SplitPlace: Intelligent Placement of Split Neural Nets in Mobile Edge Environments

Shreshth Tuli
Advisors: Dr. Giuliano Casale and Prof. Nicholas R. Jennings
Department of Computing, Imperial College London

1. Challenges and Motivation
Deep learning models have become ubiquitous and deploying the modern neural models has several challenges:
1. Exorbitantly high resource requirements.
2. High Cost of revamping execution infrastructure.
3. Environmental impact and sustainability issues.

SplitPlace is the first intelligent approach for efficient deployment of large-scale deep learning in mobile-edge environments leveraging layer and semantic splitting.

2. Layer and Semantic Splitting
We leverage the only two available schemes for splitting neural networks into lightweight modular smaller components: layer and semantic splitting [1, 2]. The layers splits are disjoint sequential neural networks that produce intermediate results, whereas the semantic splits are parallel models that produce a part of the result.

3. Exploiting the Trade-off
- Layer Split:
  - Higher response time (sequential execution).
  - Higher accuracy (parameter sharing among splits).
  - No re-training required.
  - Best for accuracy-critical tasks.
- Semantic Split:
  - Lower response time (parallelized execution).
  - Lower accuracy (no parameter sharing).
  - Re-training required.
  - Best for time-critical tasks.

4. System Model
In this work, we assume a scenario with multiple heterogeneous mobile-edge nodes in a master/slave fashion. All tasks are received from an IoT layer to collect data from the users and sent it across to the edge broker via the gateway devices. The edge broker then decides which split strategy to use and schedules these fragments to various edge nodes based on deployment constraints.

The distribution of network splits is based on the resource availability, computations required to be performed in each section and the capabilities of the nodes (obtained by the Resource Monitor). The allocation and migration of neural network splits is done by the Container Orchestrator.

5. Multi-Armed Bandit Model
We utilize a Multi-Armed Bandit (MAB) formulation of the splitting scheme selection problem where the reward metric for a set of workloads (W) is expressed as

\[ R = \frac{\sum_{i \in W} \left( \text{Response Time}_i \leq S \cdot A_{\text{opt}} \right) + \text{Accuracy}_i}{|W|} \]

In the SplitPlace model, we maintain MABs for two different contexts: 1) when SLA > response time estimate for a layer decision, 2) when SLA < this estimate.

6. Experiments and Results
We perform experiments by running the SplitPlace model on 100 fixed size scheduling intervals, each 5 minutes long.
- Setup: 50 diverse and Raspberry Pi like VMs with highly-constrained compute and memory resources.
- Workloads: Our workloads are popular image-classification models of ResNet50-V2, MobileNetV2 and InceptionV3 on MNIST, FashionMNIST and CIFAR100 datasets. We create workloads using Poisson distribution in each interval.
- Placement: SplitPlace is agnostic to the underlying placement module. For our experiments, we combine it with a popular Asynchronous-Agent-Critic (based on reinforcement learning) scheduler [4].

To emulate mobility, we use Gaussian noise in the network latency using the netlimiter Linux tool.

7. Key Takeaways
1. SplitPlace uses Multi-Armed-Bandits to exploit the trade-off between layer and semantic split to efficiently optimize the total reward.
2. Both MAB and reinforcement learning placement models are dynamically tuned to adapt to volatile scenarios.
3. SplitPlace outperforms state-of-the-art methods giving higher inference accuracy and lower SLA violations.

8. Future Plans
1. Explore how to dynamically adapt the splitting configuration for more heterogeneous edge environments.
2. Adapt the scheduling policy to be aware of the split decisions for more focused task allocation.
3. Incorporate other deep learning models like attention based neural networks or transformer models.

9. References