INTERNATIONAL CONFERENCE ON PARALLEL PROCESSING

ICPP / 2021 / CHICAGO / USA

acm In-Cooperation

sighpc

AUGUST 9-12, 2021

AMPS-Inf: Automatic Model Partitioning for Serverless Inference with Cost Efficiency

Jananie Jarachanthan[*], Li Chen[*], Fei Xu[†], Bo Li[‡]

[*] University of Louisiana at Lafayette, [†] East China Normal University,

[‡]Hong Kong University of Science and Technology

UNIVERSITY of LOUISIANA LAFAYETTE

Serverless Computing

- Serverless computing provides an abstraction over complex operations and programming of servers.
- Serverless computing becomes increasingly popular due to its auto-scaling and pay-per-use natures.

AWS Lambda

Google Cloud Functions

Azure Functions

Serverless Computing

- Serverless applications.
  - Real-time video encoding: ExCamera[1]
  - Interactive data analytics: Locus[2], PyWren[3]
  - Web applications
  - Machine learning: Cirrus[4], SIREN[5]

AWS Lambda

Google Cloud Functions

Azure Functions

[1] Encoding, Fast and Slow: Low-Latency Video Processing Using Thousands of Tiny Threads, NSDI'17.
[2] Shuffling, Fast and Slow: Scalable Analytics on Serverless Infrastructure, NSDI'19.
[3] Occupy the Cloud: Distributed Computing for the 99%, SoCC'17.
[4] Cirrus: a Serverless Framework for End-to-end ML Workflows, SoCC '19.
[5] Distributed Machine Learning with a Serverless Architecture. In IEEE Conference on Computer Communications, INFOCOM 2019.

Serverless Machine Learning

- How can machine learning applications exploit serverless computing?

"Serverless Inference?"

Challenges

- Faster growth of the size and complexity of advanced neural network models.

Neural network models	Model size	Deployment size
ResNet50	98MB	267MB
InceptionV3	92MB	261MB

ResNet50 model has 25,636,712 parameters and its size is (25,636,712 ∗4)/1024/1024 ≈ 98MB.
Deployment size=Keras dependencies (169 MB)+Model size

Serverless Machine Learning

- How can machine learning applications exploit serverless computing?

"Serverless Inference?"

Challenges

- Minimization of the billing cost without violating a pre-defined Service Level Objective or SLO in term of query response time.

Serverless Inference

Partition the Model

- How to split the complex computation graph?

  How to find specific partition among a gigantic number of possible ones?
- How to coordinate the partitions?

  How to coordinate intermediate outputs?
- Which function resource type to specify for each partition?

  How to choose from a large resource configuration space?

"End-to-end query response time and the total billing cost"

Serverless Inference

Partition the Model

- How to split the complex computation graph?

  How to find specific partition among a gigantic number of possible ones?

- How to coordinate the partitions?

  How to coordinate intermediate outputs?

- Which function resource type to specify for each partition?

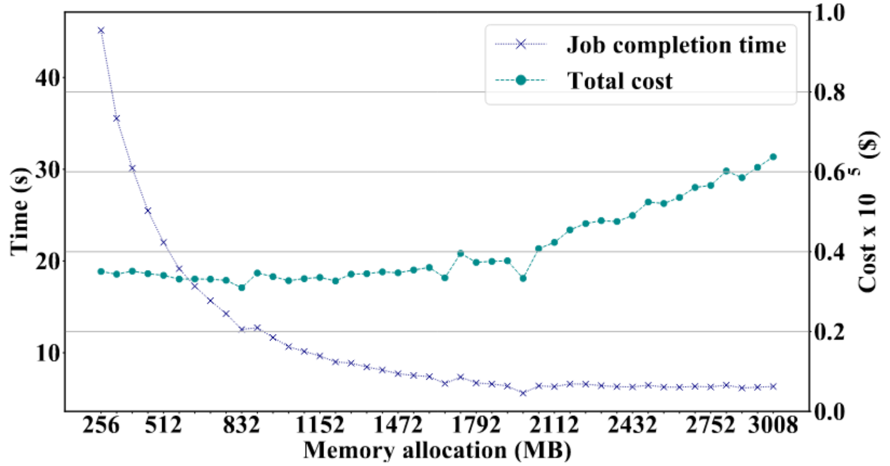
  How to choose from a large resource configuration space?

AMPS-Inf

Serverless Inference

Motivating experiments

- Deployment and Configuration:
  - o Lambda function with the serving code (e.g., Python).
  - o Model weights (.h5).
  - o Associated ML dependency libraries (e.g., Keras).
  - o ML model to be inferenced (.YAML).
  - o Resource configurations (e.g., the allocated memory block).



Serverless Inference

Motivating experiments

- Inference of pre-trained Keras MobileNet (<250MB).
- Inference of pre-trained Keras ResNet50 (>250MB).

AWS Lambda

Amazon SageMaker

- Sage 1: Jupyter Notebook instance.
- Sage 2: Hosting instance.

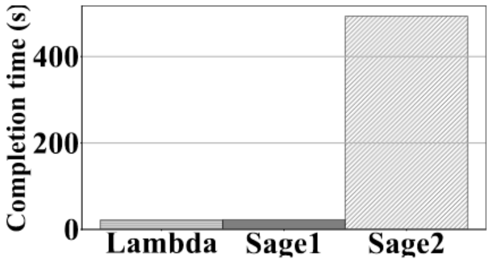
Serverless Inference

Motivating experiments

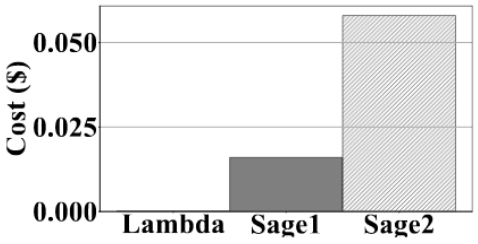
- Inference of pre-trained Keras MobileNet (<250MB).

Inference with one lambda.

	Lambda settings	SageMaker settings
Input	.pkl image	Same image (.jpg)
Model	YAML	JSON



(a)



(b)

Memory (MB)	512	1024	1536	2048
Time (s)	22.03	10.65	7.52	6.38
Cost ($)	0.00018	0.00017	0.00019	0.00021

Serverless Inference

Motivating experiments

- Inference of pre-trained Keras MobileNet (<250MB).
- Inference of pre-trained Keras ResNet50 (>250MB).

<u>Inference across lambdas</u>

Settings	Sage 1	Sage 2	Lam. 512MB	Lam. 1024MB
Time (s)	33.346	484.509	47.078	21.799
Cost ($)	0.014	0.056	0.0017	0.0011

Completion time and cost of ResNet50 serving (one image) in different settings.

Serverless Inference Model For Cost-efficiency And Timely Response

- Consider a pre-trained neural network model with Y layers.
- There is N complete set of possible model partition combinations.
- Given a particular partitioning $g \in N$, we specify k, $k \leq K$ lambdas to be coordinated for model serving.
- K is the limit on the maximum number of lambdas that can be requested.
- The number of layers in the partition that the i-th lambda ($i \in \{1,2,\cdots,k\}$) will be allocated is represented by an integer variable y_i^g
- Each lambda's memory allocation can be any from L memory blocks.
- When i-th lambda is allocated with the j-th type of memory ($j \in \{1,2,\cdots,L\}$), binary variable denotes $x_{j,i}^g$ the memory allocation.

Serverless Inference Model For Cost-efficiency And Timely Response

- The completion time of i^{th} lambda consists of computation time and data transfer time.
- The monetary cost incurred by the lambda depends on the execution time, price of its allocated memory, S3 storage cost, cost of Get and Put request from S3, and the lambda invocation cost.

It is a mixed-integer quadratic programming (MIQP) and can be solved by any MIQP solver.

$$\min_{\boldsymbol{x},\boldsymbol{y}} \quad S^g_{i,j}$$ Cost minimization problem

$$\text{s.t.} \quad y^g_i e^g_i + D + F \leq A$$ Deployment size

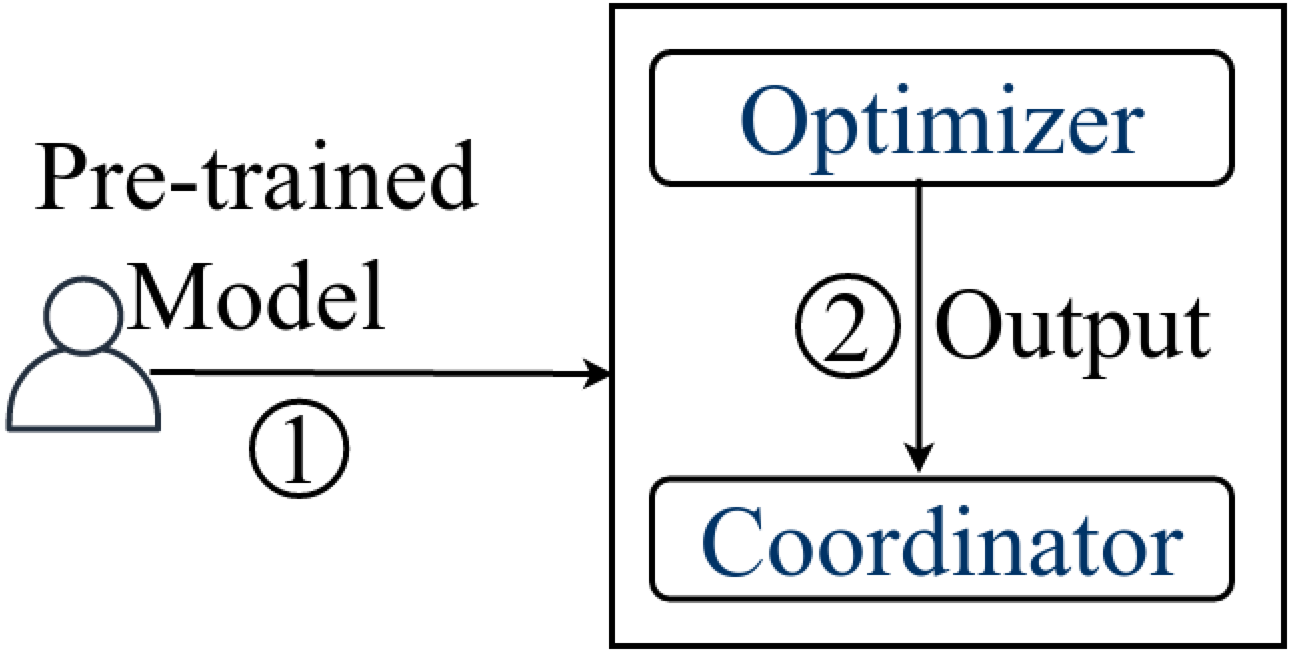
$$y^g_i z^g_i + p^g_{i-1} \leq J$$ Temporary storage size

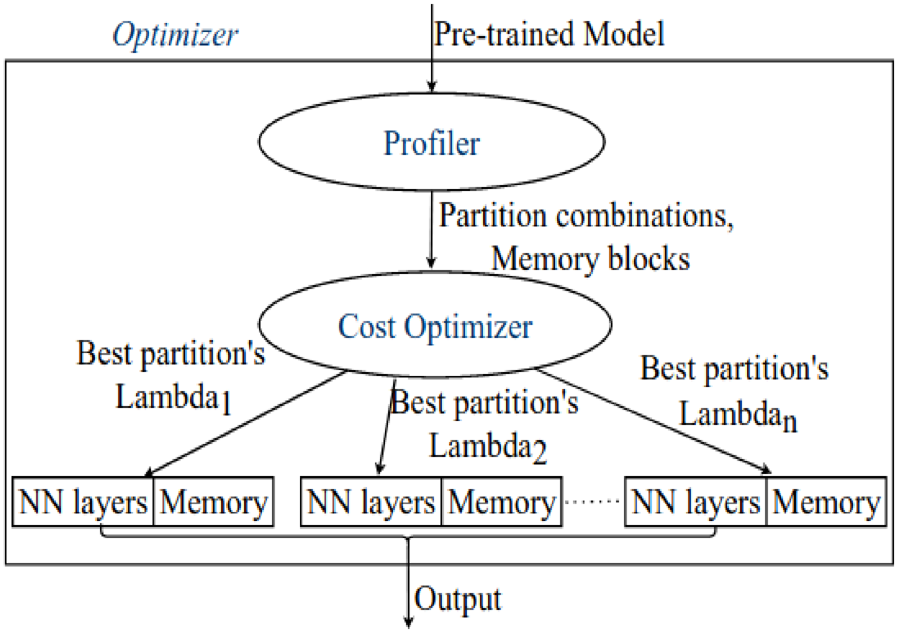
$$y^g_i \leq \lceil Y/k \rceil$$ Number of neural network layers per partition

$$1 + \lceil ((\sum_{j \in L} x^g_{j,i} z^g_i) + D + F - M)/\beta) \rceil \leq j$$ Number of memory blocks

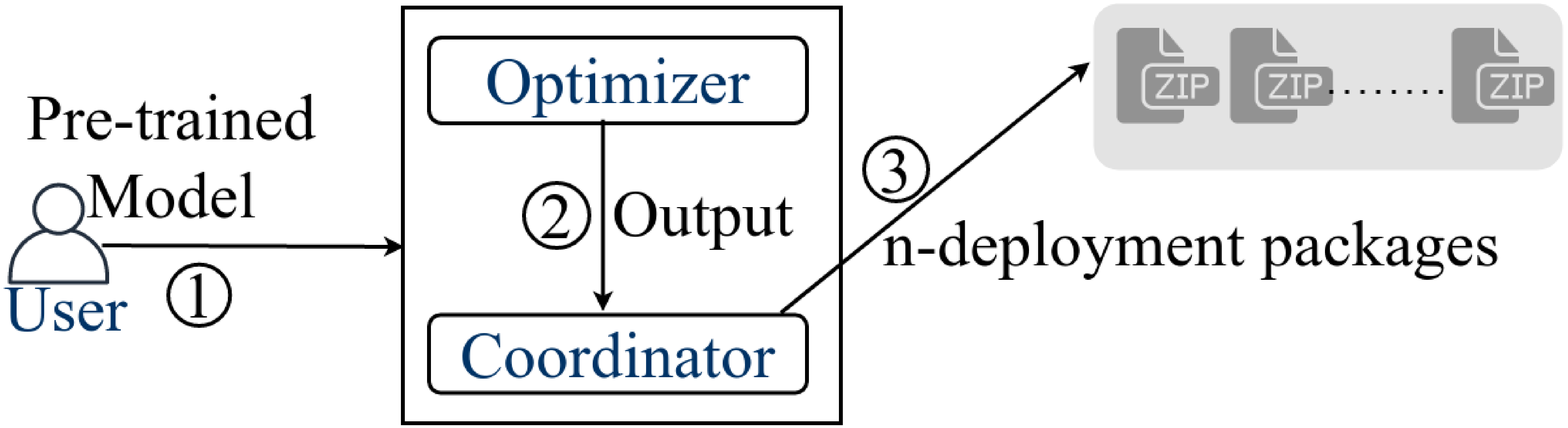
$$x^g_{j,i} \in \{0, 1\}$$

Architecture overview of AMPS-Inf





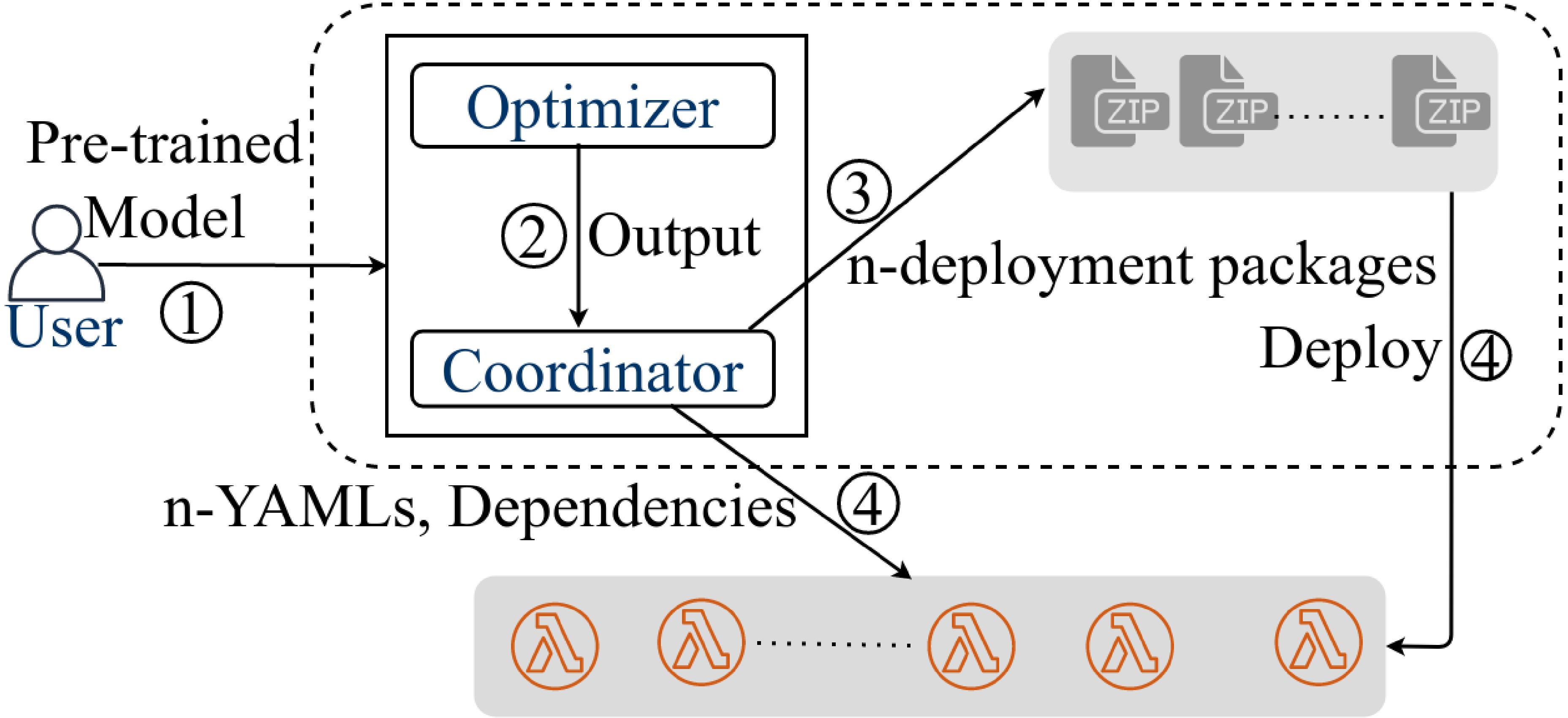
Architecture overview of AMPS-Inf



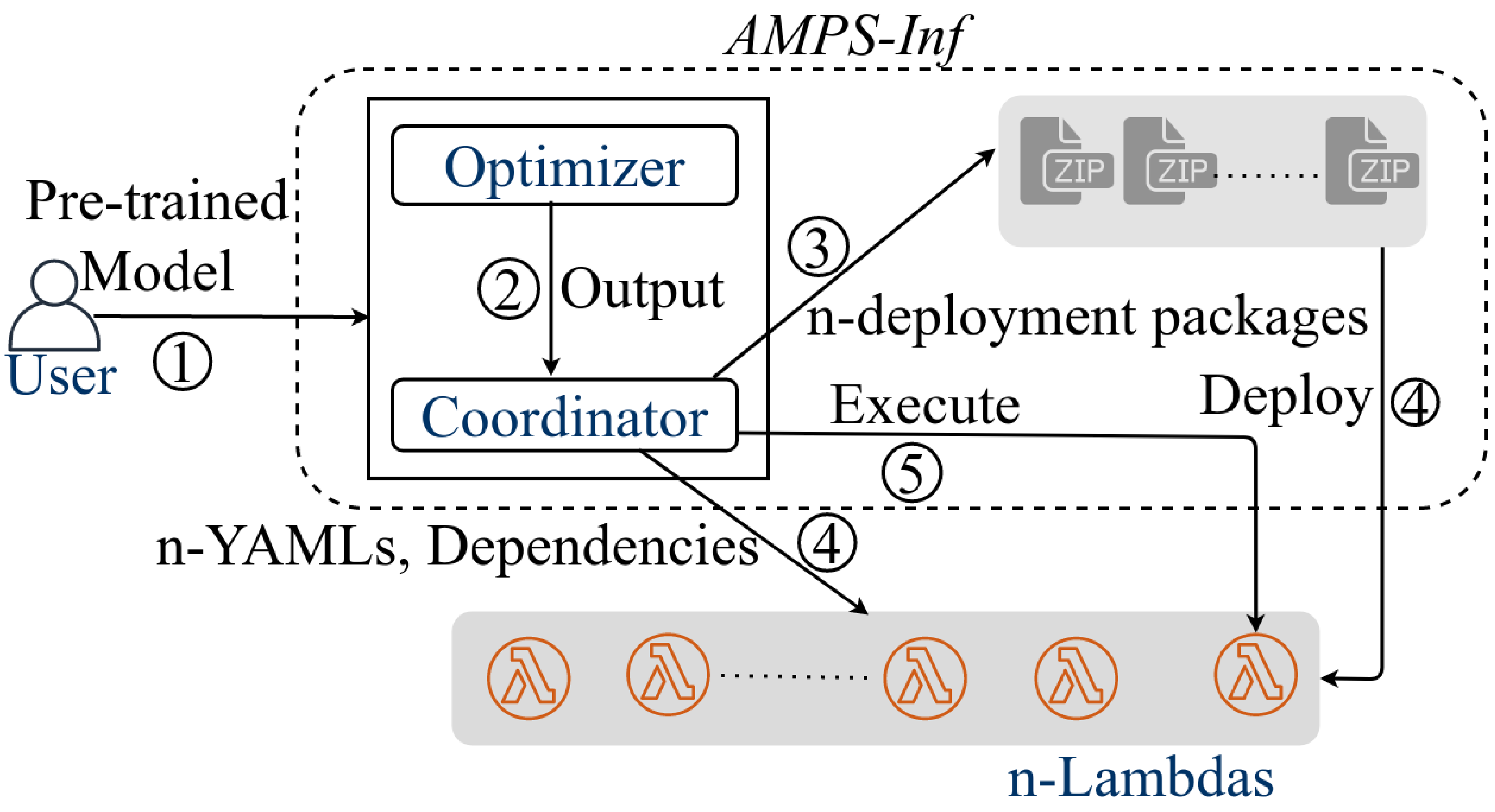
= Funtion +Weights(.h5)

Output = Best configuration (Partitions, Lambdas' memories)

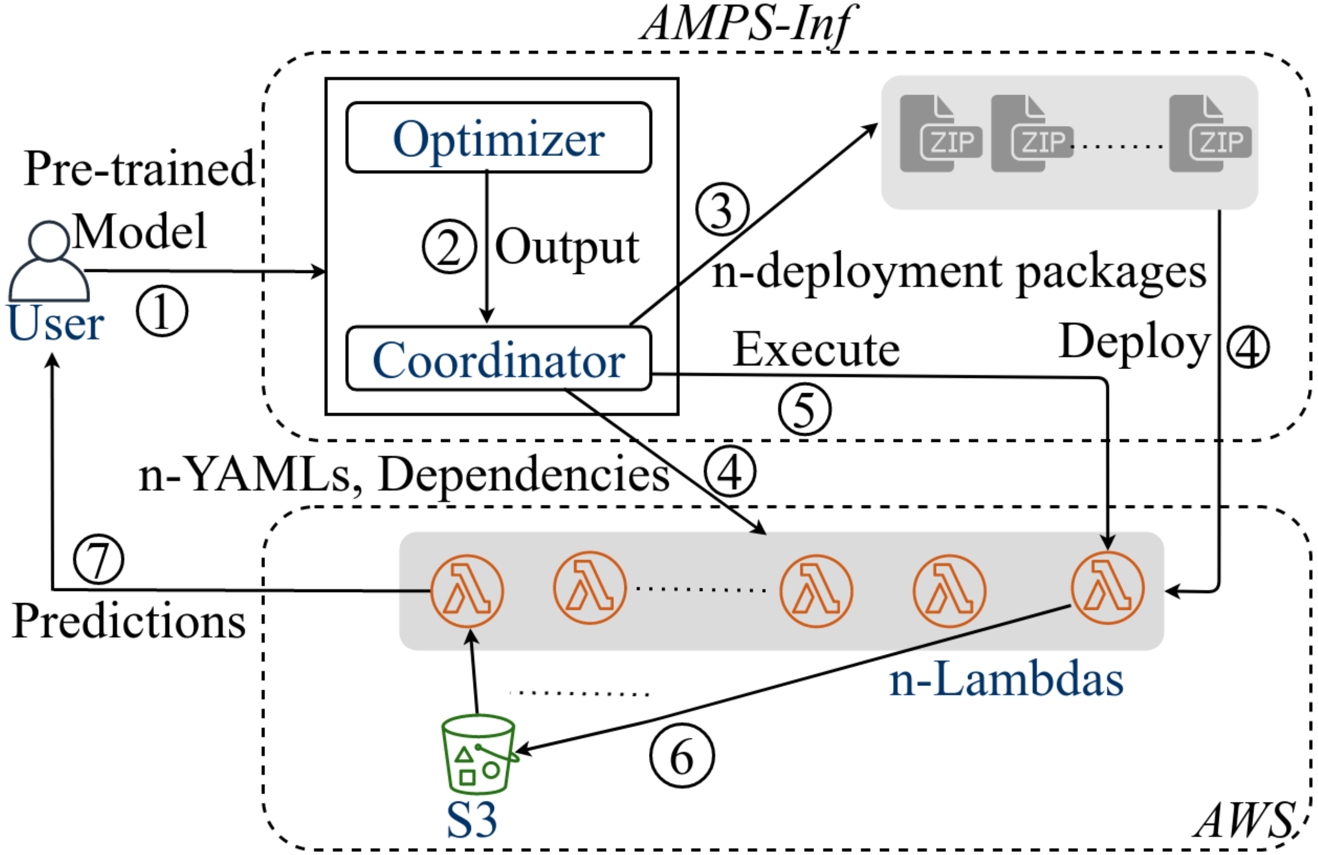
Architecture overview of AMPS-Inf



Architecture overview of AMPS-Inf



Architecture overview of AMPS-Inf



= Funtion +Weights(.h5)

Output = Best configuration (Partitions, Lambdas' memories)

Performance Evaluation

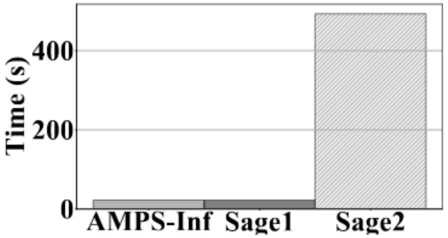
Experimental setups

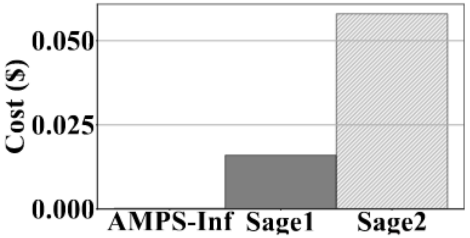
- Input of AMPS-Inf: .pkl single image, YAML pre-trained model and weights (.h5) .
- AMPS-Inf settings: AWS Lambda, AWS S3

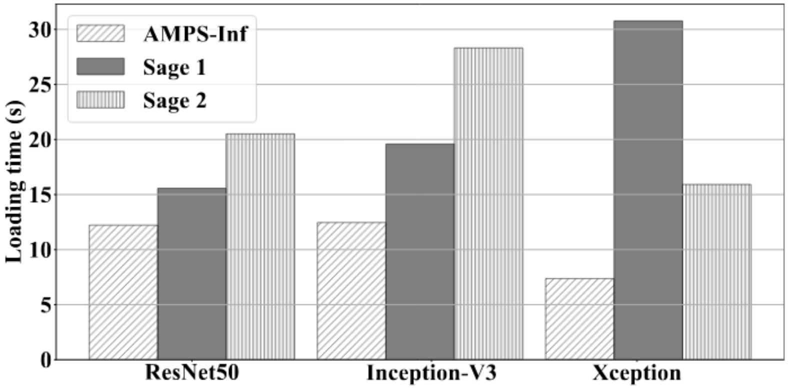
Comparison with	Settings	Platform	Input	Models
Amazon Sage Maker	Sage 1	Instance-based notebook (ml.t2.medium)	Model (JSON), Weights (.h5), Single image	MobileNet, ResNet50, InceptionV3, Xception
	Sage 2	Instance-based notebook (ml.t2.medium), hosting instance (ml.m4.xlarge), AWS S3		

Performance Evaluation

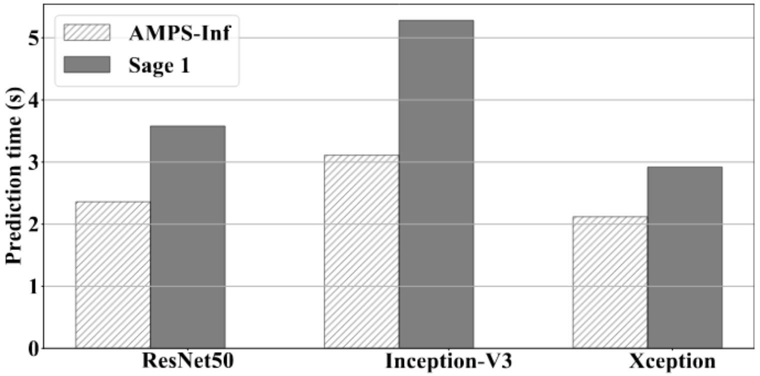
Small model (MobileNet inference)







The time for loading model and weights



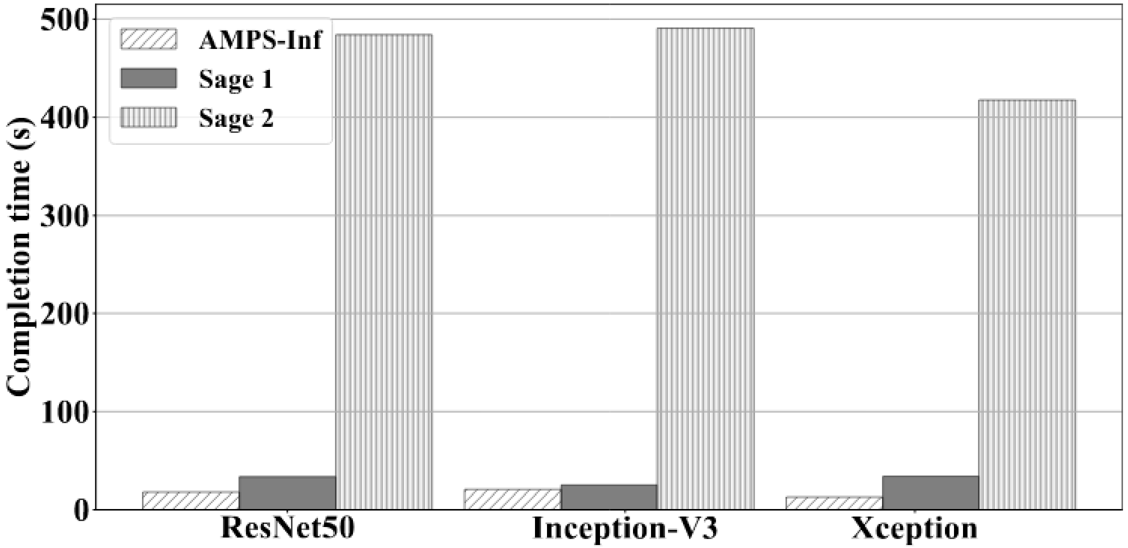
The time for prediction

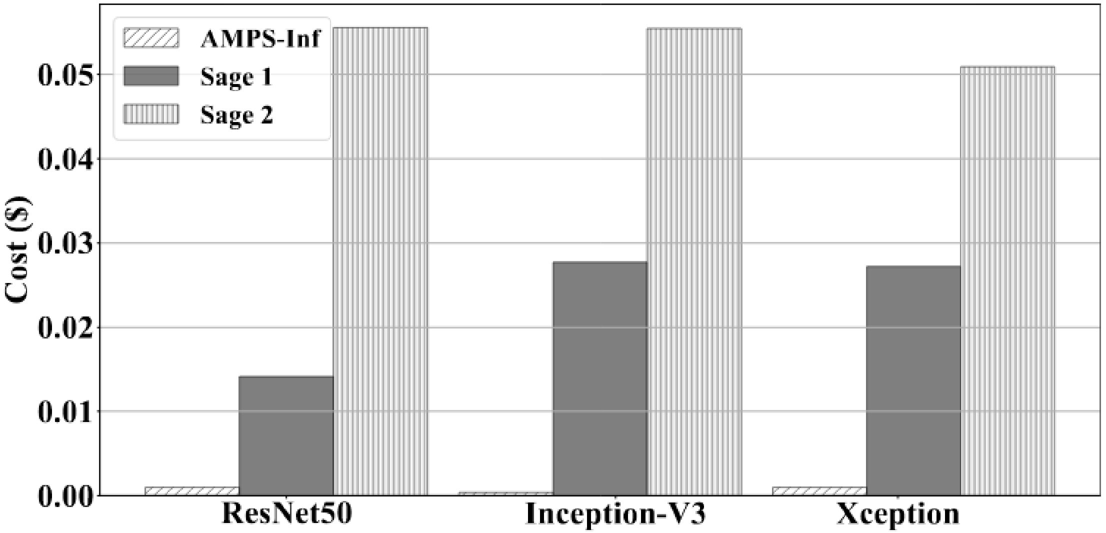
ResNet50	Inception-V3	Xception
463.48	462.30	401.78

The overall time spent for deployment and prediction in Sage 2

Performance Evaluation

Comparison with SageMaker for one image request:



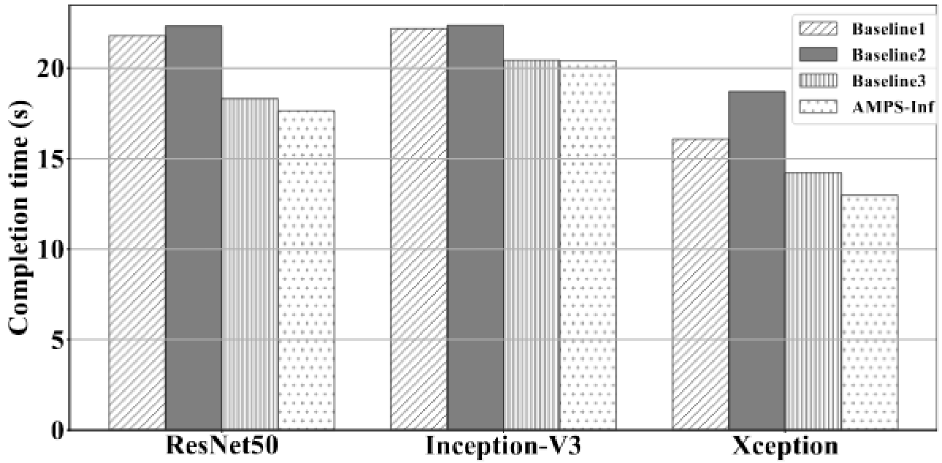


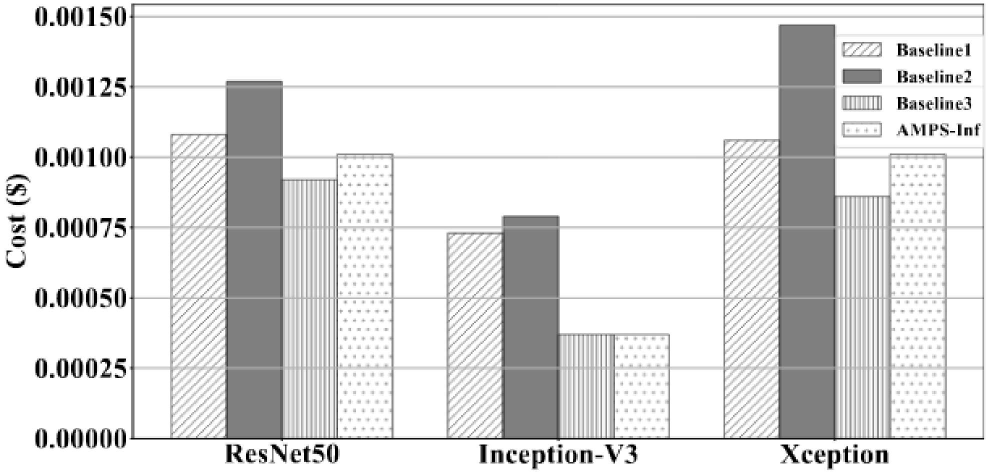
- Cost reduction of AMPS-Inf for ResNet50, Inception-v3, and Xception by 92.85%, 98.67%, and 96.29%, respectively, when compared to Sage 1.
- In comparison with Sage 2, AMPS-Inf achieves cost reduction of 98.18%, 99.33%, and 98.02%, respectively, for the three models.

Performance Evaluation

Experimental setups

Comparison with	Settings	Platform	Input	Models
Baselines	Random baseline	AWS Lambda, AWS S3.	Model (YAML), Weights (.h5), Single image (.pkl).	ResNet50, InceptionV3, Xception.
	Heuristic baseline			
	Optimal baseline			

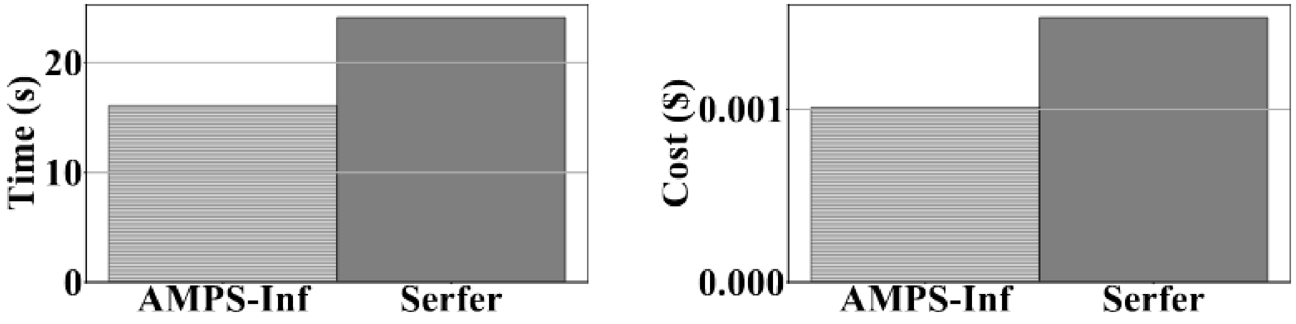




Performance Evaluation

Experimental setups

Comparison with	Settings	Platform	Input	Models
state-of-the-art [1]	Same partition and configuration as AMPS-Inf.	AWS Lambda, AWS S3, AWS EC2, AWS Step functions.	Model (Python function), Single image.	ResNet50.



(a) (b)

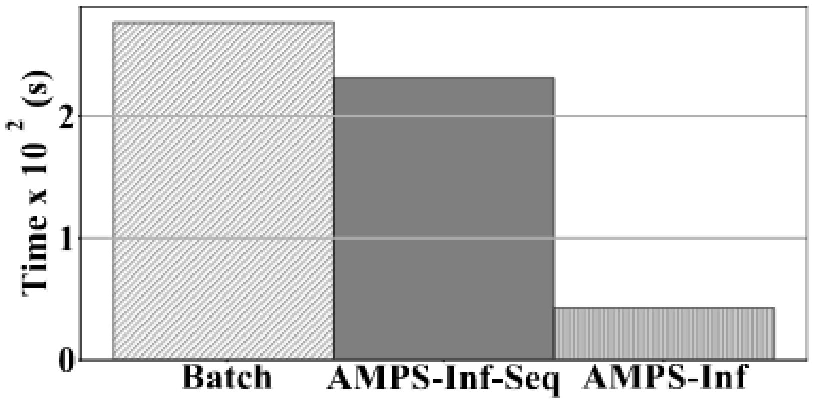
[1] SerFer: Serverless Inference of Machine Learning Models, https://divatekodand.github.io/files/serfer.pdf

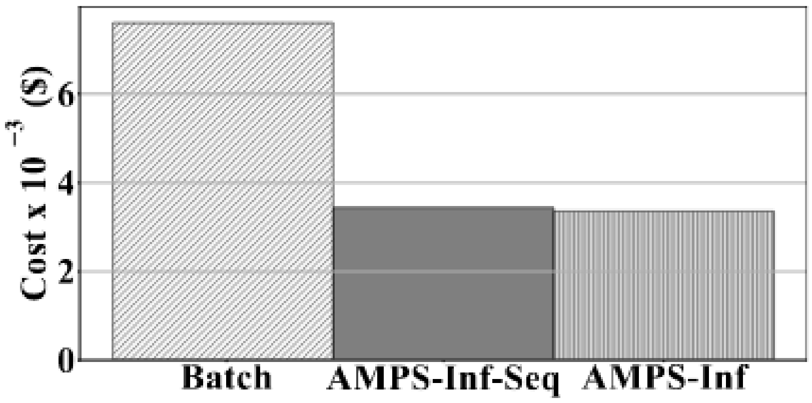
Performance Evaluation

Experimental setups

AMPS-Inf : Two lambdas/partitions per batch.

Comparison with	Settings	Platform	Input	Models
BATCH[1]	100 images in 10 batches, Single lambda per batch.	AWS Lambda, AWS S3.	Single image	MobileNet





[1] Batch: Machine Learning Inference Serving on Serverless Platforms with Adaptive Batching, SC20

AMPS-Inf

- To address the challenges on splitting model and coordinating partitions, and to hide these complexities from users, we design and implement AMPS-Inf, an automated framework for serverless machine learning inference towards cost-efficiency and timely-response.
- AMPS-Inf is evaluated with four pre-trained models, in comparison with Amazon SageMaker and three different baselines.
- AMPS-Inf outperformed the state-of-the-art in cost and performance.
- AMPS-Inf extended for the batch inference and compared with an existing work BATCH.
- Results demonstrate that AMPS-Inf, by finding the best configuration of lambda resource type and model partitions, achieves cost saving of up to 98% without degrading the response time performance.

AMPS-Inf's future

- Extend to other platforms.
- Extend the design for batch inference serving at a very large scale.
- Evaluate AMPS-Inf for the models in other frameworks such as Tensorflow, PyTorch, and etc.
- Quantize weights for models with complex single layer.