Distributed Model Training Based on Data Parallelism in Edge Computing-Enabled Elastic Optical Networks

Yajie Li, Zebin Zeng, Jun Li, Boyuan Yan, *Member, IEEE*, Yongli Zhao, *Senior Member, IEEE*, and Jie Zhang

***Abstract*— The emergence of edge computing provides an effective solution to execute distributed model training (DMT). The deployment of training data among edge nodes affects the training efficiency and network resource usage. This letter aims for the efficient provisioning of DMT services by optimizing the partition and distribution of training data in edge computing-enabled optical networks. An integer linear programming (ILP) model and a data parallelism deployment algorithm (DPDA) are proposed to solve this problem. The performance of the proposed approaches is evaluated through simulation. Simulation results show that the proposed algorithm can deploy more DMT services compared with benchmark.**

***Index Terms*— Data parallelism, distributed model training, edge computing, optical networks.**

I. Introduction

With the development of artificial intelligence (AI), there is a booming number of AI-based applications and services. More and more enterprises require AI-based applications, including Web search, recommendation systems, and document analysis [1]. In this paradigm, cloud service providers are expected to provide AI services (e.g., data analysis and model training) for enterprises, which usually lack AI-related technologies and network infrastructure (e.g., computing and storage). Model training is a time-consuming but necessary part during the provisioning of AI applications, which requires high storage and computing resources to process a large amount of raw data.

To shorten the training time and relieve the resource demand on a single node, distributed model training (DMT) has been proposed in the form of cloud-edge coordination [2], [3], which mainly includes model parallelism [4] and data parallelism [5]–[7]. In practical system, Parameter Server can be used to run distributed training for models such as Sparse Logistic Regression and Latent Dirichlet Allocation [5]. In the case of data parallelism, the training data is partitioned and distributed into multiple edge nodes. The training is completed through a number of iterations. Each iteration includes data interaction steps between the edge and the cloud. Once the required model accuracy is reached, the training process will be terminated. Note that the distribution of training data among edge nodes brings a high demand of transport resources. With long distance and high capacity transmission, elastic optical networks (EONs) is a promising solution for the distribution of training data.

Although improving the efficiency of model training, data parallelism-based DMT is facing the challenge that how to efficiently deploy training data among multiple edge nodes. Specifically, different partition and distribution of training data will affect the usage of computing and transport resources in the network. With limited resources in networks, given a batch of training tasks from users, cloud service providers wish to find the optimal data partition and distribution for each task so as to perform as many DMT tasks as possible.

Many works have been conducted on data parallelism-based DMT. Deep gradient compression is proposed to reduce the communication bandwidth [6], and the authors in [7] propose two methods to reduce the uplink communication cost. The authors in [8] formulate a decentralized machine learning in wireless network as an optimization problem that captures the trade-off between the training time and energy consumption. The work in [9] describes the DNN model partition and deployment in metro optical networks. However, few works investigate how to efficiently execute DMT based on dynamic partition and distribution of training data in edge computing enabled EONs (EC-EONs).

In this letter, we focus on how to efficiently provide data parallelism-based DMT services in EC-EONs. The objective is to maximize the provisioning of DMT services by optimizing the partition and distribution of training data among edge nodes. This problem is formulated as an ILP model in a small network. Meanwhile, a heuristic algorithm is also designed to solve the problem in a large size network. Simulation results show that the ILP model has an exact solution, and the heuristic algorithm can deploy more DMT services than benchmark.

II. Problem Description

An EC-EON can be presented as a directed graph $G(V, E)$. V and E represent the sets of nodes and fiber links, respectively. A DMT task is denoted as $\{R_r(D_r, s_r, \theta_r, T_r)\}$, where r is the task index and D_r is the size of training data. s_r represents the source node where the training task is generated. θ_r denotes the model accuracy, i.e., the ratio of loss function gradients between two adjacent iterations. With a smaller θ_r, more iterations are required to complete a training task. T_r is its latency tolerance. Figure 1 illustrates the data parallelism-based DMT in a six-node EC-EON, with one cloud node and five edge nodes.

Manuscript received November 18, 2020; accepted November 25, 2020. Date of publication December 1, 2020; date of current version April 9, 2021. This work has been supported in part by National Key Research and Development Program of China (2020YFB1805602) and the NSFC Projects (Grant No. 61901053, 61831003, 61822105). The associate editor coordinating the review of this letter and approving it for publication was X. Chu. *(Corresponding author: Yongli Zhao.)*

Yajie Li, Zebin Zeng, Boyuan Yan, Yongli Zhao, and Jie Zhang are with the School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: yonglizhao@bupt.edu.cn).

Jun Li is with the Department of Electrical Engineering, Chalmers University of Technology, 41296 Gothenburg, Sweden (e-mail: ljun@chalmers.se).

Digital Object Identifier 10.1109/LCOMM.2020.3041453

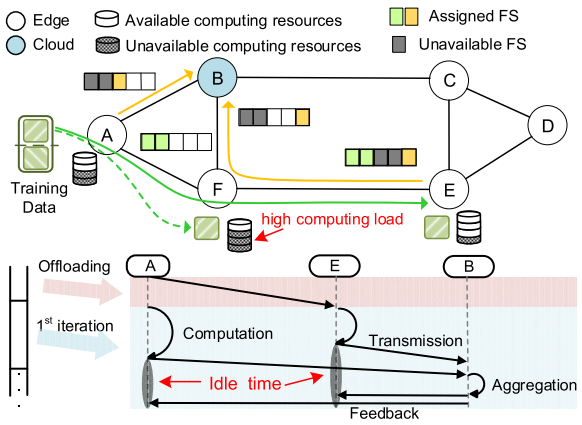

Fig. 1. Illustration of data parallelism-based DMT in EC-EONs.

A training task R_i is generated from node A, with a certain size of training data D_i. Half of training data is offloaded to edge node E to perform distributed training. Although being the nearest to node A, node F is suffering from the high computing load. We can see that two frequency slots (FSs) are required to support the offloading of training data. The cloud node is assumed to provide the complete model to be trained for each training node. Then, the model will be trained by node A and E through a number of iterations. The training iteration accuracy θ_i determines the general upper bound of iteration times [8]. The increase of participating nodes will shorten the time of each iteration, at the cost of higher usage of network resources. For the sake of clarity, we just take two nodes as example.

After the distribution of training data, node A and E independently train the model with its own training data in each iteration. The update of model parameters in node A and E are respectively uploaded to cloud node. One FS is required to transmit the update of model parameters, which is much smaller than the size of training data. The cloud node aggregates the updates from node A and E to generate global model for next iteration. Next iteration will start after the updated global model is sent to node A and E. Assume that the exchange of control information among edge and cloud nodes can be realized with a network controller. Thus, each iteration mainly consists of four stages, i.e., computation, transmission, aggregation and feedback. Since the downlink bandwidth is larger than uplink and the capacity of computing resources in cloud node is much higher than edge node, this letter does not consider the aggregation time in the cloud as well as the feedback time. Note that with fewer computing resources than node E, node A is the straggler of the training process. Consequently, node A limits the iteration time and leads to the waste of computing resources in node E. To evaluate the impact of the straggler on DMT, we calculate the ratio of average idle time to completion time (RIC). The smaller the RIC, the less impact the straggler has on the iteration time, which corresponds to a higher efficiency of distributed training.

Thus, the partition and distribution of training data directly affect the efficiency of DMT as well as the usage of computing and transport resources. It is significant to investigate how to efficiently provide DMT services by optimizing the partition and distribution of training data in EC-EONs.

III. ILP Formulation

This section formulates an ILP model to find the optimal solution to provide data parallelism-based DMT services in EC-EONs. Note that we focus on static deployment scenario where all the training tasks are known in advance.

A. Notations

- N : set of edge nodes.
- U : set of cloud nodes.
- L : a large number.
- W : number of FSs on each fiber link.
- B : capacity of an FS.
- b_r : number of FSs allocated for $R_r$to offload data.
- m : modulation level of each link.
- τ : propagation delay of each link.
- σ : background traffic metric of each link.
- ω : updated data size of each iteration.
- n : number of participating nodes for each task.
- ϵ : percentage of training data deployed on source node.
- η : required computing resources per bit in each iteration.
- $P^{x,y,k}$: path set that contains K shortest paths from x to y, where $x, y \in V$and $k = 1, 2, \ldots, K$.
- M_i : maximum number of tasks deployed on edge node i.
- C : total computing resources in each edge node.

B. Variables

- O_r^i : boolean variable that equals 1 if training task $r$is deployed on edge node i.
- d_r^i : integer variable that indicates the training data size of request r deployed on edge node i.
- $Z_r^{x,y,k,l,w}$: boolean variable that equals 1 if the w^{th} FS on the link l of k^{th} shortest path from x to y is used by request r, and 0 otherwise.
- $S_r^{x,y,k}$: boolean variable that equals 1 if the k^{th} shortest path from x to y is used by request r, and 0 otherwise.

C. Objective

The objective is to minimize the utilization of transport and computing resources, as

$$Minimize\ \alpha \times \sum_{\substack{\forall r,\forall x,\forall y,\\ \forall k,\forall l,\forall w}} Z_r^{x,y,k,l,w}/(W \times |E|) + \beta \times \sum_{\forall r,\forall i} (O_r^i/M_i)/|N| \quad (1)$$

where α and β are weight factors to adjust the importance of the two terms, and can be set based on the specific situation.

$$K(\theta_r) \times \max_{i \in N}(T_{com_r}^i + T_{tran_r}^i) + \max_{i \in N} T_{off_r}^i \leq T_r \quad (2)$$

$$T_{com_r}^i = \eta \times d_r^i \times M_i / C \quad (3)$$

$$T_{tran_r}^i = (\omega \times O_r^i/(B \times m) + \tau \times \sum_{\forall y \in U,\forall k,\forall l,\forall w} Z_r^{i,y,k,l,w}) \times \sigma \quad (4)$$

$$T_{off_r}^i = (d_r^i/(B \times m \times b_r) + \tau \times \sum_{\forall x \in s_r,\forall k,\forall l,\forall w} Z_r^{x,i,k,l,w}/b_r) \times \sigma \quad (5)$$

Eq. (2) ensures that each task is completed within a provided latency tolerance. $K(\theta_r)$ is the general upper bound on

training iterations. Eqs. (3)-(4) show the calculation of computation and transmission time, respectively. Eq. (5) is the offloading time and $T^i_{off_r} = 0$ for the source node. The background traffic is assumed to share the transport resources with the training tasks.

D. Constraints

1) Task Compliment Constraints:

$$O^i_r = \begin{cases} 0 & d^i_r = 0 \\ 1 & d^i_r \neq 0 \end{cases}, \forall i, r, \tag{6}$$

$$\sum_{\forall i} O^i_r = n \quad \forall r \tag{7}$$

$$d^i_r \geq \epsilon \times D_r \quad \forall r, \ \forall i \in s_r, \tag{8}$$

$$\sum_{\forall i} d^i_r = D_r \quad \forall r, \tag{9}$$

$$d^i_r = 0 \quad \forall r, \ \forall i \in U, \tag{10}$$

$$\sum_{\forall r} O^i_r \leq M_i \quad \forall i. \tag{11}$$

Eqs. (6)-(7) ensure that each training task is deployed on n edge nodes. Eq. (8) means that part of training data is deployed on the source node. The training data should not be deployed on the cloud according to Eqs. (9)-(10). Eq. (11) imposes a constraint on the number of tasks deployed on each edge node.

2) Resources Constraints:

$$\sum_{\forall k} S^{x,y,k}_r = \begin{cases} O^y_r & x \in s_r, x \neq y, y \notin U \\ O^x_r & y \in U \\ 0 & else \end{cases} \times \forall r, \tag{12}$$

$$\sum_{\forall w} Z^{x,y,k,l,w}_r = \begin{cases} S^{x,y,k}_r \times b_r \\ \quad \forall x \in s_r, y \notin U, \forall l \in P^{x,y,k} \\ S^{x,y,k}_r \\ \quad \forall x, y \in U, \forall l \in P^{x,y,k} \\ 0 \quad else \end{cases} \times \forall r, \quad \forall k, \tag{13}$$

$$\sum_{\forall r, \forall x, \forall y, \forall k} Z^{x,y,k,l,w}_r \leq 1 \quad \forall w, \forall l, \tag{14}$$

$$Z^{x,y,k,l,w}_r = Z^{x,y,k,l',w}_r \quad \forall r, \ \forall x, \ \forall y, \ \forall k, \ \forall w, \ \forall l, \ l' \in P^{x,y,k} : l \neq l', \tag{15}$$

$$(Z^{x,y,k,l,w}_r - Z^{x,y,k,l,w+1}_r - 1) \times (-L) \geq \sum_{\forall u} Z^{x,y,k,l,u}_r \quad \forall r, \quad \forall x, \forall y, \ \forall k, \ \forall l \in P^{x,y,k}, \quad \forall w, \ u \geq w + 2. \tag{16}$$

Eq. (12) ensures only one candidate path will be selected, and Eq. (13) allocates transport resources on the links of the selected path. Specifically, the offloading bandwidth is determined by the training data size of each task, and one FS is required to upload the updated model parameters to cloud. The spectrum non-overlapping, continuity and contiguity constraints are expressed by Eqs. (14)-(16), respectively.

IV. Heuristic Algorithm

This section proposes a data parallelism deployment algorithm (DPDA) to provide data parallelism-based DMT services in a larger network topology. We refer to the factor in [10] to express the usage of network resources for each task.

$$\varphi(i) = \varphi_b(s_r, e_i) + \varphi_b(e_i, u) + \varphi_c(i) \tag{17}$$

$$\varphi_b(n_1, n_2) = \sum_{\forall k} h^k_{n_1,n_2}/K \times b/(\sum_{\forall k} h^k_{n_1,n_2} v^k_{n_1,n_2} / \sum_{\forall k} h^k_{n_1,n_2}) \tag{18}$$

Eqs. (17)-(18) calculate the network occupation factor. e_i is the offloading edge node and u is the cloud node. $\varphi_c(i)$ is the ratio of required computing resources to the available computing resources in e_i. Eq. (18) is the product of two parts. The first part is the average length of k-shortest paths from n_1 to n_2, and $h^k_{n_1,n_2}$ represents the length of k^{th} path. The second part is the ratio of required FS number b to average free FSs on each link, and $v^k_{n_1,n_2}$ denotes the free FSs number on path k.

Algorithm Data Parallelism Deployment Algorithm

```
for each training task R_r(D_r, s_r, θ_r) do
  get the set of offloading edge nodes E_N^r
  for each available edge node e_i ∈ E_N^r do
    calculate φ(i) according to Eq. (17);
  end
  sort all the nodes in E_N^r in ascending order of φ(i);
end
for ∀e_i ∈ s_r ∪ E_N^r do
  search spectrum and calculation resource;
  if have enough resources
    add e_i to the offloading nodes set F_N^r;
    perform routing and spectrum allocation;
    if |F_N^r| = n
      deploy training data according to the resources;
      if completion time ≤ T_r
        break;
    else
      add the next offloading node to F_N^r recursively;
    remove e_i from F_N^r;
end
if |F_N^r| < n
  R_r(D_r, B_r, θ_r) is blocked;
end
```

The DPDA jointly considers the computing and transport resources to optimize the partition and distribution of training data. Firstly, calculate the resource occupation factor for each candidate node and add the available edge nodes to E^r_N in ascending order of $\varphi(i)$. Secondly, select the required edge nodes from E^r_N including the source node. For each node, DPDA will search the required resources on the K-shortest paths to perform routing and spectrum allocation. The training data is partitioned in proportion to the provided computing resources. If latency constraint is not satisfied, the DPDA will release the reserved resources and check the remaining nodes in E^r_N to perform the resource allocation described above. The complexity of the resource allocation algorithm is related to the network size, the number of participating nodes and the FS number per optical link.

V. Performance Evaluation

A six-node topology in Fig. 1 and NSFNET topology in Fig. 2 are used in performance evaluation. The DMT tasks are randomly generated at the edge nodes. The number of FSs required for offloading is 2 if $D_r \geq 80$, and 1 otherwise. Assume at least 30% of training data is allocated on the

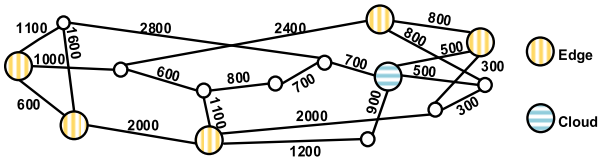

Fig. 2. NSFNET topology.

TABLE I
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
C (cycles/s)	300×10^9	W	15
M_i	[1,10]	n	2
ω (GB)	1	τ (ms)	0.2
B (GHz)	6.25	D_r (GB)	[50,100]
η (cycles/bit)	30	T_r (s)	$[5,7]\times10^3$
m	2	θ_r	[0.1,0.4]

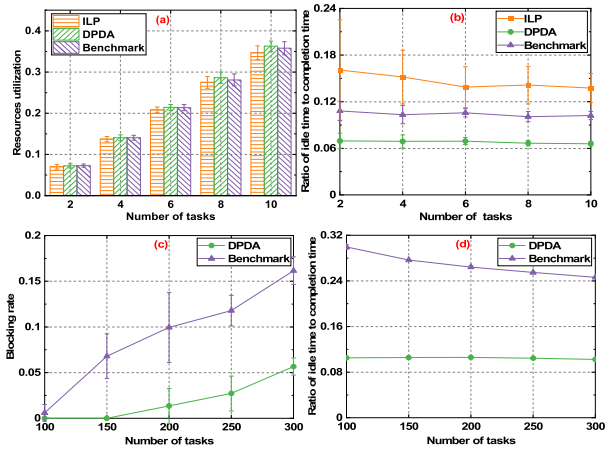

Fig. 3. Simulation results. (a) resources utilization and (b) RIC with six nodes topology; (c) blocking rate and (d) RIC with NSFNET topology.

source node. The number of training iterations can be simplified as $K(\theta_r) = 100 \times \log(1/\theta_r)$ [8]. $\alpha$and β in Eq. (1) are 0.5, which means the transport and computing resources are equally important. The other parameters are listed in Table I. The resource capacity in NSFNET is twenty times of six-node topology. The propagation delay is 0.5 us/km. The benchmark algorithm sorts the edge nodes in ascending order of distance from the source node. Then it traverses all nodes to select the required number of nodes that meet the latency constraint. Note that training data is equally distributed among all the participating nodes.

Fig. 3(a) shows the results of objective function in Eq. (1) for three schemes. The ILP can obtain the lowest value. The benchmark algorithm may obtain lower value than DPDA by selecting the closest nodes for offloading in six-node topology. Fig. 3(b) shows that DPDA achieves a lower RIC than the benchmark. It is interesting to note that ILP has a higher value in the RIC. The reason is that the objective function just minimizes the resource usage.

Fig. 3(c) depicts the blocking rate of training tasks in NSFNET. DPDA can deploy more training tasks with the same network resources. With the increase of tasks, the blocking rate of DPDA is reduced by about 10% compared with the benchmark. Fig. 3(d) shows the results of the RIC in NSFNET.

TABLE II
AVERAGE RUNNING TIME (SECONDS)

	Six nodes topology			NSFNET topology		
# of Tasks	2	6	10	100	200	300
ILP	1.35	6.97	17.10	-	-	-
DPDA	0.012	0.013	0.015	0.031	0.054	0.073
Benchmark	0.011	0.012	0.014	0.026	0.045	0.062

DPDA also has a lower RIC than the benchmark, which means that DPDA can effectively allocate computing resources and reduce the impact of stragglers. In addition, as the number of tasks increases, the RIC of benchmark will decrease while DPDA maintains at about 10%.

Table II shows the average running time of the algorithms. With six-node topology, ILP has the longest running time for its high complexity. As the number of tasks increases, the running time of ILP increases rapidly. When the number of tasks is higher than 100 in NSFNET, ILP cannot solve this problem with an acceptable running time, while the running time of DPDA and benchmark is relatively short. Meanwhile, the time cost of DPDA is slightly higher than the benchmark, due to the calculation of resource occupation factor to sort the available edge nodes.

VI. CONCLUSION

This letter investigates the efficient provisioning of DMT services by optimizing the partition and distribution of training data in EC-EONs. The problem is solved by an ILP model as well as a heuristic algorithm. The simulation results show that the proposed algorithms can effectively allocate resources for DMT services and deploy more DMT services than benchmark.

REFERENCES

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proc. IEEE*, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.
[2] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan. 2018.
[3] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol. (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.
[4] S. A. Osia *et al.*, "A hybrid deep learning architecture for privacy-preserving mobile analytics," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4505–4518, May 2020.
[5] Q. Ho *et al.*, "More effective distributed ml via a stale synchronous parallel parameter server," in *Proc. Adv. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2013, pp. 1223–1231.
[6] Y. Lin, S. Han, H. Mao, Y. Wang, and W. J. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," 2017, *arXiv:1712.01887*. [Online]. Available: http://arxiv.org/abs/1712.01887
[7] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016, *arXiv:1610.05492*. [Online]. Available: http://arxiv.org/abs/1610.05492
[8] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Apr. 2019, pp. 1387–1395.
[9] M. Liu, Y. Li, Y. Zhao, H. Yang, and J. Zhang, "Adaptive DNN model partition and deployment in edge computing-enabled metro optical interconnection network," in *Proc. Opt. Fiber Commun. Conf. (OFC)*, 2020, p. Th2A.28.
[10] Y. Li *et al.*, "Joint balancing of IT and spectrum resources for selecting virtualized network function in inter-datacenter elastic optical networks," *Opt. Exp.*, vol. 27, no. 11, pp. 15116–15128, 2019.