models
- Helps users to build an optimal serverless DNN inference system
Q&A