DNN Model Placement for Edge Intelligence

Convex Optimization 2 Course Project, Nima Samadi, Mohammad Javad Mohammadi

***Abstract*—Edge Intelligence (EI) has emerged as a promising area in recent years. In order to mitigate some of the challenges associated with using AI in cloud servers, EI utilizes edge computing to deliver AI services to users. The dominant approach to creating AI services is deep neural networks (DNN) which have shown significant results in computer vision, language processing, robotics, etc., tasks. In the cloud, these models run on powerful servers, but they are slow to respond, have privacy concerns, and have low reliability. EI can help to overcome these obstacles. However, edge servers are typically constrained in terms of processing power and energy. Therefore, special care must be taken to deploy DNN models on edge servers. Our purpose in this project is to partition DNN models and place each part either on edge servers or the user device. We use optimization approaches to analyze this problem.**

***Index Terms*—Edge intelligence, Model placement, Neural network, Convex optimization**

I. Introduction

Deep Neural Networks (DNNs) have become increasingly popular as the core machine learning technique in many fields, such as speech recognition, image classification, translation, language modeling, and video captioning. DNNs are widely used to perform these tasks due to their high accuracy and adaptability. The training of model parameters requires massive amounts of data that can only be achieved with powerful hardware and adequate time. Additionally, since neural networks are widely used, it is essential to research how to utilize and train them optimally.

One solution to tackle this issue is cloud computing, which undoubtedly poses real challenges to network capacity and the computing power of cloud computing infrastructures. Also, many new applications, e.g., cooperative autonomous driving, are sensitive to delay requirements that the cloud infrastructures would have difficulty meeting since they may be far from the users.

Another solution that has become a hot topic in the computation context is edge computing. Edge computing has many benefits, such as alleviating the network traffic load as less data is exchanged with the cloud compared to the cloud-only scenario. Furthermore, services hosted at the edge can substantially reduce the delay time of data transmissions and improve the response time. The users' privacy is also enhanced as the private data is stored locally on the edge or user devices instead of cloud servers. The hierarchical computing architecture provides more reliable computation, and finally, edge computing can promote the pervasive application of DNNs.

Cloud computing and edge computing aren't mutually exclusive, and that's crucial to understand. In other words, edge computing complements and extends cloud computing. High processing power, giant storage, and data backup are some of the capabilities of cloud servers that edge servers lack. On the contrary, low latency, privacy assurance, and real-time processing are some of the capabilities of edge servers that cloud servers lack. The combination of these computation paradigms enables cumulating advantages of each and reduces the disadvantages.

Our focus is on the DNN model placement in the context of edge intelligence.

II. Related work

Throughout this section, we review some research related to mobile edge intelligence, server selection, and model placement.

Edge intelligence (EI), the integration of mobile edge computing (MEC) and AI technologies, has recently emerged as a promising paradigm to support computation-intensive AI applications at the network edge [1]. An edge-based MEC network with multiple edge servers is studied in [2] to maximize the number of computing offloading requests under edge storage, computation, and communication constraints. To cope with the unknown and fluctuating service demand, [3] proposed an online learning algorithm to optimize spatial-temporal dynamic service placement decisions among multiple edge servers to minimize the computation delay. Considering parallel computing at both cloud and edge servers, [4] and [5] studied collaborative service placement and computation offloading to minimize the computation latency.

Due to resource limitations on the edge server, server selection has received considerable attention in recent years [6]. Researchers have proposed techniques to achieve optimal goals when selecting servers, such as minimizing the average delay [7], maximizing the resources utilization efficiency [8], minimizing energy consumption [9] and etc.

In some researches like [10], to deal with edge computing's challenges, they model the problem of continuous server selection as a Markov Decision Process (MDP). The difficulty of this problem is that achieving long-term optimum requires future knowledge, such as usermobility, serverworkload, etc, which is not known a priori and they for dealing to this issue, they have proposed Deep Reinforcement Learning (DRL). Although, this technique can reach to good solutions but it's crucial to consider that the amount of time and computations to train DRL models is considerable. Besides, the DRL model should be trained periodically thus it seems this method can't be use in many situations specially in real-time application.

III. Methodology

We analyze this problem from an optimization point of view. This technique is used in [11], [12], and [13] articles. Every DNN model can be thought of as a directed graph in which

nodes are operations and edges are data (tensor in software terms). The direction of the edge shows the dependency between nodes. One can model this graph with three possible granularities: 1)neuron, 2)operation, and 3)layer. With neuron granularity, the dimension of the problem is quite high and the number of connections between neurons prevents partitioning the graph effectively. Layer abstraction, however, results in a small graph, which makes parallelization difficult. Therefore, we use operation granularity in this project.

Placement problems consist of two subproblems: 1)graph partitioning and 2)task assignment. Graph partitioning requires segmentation points that enable parallel execution of each subgraph and minimizes dependency between subgraphs. When subgraphs are created, they must be processed on the user device or an edge server. This task assignment problem can be solved via optimization techniques.

Let $G(V, E)$ to be the model graph and $\mathcal{P} = \{p_1, p_2, \cdots p_k\}$ be the list of k processors. A model placement means finding the set of $\{(p_1, s_1), (p_2, s_2), \cdots, (p_k, s_k)\}$, where each s_i is a subpraph and all s_i form a partition of graph. In other words, $\bigcup s_i = V$, and $\bigcap s_i = \emptyset$. Among all feasible points, we choose one that minimizes total latency of model execution. The objective function depends on how latency is defined and will be the subject of project itself.

We'll propose a model placement algorithm that considers dynamic changes in the environment, edge servers, and the model itself. This algorithm will be simulated in various scenarios, and results will be compared with realated works.

REFERENCES

[1] Z. Lin, S. Bi, and Y.-J. A. Zhang, "Optimizing ai service placement and resource allocation in mobile edge intelligence systems," *IEEE Transactions on Wireless Communications*, vol. 20, no. 11, pp. 7257–7271, 2021.

[2] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 10–18.

[3] L. Chen, J. Xu, S. Ren, and P. Zhou, "Spatio–temporal edge service placement: A bandit learning approach," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 8388–8401, 2018.

[4] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 207–215.

[5] L. Chen, C. Shen, P. Zhou, and J. Xu, "Collaborative service placement for edge computing in dense small cell networks," *IEEE Transactions on Mobile Computing*, vol. 20, no. 2, pp. 377–390, 2019.

[6] H. Liu and G. Cao, "Deep reinforcement learning-based server selection for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 351–13 363, 2021.

[7] M. Sheng, Y. Dai, J. Liu, N. Cheng, X. Shen, and Q. Yang, "Delay-aware computation offloading in noma mec under differentiated uploading delay," *IEEE Transactions on Wireless Communications*, vol. 19, no. 4, pp. 2813–2826, 2020.

[8] K. C.-J. Lin, H.-C. Wang, Y.-C. Lai, and Y.-D. Lin, "Communication and computation offloading for multi-rat mobile edge computing," *IEEE Wireless Communications*, vol. 26, no. 6, pp. 180–186, 2019.

[9] M. Li, N. Cheng, J. Gao, Y. Wang, L. Zhao, and X. Shen, "Energy-efficient uav-assisted mobile edge computing: Resource allocation and trajectory optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3424–3438, 2020.

[10] H. Liu and G. Cao, "Deep reinforcement learning-based server selection for mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 351–13 363, 2021.

[11] J. Tarnawski, A. Phanishayee, N. R. Devanur, D. Mahajan, and F. N. Paravecino, "Efficient algorithms for device placement of DNN graph operators," *CoRR*, vol. abs/2006.16423, 2020. [Online]. Available: https://arxiv.org/abs/2006.16423

[12] P. Lin, Z. Shi, Z. Xiao, C. Chen, and K. Li, "Latency-driven model placement for efficient edge intelligence service," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 591–601, 2022.

[13] I. Lujic, V. De Maio, S. Venugopal, and I. Brandic, "Sea-leap: Self-adaptive and locality-aware edge analytics placement," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 602–613, 2022.