Efficient Data Communication for Deep Learning Application via Latent Code Transmission in B5G Wireless Networks

Dong Min Kim[1,2,*], Young Hoon Suh[1], Issamu Fred Ngwalo[2]
[1]Department of Internet of Things, Soonchunhyang University, South Korea
[2]Department of ICT Convergence, Soonchunhyang University, South Korea
[*]Corresponding author: Dong Min Kim (dmk@sch.ac.kr)

***Abstract*—In B5G networks, we introduce an innovative data transmission strategy, latent code transmission, tailored for applications that gather data from distributed devices and run deep learning algorithms on a centralized server. The essence of our approach is its adaptability to varying wireless channel conditions: it sends comprehensive original data during optimal channel conditions and, when conditions are suboptimal, processes the preliminary stages of the deep learning model at the device end, thereby shrinking the data size before transmission. The distinct advantage of our method is its ability to naturally reduce data size through the deep learning pipeline, obviating the need for an additional compression/decompression phase. This results in minimal extra computational demands compared to locally executing the deep learning model.**

***Index Terms*—B5G network, Latent Code Transmission, Deep Learning Application.**

I. Introduction

The burgeoning interest in autonomous driving technology is evident in today's era. The automotive industry, paralleling the evolution of electric vehicles, is heavily investing in the research and development of autonomous driving technology, viewing it as a catalyst for mobility innovation [1]. However, transitioning to autonomous vehicles is not as simple as merely upgrading the current vehicles within the existing infrastructure. Enhancing safety would involve acquiring more information through communication with Road Side Units (RSUs) or other vehicles. In this context, the integration of the Cellular Vehicle-to-Everything (C-V2X) technology can foster the development of a connected intelligence-based autonomous driving system, marking a genuine stride towards mobility transformation [2].

Meanwhile, the manufacturing and logistics sectors are showing an intensified interest in robotic innovations [3]. Several countries face challenges in recruiting workforce in these sectors. The surge in labor costs, a decline in the working-age population, and a societal inclination to avoid hazardous working environments contribute to these challenges. Thus, the introduction of robots to enhance productivity and efficiency is garnering attention. Similar challenges observed in the autonomous driving sector emerge here again. No matter how advanced an individual robot becomes, building a purely robotic system without environmental interactions is exceedingly challenging. Constructing collective intelligence through information exchange between robots or with their environments might be the solution. Depending on the situation, specialized networking systems like private 5G can be employed [4].

In the past decade, the rapid progression of deep learning technologies has spurred attempts to integrate deep learning into various sectors. Deep learning with Internet-of-Things (IoT), in particular, has risen as a research topic of significant interest in recent years [5]. Advances in data communication and processing capabilities have made it feasible to gather and share data across multiple devices. Unlike the traditional centralized deep learning models, where a central server processes all the data, distributed deep learning leverages data scattered across numerous devices. This approach offers several advantages: 1) by allowing devices to learn directly from their data, it cuts down on the time and cost associated with central data transmissions. 2) processing data on-site can minimize risks related to data breaches and privacy concerns. However, distributed deep learning comes with its challenges, such as the complex issue of synchronizing models being trained simultaneously on different devices. Variabilities in computational capacities and storage across devices can also pose difficulties in ensuring a uniform learning environment. Moreover, real-world applications need to consider challenges like communication delays between devices, data imbalances, and data quality variations. Researchers are developing diverse techniques, including efficient communication mechanisms, model synchronization strategies, and data quality adjustment methods, to address these issues [6]–[8]. Recently, the research focus has shifted beyond simple distributed learning methodologies, emphasizing applicability and efficiency in real-world scenarios. Notably, the convergence of emerging technologies like edge computing, beyond 5G (B5G), and 6G is further enhancing the performance and efficiency of applications with deep learning. There are applications mandatorily requiring consistent and real-time B5G wireless communication:

- **Disaster Response Drones**: In the event of natural disasters or accidents in specific areas, drones capture the on-site situation and transmit it in real-time to rescue teams. These drones then employ deep learning to analyze optimal rescue or evacuation routes.

- **Remote Oceanic Observation Stations**: Stations located deep within the ocean send collected data to research centers via satellite communication. This data is used for monitoring marine ecosystems or changes in sea temperatures.
- **Unmanned Vehicles (e.g., UAVs or underwater robots)**: Deployed for exploration or monitoring purposes in specific areas, they continuously transmit data to a central server or control center, depending on their location or environmental conditions.
- **Smart City Infrastructure**: Data collected from sensors and cameras throughout the city are wirelessly transmitted in real-time to a central data center. This data is then utilized for optimizing tasks like traffic management, energy consumption, and safety monitoring.
- **Real-time Location-based Services**: These services track the user's current location and movements in real-time. Based on this information, they offer services like nearby points of interest, warnings of dangerous areas, or route recommendations.

In the context of B5G networks, we propose an efficient data transmission technique designed for applications that receive data from distributed devices and execute deep learning algorithms on a server. Our proposed method adapts to wireless channel conditions: when the channel state is favorable, it transmits the original data with extensive content; conversely, in unfavorable conditions, it initiates the initial layers of the deep learning model on the device side, reducing the data size prior to transmission and the latent code is transmitted. A key advantage of our approach is that, instead of adding a separate compression/decompression process to decrease data size, it leverages aspects of the deep learning process where size reduction naturally occurs. This not only eliminates the need for additional processes but also ensures minimal additional computational overhead compared to executing the deep learning model locally.

II. Related Work

In the work by Sun et al. [9], the authors addressed the optimization problem of wireless resource management using Deep Neural Networks (DNN). They proposed a learning-based approach to facilitate real-time resource allocation by approximating the complex mapping between the inputs and outputs of resource allocation algorithms. Our study, in contrast, centers on formulating data transmission strategies by repurposing the architecture of pre-existing deep learning models to adapt to real-time channel changes, aiming to utilize the given wireless network environment efficiently.

Le et al. [10] underscored the pivotal role of accurate channel estimation in wireless networks and introduced a novel approach using machine learning. By leveraging deep learning techniques, they surpassed traditional methods in enhancing channel estimation accuracy. Our research, focused on dynamic data transmission adaptation, capitalizes on the merits of pre-existing deep learning techniques for specific applications. We can assert its progressive nature in recycling these techniques for improving wireless communication strategies, thereby reducing the overhead of additional implementation or execution.

Patnaik et al. [11] expanded on the capabilities of neural networks and explored their diverse applications in wireless communication engineering. Their paper highlights the broader potential of neural networks in the optimization of wireless communication systems, emphasizing the transition from extended analysis phases to rapid product development. While our emphasis is on adjusting data transmission strategies, their work elucidates the extensive utility of neural networks in the wireless domain.

III. Proposed Scheme

A. System Model and Problem Statement

Consider a network-based deep learning application. The server designed for this application receives data from remote data acquisition or sensing equipment. Equipment in remote locations transmits data via the B5G network. It is presumed that the wireless channel fluctuates over time, and the B5G network is capable of estimating these changes. Regarding the deep learning model, it is assumed that when data passes through the input layer, certain layers may experience a reduction in the overall data size of the latent code. Indeed, many existing deep learning models exhibit this characteristic. Additionally, we only consider models that have already been trained. Training processes, such as backpropagation, are outside the scope of this study, and we focus on the inference procedure.

Conventional data transmission systems predominantly utilize static server-client architectures, which can result in suboptimal performance under dynamic and bandwidth-limited conditions. In such contexts, the primary challenge lies in efficiently transmitting substantial data volumes with minimized latency, given the variable quality of the communication channel. In order to make the data size suitable for transmission, additional compression and decompression processes were required. We want to investigate a method that produces the same performance while excluding additional processes as much as possible by utilizing the compressibility of the latent code of the deep learning model.

B. Latent Code Transmission

The proposed system involves an IoT device equipped with onboard processing capabilities and the ability to interact with a central server. The device and server collaboratively make decisions on data transmission strategies based on the real-time assessment of the network conditions. When the network channel is optimal, the IoT device serves as a direct pathway for data to travel through, facilitating rapid data transfer between endpoints. Meanwhile, when the network quality deteriorates, latent code transmission comes into play. The rationale is to enhance adaptability by enabling the system to deal with varying network conditions more effectively. The process of latent code transmission is as follows.

1) Channel State Information (CSI) Acquisition: In a B5G network, the User Equipment (UE), which in this context is the IoT device, transmits its channel state information through the PUCCH (Physical Uplink Control Channel) or PUSCH (Physical Uplink Shared Channel). In response, the base station (either eNB or gNB) provides feedback to the device using the PDCCH (Physical Downlink Control Channel).
2) Adaptive Modulation and Coding (AMC): Based on the CSI, both the IoT device and the base station decide on an appropriate modulation and coding scheme for transmissions over PDCCH and PUSCH. In good channel conditions, a higher rate modulation scheme can be used, while in poor channel conditions, a more robust modulation and coding scheme is chosen to reduce the probability of errors.
3) Latent code determination: Depending on the determined data rate, the amount of data that can be transmitted during the limited time budget required by the given application is determined, and the deep learning model is advanced to the level required locally to obtain a latent code that can transmit this amount of data. At this time, the payload is added because the server needs to be informed of which layer the data to be transmitted is, but the amount is very small.
4) Transmission Power Control: The IoT device adjusts its transmission power for signals sent over the PUCCH and PUSCH. This adjustment is based on the feedback received from the base station over the PDCCH.

Overall procedure is illustrated in Fig. 1. Our approach is designed to address the challenges posed by dynamic network conditions in B5G communication environments. To achieve efficient and low-latency data transmission, we propose a novel algorithm that combines adaptive neural network partitioning, localized processing, and distributed computation. Leveraging real-time channel information, our algorithm dynamically adjusts the data transmission strategy based on network quality. Since the device has onboard processing capabilities, we consider splitting the deep learning model into two parts: one part that runs directly on the device and another part that runs on the central server. This way, the device can handle initial data processing and feature extraction locally, reducing the amount of data that needs to be transmitted. The central server can then focus on higher-level decision-making.

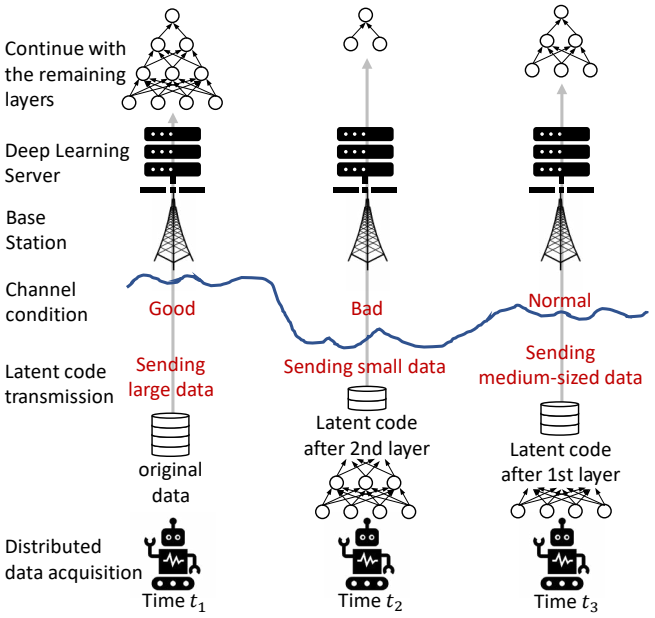

Fig. 1: Diagram of Proposed Scheme.

TABLE I: Model parameters.

Layer	Output Dimension	Data Size (MByte)	Latent Code Transmission
Original	896×896×3	2.296875	Tx. in good channel
Preprocess	224×224×3	-	
Conv1	224×224×64	-	
Conv2	224×224×64	-	
Pool	112×112×64	3.0625	
Conv3	112×112×128	-	
Conv4	112×112×128	-	
Pool	56×56×128	1.53125	Tx. in normal channel
Conv5	56×56×256	-	
Conv6	56×56×256	-	
Conv7	56×56×256	-	
Pool	28×28×256	0.765625	
Conv8	28×28×512	-	
Conv9	28×28×512	-	
Conv10	28×28×512	-	
Pool	14×14×512	0.3828125	Tx. in bad channel
Conv11	14×14×512	-	
Conv12	14×14×512	-	
Conv13	14×14×512	-	
Pool	7×7×512	0.095703125	
FC1	4096	-	
FC2	4096	-	
FC3	1000	-	

IV. Experiment Results

In our study, we executed an experiment to validate the efficacy of the Latent Code Transmission approach. We utilized a mobile robot, equipped with a camera, that continuously transmitted the captured images to a designated server using a 5G network. The images captured by the camera had a resolution of 896×896×3 in the RGB format. Upon receiving, the server processed these images with the pretrained VGG model [12] to conduct image classification. For our experiment, the VGG-16 is adjusted to accept an input size of 896×896×3. Following the preprocessing layer, several convolutional layers are sequentially aligned as described in Table I.

We categorized the network's status into three levels—good, normal, and bad—based on the Reference Signal Received Power (RSRP) and Reference Signal Received Quality (RSRQ) values. In optimal channel conditions, the camera's captured image was relayed unaltered, resulting in a transmission of approximately 2.296875 MB per image. Under normal channel conditions, the robot transmitted the latent code after processing up to the 4th convolutional and pooling layer, leading to a transmission of around 1.53125 MB per

Algorithm 1 Channel-aware Latent Code Transmission

Constants:
    $rsrpGoodThreshold = -80$
    $rsrpNormalThreshold = -100$
    $rsrqGoodThreshold = -10$
    $rsrqNormalThreshold = -20$

Get RSRP and RSRQ values
if $RSRP \geq rsrpGoodThreshold$ **then**
    $rsrpCondition \leftarrow$ `Good`
else if $RSRP \geq rsrpNormalThreshold$ **then**
    $rsrpCondition \leftarrow$ `Normal`
else
    $rsrpCondition \leftarrow$ `Bad`
end if
if $RSRQ \geq rsrqGoodThreshold$ **then**
    $rsrqCondition \leftarrow$ `Good`
else if $RSRQ \geq rsrqNormalThreshold$ **then**
    $rsrqCondition \leftarrow$ `Normal`
else
    $rsrqCondition \leftarrow$ `Bad`
end if
if *rsrpCondition* == `Bad` **or** *rsrqCondition* == `Bad` **then**
    Set *channelCondition* to `Bad`
else if *rsrpCondition* == `Normal` **or** *rsrqCondition* == `Normal` **then**
    Set *channelCondition* to `Normal`
else
    Set *channelCondition* to `Good`
end if
Call LatentCodeTransmissionMode(*channelCondition*)

image—this constitutes just 66.7% of the raw data.[1] Conversely, in poor channel conditions, processing was extended up to the 10th convolutional and pooling layer prior to latent code transmission. Consequently, only about 0.3828 MB was transmitted per image, which amounts to a mere 16.7% of the raw data. Algorithm 1 describes a technique for transmitting the appropriate Latent Code based on the perceived channel condition.

A pre-trained model is installed in advance on both the server and the robot. Depending on the channel conditions, the robot processes up to a specific layer and then transmits the generated latent code. Through our experiments, we achieved a top-5 accuracy performance of 88.94%. When processing the same image on the server compared to locally, the performance exhibited a top-5 accuracy of 90.38%. Additionally, through Latent Code Transmission, we reduced the data transmission volume by approximately 30%. We confirmed that the scheme we proposed operates effectively under varying channel conditions. Executing portions of the deep learning model within the robot, based on the channel state, offers the advantage of diminishing the data size for transmission. However, this approach utilizes more of the robot's energy. Given that the robot's computing capacity is inferior to that of the server, the execution time of the deep learning model could be extended. Consequently, it is not always advantageous to process the deep learning model through deeper layers, even if it results in a reduction in data size.

[1]We assumed that each element of the latent code tensor utilizes the float32 data type.

V. Conclusion

Within the context of B5G networks, our research introduced a novel data transmission approach, specifically tailored for applications that harnessed data from geographically dispersed devices to execute deep learning algorithms at a central server. Central to the methodology we developed was its inherent adaptability to fluctuating wireless channel conditions. We transmitted exhaustive original data during optimal channel conditions and minimized data size by initiating preliminary deep learning processing during suboptimal conditions, optimizing data transfer, i.e., latent code transmission. The most distinguishing feature of our proposed technique was its seamless integration of data size reduction within the deep learning process, which eliminated the need for separate compression/decompression steps. Furthermore, our empirical evaluations indicated only marginal additional computational loads compared to standard local deep learning model implementations. This research has provided a foundation for enhancing efficiency in data-driven applications within the burgeoning B5G ecosystem.

This research can be interpreted as an integration of edge computing and deep learning. For a more equitable performance assessment, it is imperative to consider processing delay. Specifically, servers with superior processing capabilities will likely experience reduced processing delays, while mobile robots with comparatively inferior processing performance might face increased delays. Incorporating performance degradation, attributed to on-robot processing for latent code transmission, into the evaluation will enable a more balanced assessment of the system's efficacy. This is one of our potential future research directions.

Acknowledgment

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2019R1G1A1100699). This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2022S1A5A2A03052880). This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the National Program for Excellence in SW, supervised by the IITP (Institute of Information & communications Technology Planning & Evaluation) in 2021 (2021-0-01399).

References

[1] J. Deichmann, E. Ebel, K. Heineke, R. Heuss, M. Kellner, and F. Steiner, "Autonomous driving's future: Convenient and connected." [Online]. Available: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/autonomous-drivings-future-convenient-and-connected

[2] J. Kim, Y.-J. Choi, G. Noh, and H. Chung, "On the feasibility of remote driving applications over mmwave 5G vehicular communications: Implementation and demonstration," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 2, pp. 2009–2023, 2022.

[3] J. Lee, P. C. Chua, L. Chen, P. H. N. Ng, Y. Kim, Q. Wu, S. Jeon, J. Jung, S. Chang, and S. K. Moon, "Key enabling technologies for smart factory in automotive industry: status and applications," *International Journal of Precision Engineering and Manufacturing*, vol. 1, no. 1, pp. 94–105, 2023.

[4] M. Wen, Q. Li, K. J. Kim, D. López-Pérez, O. A. Dobre, H. V. Poor, P. Popovski, and T. A. Tsiftsis, "Private 5G networks: Concepts, architectures, and research landscape," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 1, pp. 7–25, 2022.

[5] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the internet of things with edge computing," *IEEE network*, vol. 32, no. 1, pp. 96–101, 2018.

[6] M. U. A. Siddiqui, H. Abumarshoud, L. Bariah, S. Muhaidat, M. A. Imran, and L. Mohjazi, "URLLC in beyond 5G and 6G networks: An interference management perspective," *IEEE Access*, 2023.

[7] Y. Han, D. Niyato, C. Leung, C. Miao, and D. I. Kim, "A dynamic resource allocation framework for synchronizing metaverse with IoT service and data," in *Proc. IEEE International conference on Communications (ICC 2022)*. IEEE, 2022, pp. 1196–1201.

[8] Z. Xiong, W. Li, and Z. Cai, "Federated generative model on multi-source heterogeneous data in iot," in *Proc. the AAAI Conference on Artificial Intelligence 2023*, vol. 37, no. 9, 2023, pp. 10 537–10 545.

[9] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *Proc. IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2017)*, 2017.

[10] H. A. Le, T. Van Chien, T. H. Nguyen, H. Choo, and V. D. Nguyen, "Machine learning-based 5G-and-beyond channel estimation for MIMO-OFDM communication systems," *Sensors*, vol. 21, no. 14, 2021.

[11] A. Patnaik, D. E. Anagnostou, R. K. Mishra, C. G. Christodoulou, and J. C. Lyke, "Applications of neural networks in wireless communications," *IEEE Antennas and Propagation Magazine*, vol. 46, no. 3, pp. 130–137, 2004.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. the 3rd International Conference on Learning Representations (ICLR 2015)*.