2021 IEEE 32nd Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC) | 978-1-7281-7586-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/PIMRC50174.2021.9569707

Energy-Efficient Model Compression and Splitting for Collaborative Inference Over Time-Varying Channels

Mounssif Krouka, Anis Elgabli, Chaouki Ben Issaid and Mehdi Bennis
Centre for Wireless Communications (CWC), University of Oulu, 90014 Oulu, Finland,
Email: {mounssif.krouka, anis.elgabli, chaouki.benissaid, mehdi.bennis}@oulu.fi.

***Abstract*—Today's intelligent applications can achieve high performance accuracy using machine learning (ML) techniques, such as deep neural networks (DNNs). Traditionally, in a remote DNN inference problem, an edge device transmits raw data to a remote node that performs the inference task. However, this may incur high transmission energy costs and puts data privacy at risk. In this paper, we propose a technique to reduce the total energy bill at the edge device by utilizing model compression and time-varying model split between the edge and remote nodes. The time-varying representation accounts for time-varying channels and can significantly reduce the total energy at the edge device while maintaining high accuracy (low loss). We implement our approach in an image classification task using the MNIST dataset, and the system environment is simulated as a trajectory navigation scenario to emulate different channel conditions. Numerical simulations show that our proposed solution results in minimal energy consumption and CO_2 emission compared to the considered baselines while exhibiting robust performance across different channel conditions and bandwidth regime choices.**

***Index Terms*—Deep learning, remote inference, edge computing, energy efficiency, split learning, model compression.**

I. Introduction

Recent advances in artificial intelligence (AI) have greatly led to developing new avenues for intelligent applications such as self-driving cars, social media, and proactive healthcare management [1]–[3]. To achieve high accuracy, these applications require complex machine learning (ML) techniques such as deep neural networks (DNN) being the most widely used [4], [5].

The traditional way to perform remote ML inference is to let the node exposed to the input data (*client*) transmit its data to a remote inferring node (*server*), usually through a wireless link where the intensive DNN computations are performed. However, the large size of the data to be shared as well as the privacy constraint limit the feasibility of this approach in practice [6]. Recently, embedded devices equipped with more enhanced computation capabilities allow for more effective data processing at the network edge [7], [8]. However, for large-size models and under edge devices' energy constraints, the inference task may not be locally performed. One way to minimize the energy bill of the client while ensuring high inference accuracy at the server is to split the model between both nodes. In this paper, we propose a novel approach that addresses the problem of minimizing the total energy (local computation and transmission energies) by: (i) sparsifying the model while ensuring low loss (high accuracy) and (ii) introducing a time-varying model splitting strategy between the client and the server. Formally, we consider the problem of minimizing the total energy under time-varying wireless channels while maintaining high accuracy (low loss).

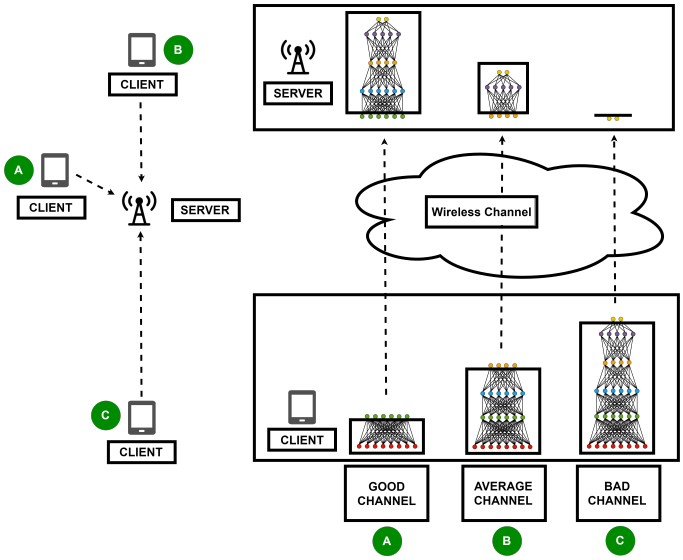

Fig. 1: Remote inference system model where the edge device (client) sends a time-varying communication-efficient data representation to the remote node (server).

To tackle this problem, some recent works [9]–[12] propose task offloading to reduce the latency and throughput where the DNN is split into two parts, one at the client and one at the server. For example, in [9], model splitting is performed based on per-layer energy consumption and latency, and the work in [10] suggests a technique to divide a DNN into multiple partitions based on a matching game theoretic approach. Authors in [13], [14] consider both model splitting and model *compression* in order to improve the device computation efficiency. *Compression* is performed by removing some weights/biases. In [13], authors propose a joint feature compression and transmission scheme for efficient inference tasks, for different network splitting points. The work in [14]

puts forward a 2-step pruning framework for DNN partitioning between mobile devices and edge servers. Nevertheless, none of aforementioned works combine model *compression* with the dynamic effect of the instantaneous time-varying channel gain on the DNN splitting strategy.

In this paper, we analyse different aspects of energy and attempt to minimize the total energy at the client-side by performing model *compression* in order to reduce the model size and time-varying model splitting between the client and the server to account for the trade-off between local computation and transmission energies over time-varying wireless channels. Note that the output of each layer in the DNN is considered as one representation for the input data; hence, as shown in Fig. 1, at any given time, we decide which representation to send such that the total computation energy to produce that representation plus the total communication energy of this specific representation is minimized for the given channel state. Our simulation results show that our proposed approach significantly reduces the total energy bill of the client as well as its corresponding CO_2 emission, and achieves better performance compared to several baselines under different channel gains and bandwidth regimes.

The rest of the paper is organized as follows. In section II, we describe the system model and the problem formulation. In section III, we describe our proposed solution. Before concluding, we present the simulations results and discussion in section IV.

II. System model and problem formulation

In this work, we study a system that comprises a client transmitting its data representation to a remote server for a machine learning (ML) inference task. The client's goal is to operationalize a trained ML model and send a communication-efficient representation to the server, as depicted in Fig. 1. Our objective is to minimize the client's energy bill over a time-varying wireless channel while maintaining high inference accuracy. In this work, we consider the computation energy (E_c), memory access energy (E_m), and transmission energy (E_{tr}). To formulate the problem, we first need to quantify the different energy aspects.

A. Computation Energy

The computation energy (E_c) is the required energy for performing arithmetic operations such as addition and multiplication. We denote by e_M and e_A the energy for a single multiplication and addition operations, respectively. In Table I, we report the energy values of these operations based on the benchmark [15]. As a consequence, the computation energy can be computed as follows

$$E_c = e_M \times M + e_A \times A, \tag{1}$$

where A and M are the required number of addition and multiplication operations, respectively.

B. Memory Access Energy

The memory access energy (E_m) is the energy utilized to fetch data from memory (cache or RAM) and move the necessary data to the arithmetic/logic unit. From Table II, we notice that the energy cost of the DRAM access is much higher than any computation operation. This is due to the required static power to keep the I/O active [15]. Therefore, the memory access energy is given as follows

$$E_m = \Gamma \times e_m, \tag{2}$$

where e_m is the energy for fetching/decoding one element from memory, and Γ is the needed number of elements (weights and biases in the case of DNN).

Arithmetic operation	ADD	MUL
8-bit Integer	0.03 pJ	0.2 pJ
16-bit Floating point	0.4 pJ	1.1 pJ
32-bit Integer	0.1 pJ	3.1 pJ
32-bit Floating point	0.9 pJ	3.7 pJ

TABLE I: Approximate energy costs for different arithmetic operations in 45nm 0.9V.

Memory size	64-bit Memory access
8KB	10 pJ
32KB	20 pJ
1 MB	100 pJ
DRAM	1.3-2.6 nJ

TABLE II: Memory access energy expenditure (consumption) in 45nm 0.9V.

C. Transmission Energy

The transmission energy (E_{tr}) is the energy used for transmitting the data representation to the server over the wireless channel. This energy depends on both the size of the transmitted representation as well as the channel condition. For the transmission energy, we consider that the output of each neuron in the cut layer (representation) is transmitted using 32 bits (full-precision communication). We let W and $h(t)$ be the bandwidth and the flat fading channel gain at time instant t, respectively. The channel model is generated according to a Rayleigh fading with zero mean and unit variance, i.e. $h \sim \mathcal{CN}(0, 1)$. Thus, according to Shannon's formula, the asymptotic transmission rate at time t is given by

$$R(t) = W \log_2 \left(1 + \frac{P(t)|h(t)|^2}{N_0 W}\right), \tag{3}$$

where N_0 and $P(t)$ are the power spectral density and the transmission power of the client, respectively.

The time τ required to transmit d elements should satisfy the following inequality

$$\int_{t=0}^{\tau} R(t)dt \geq 32d, \tag{4}$$

where τ and d are the uploading time and the number of transmitted elements, respectively. Hence, the transmission

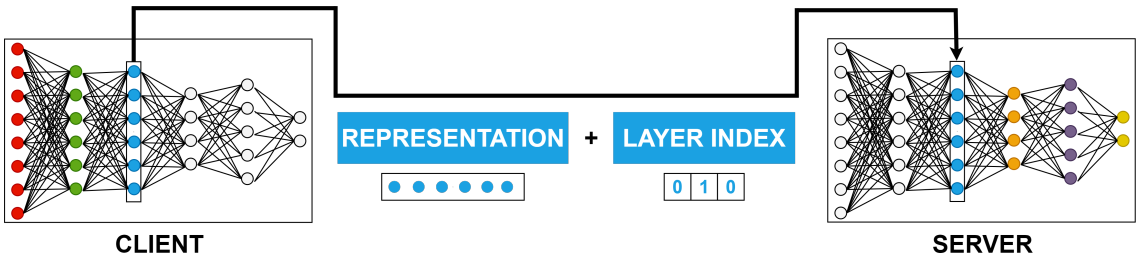

Fig. 2: Illustration of the proposed approach.

energy at the client side can be written as

$$E_{tr} = \tau \int_{t=0}^{\tau} P(t)dt. \tag{5}$$

Remark 1. *Throughout this paper, we assume that we have a perfect estimation of $h(t)$ for the period $t \in [0, \tau)$.*

As a consequence, the total energy budget of the client per inference can be expressed as follows

$$\begin{aligned} E &= E_c + E_m + E_{tr} \\ &= E_p + E_{tr}, \end{aligned} \tag{6}$$

where the processing energy E_p is the sum of the computation and memory access energies. In the remainder of this paper, we add the subscript i to the energy definition to account for the fact that the energy depends on the channel at the i^{th} inference time.

D. Problem Formulation

We aim to minimize the client's total energy bill while maintaining high accuracy. Therefore, we can formulate the following problem

$$\min_{\boldsymbol{\theta}_{c,i}, \boldsymbol{\theta}_{s,i}} \quad \lambda \cdot loss(\boldsymbol{\theta}_{c,i}, \boldsymbol{\theta}_{s,i}) + (1-\lambda) \cdot \lim_{\mathrm{I}\to\infty} \frac{1}{\mathrm{I}} \sum_{i=1}^{\mathrm{I}} E_i(\boldsymbol{\theta}_{c,i}), \tag{7}$$

where $loss$ is the training loss function, $\boldsymbol{\theta}_{c,i}$ and $\boldsymbol{\theta}_{s,i}$ are the client and server models at the i^{th} inference time, respectively, I is the total number of inferences, and $\lambda \in (0, 1)$ is a parameter that controls the trade-off between achieving low loss (high accuracy) and low energy consumption at the client's side.

Since the $loss$ function does not depend on the choice of the cut layer (transmitted representation), problem (7) can be re-written as

$$\min_{\boldsymbol{\theta}_{c,i}, \boldsymbol{\theta}_{s,i}} \quad \lambda \cdot loss(\boldsymbol{\theta}) + (1-\lambda) \cdot \lim_{\mathrm{I}\to\infty} \frac{1}{\mathrm{I}} \sum_{i=1}^{\mathrm{I}} E_i(\boldsymbol{\theta}_{c,i}), \tag{8}$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_{c,i}, \boldsymbol{\theta}_{s,i}]$ is the whole model, which is constant across different transmissions.

Problem (8) is hard to solve due to the lack of channel prediction for the whole period and non-convexity of the feasible set. In fact, the cut layer index belongs to a discrete set, which makes problem (8) a combinatorial one. Next, we propose an approximate and a low-complexity approach to solve problem (8).

III. Proposed Algorithm

The energy minimization problem formulated in (8) is controlled by two system aspects, namely the neural network (NN) complexity and the channel observation. We note that E_p depends solely on the number of weights and biases. This implies that lowering the model complexity reduces the processing energy. Accordingly, one solution to reduce the NN size is to perform model *compression* in the training phase.

On the other hand, the transmission energy depends on the size of the transmitted representation (number of neurons) as well as on the channel observation. Consequently, in addition to model complexity reduction, the proper selection of the layer over which the representation is transmitted contributes to minimizing the total energy. Therefore, our proposed solution mandates decomposing the original problem into two sub-problems: (i) model compression and (ii) representation selection.

A. Model Compression

Model *compression* leads to minimizing both the communication and computation energies since it sparsifies the model and also reduces the transmitted representation size. However, to sparsify the model while maintaining low loss/high accuracy, one can formulate the following sub-problem

$$\min_{\boldsymbol{\theta}} \quad \mu \cdot loss(\boldsymbol{\theta}) + (1-\mu) \cdot \sum_{l=1}^{L} \|\theta_l\|_0, \tag{9}$$

where L is the number of layers in the NN, θ_l is the model of the l^{th} layer, and μ is a parameter that controls the trade-off between model sparsification and the *loss* minimization. Note that problem (9) is hard to solve since the ℓ_0 norm minimization is a well-known NP-hard problem. Thus, instead of solving problem (9), we propose the following two-steps procedure

1) Solve the convex relaxation of the ℓ_0 norm minimization as follows

$$\min_{\boldsymbol{\theta}} \quad \mu \cdot loss(\boldsymbol{\theta}) + (1-\mu) \cdot \sum_{l=1}^{L} \|\theta_l\|_1. \tag{10}$$

2) Model *pruning* by eliminating all the parameters with values below a predefined pruning threshold.

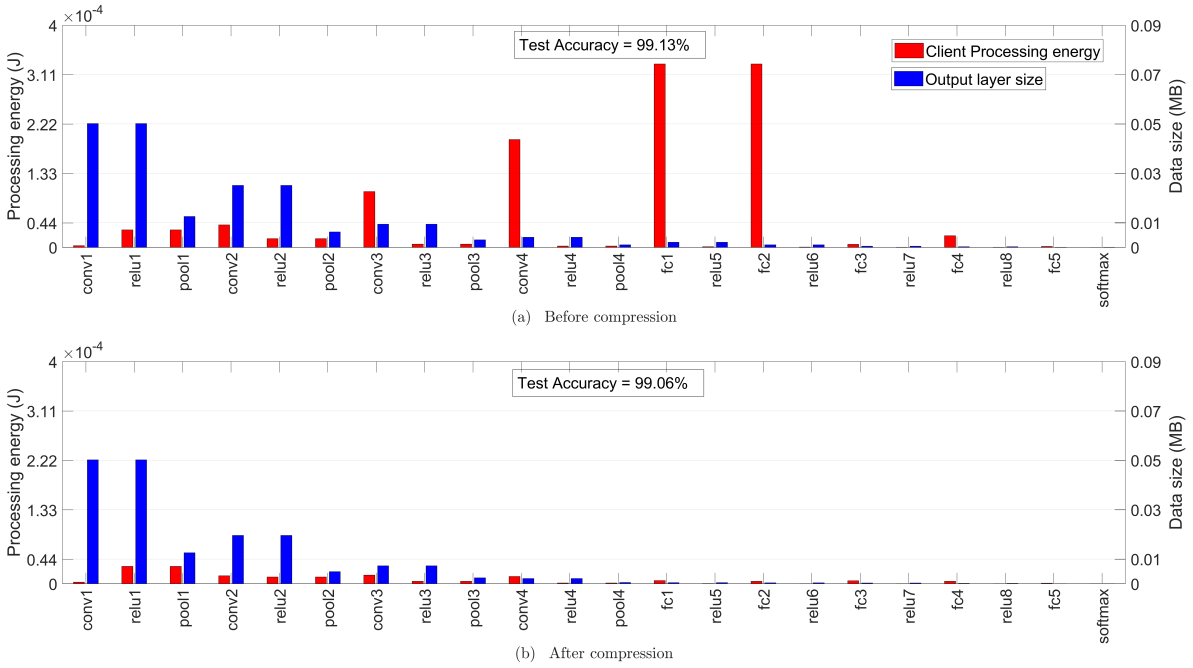

Fig. 3: Per layer processing energy and output size before and after compression.

Remark 2. *Note that problem* (10) *is solved once since its solution does not depend on the time-varying channel.*

Remark 3. *Practically,* μ *is chosen such that* $(1-\mu) << 1$. *This choice of* μ *can lead to model sparsification at a low cost in terms of accuracy drop.*

B. Representation Selection

Solving the sub-problem introduced in Sec. III-A results in a sparse version of the original model. Nevertheless, it does not provide information on which representation to send such that the communication energy is minimized. Selecting the appropriate representation, i.e., the output of which layer, to transmit is crucial in reducing the transmission energy for the given channel observation. The selection of the representation to be transmitted has a direct effect on the second term in problem (8), i.e, the total energy.

As depicted in Fig. 2, we consider that the remote server holds a copy of the client's model. When the client transmits a particular layer's output (representation), this layer's index is also shared, so the server can identify which layer to start from and carry on the inference task. Our proposed solution suggests that we can maintain a trade-off between the processing energy and the transmission energy by selecting the optimal layer $l \in \{1, \dots, L\}$ for a given channel observation. Formally, we choose the output of layer l at the i^{th} transmission, where l is the layer's index that minimizes $E_i(\boldsymbol{\theta}_{c,i})$.

IV. Simulation Results

For our simulations, we consider the image classification task where the MNIST dataset is used [16]. The dataset consists of a set of 28×28 gray-scale images with $60K$ for training and $10K$ for testing. The images represent handwritten digits ranging from 0 to 9. For our NN model, we use the convolutional NN (CNN) architecture. Our original model consists of 4 convolutional layers (*conv*) with a 5×5 filter and 16, 32, 48, and 64 channels, respectively. Each *conv* layer is followed by a *Relu* activation function and a pooling layer (*pool*) with *stride* 2 for size reduction. After that, we have 5 fully connected layers (*fc*) with 512, 256, 128, 64, and 10 neurons, respectively. The *Relu* activation function is applied to all *fc* layers expect for the last layer where *softmax* function is applied. We use $P = 1mW$, $N_0 = 10^{-9}$, batch size = 64, learning rate $\alpha = 10^{-3}$, $\mu = 0.9999$, and $W = 5MHz$.

In order to estimate the equivalent CO_2 emissions of the client, we use the data reported in [17], for different European Union (EU) countries in terms of carbon index (CI), where CI represents the greenhouse gas emission intensity calculated as kg of CO2 emissions per kWh ($kg\,CO_2e/kWh$) from public electricity production. In particular, we consider the average EU case with $CI = 0.275$.

Fig. 3 shows the processing energy and the output size for every layer in the NN before and after compression. We notice that the compression step significantly reduces the processing energy for more than 98% for both *fc1* and *fc2* layers which comprise around 60% of the NN parameters. Moreover, the decrease in the output size of the layers is reflected by eliminating entire neurons after pruning.

In Fig. 5, we plot the average total energy of the system and the testing accuracy for different values of the compression ratio C_r. The compression ratio is a measure of the relative

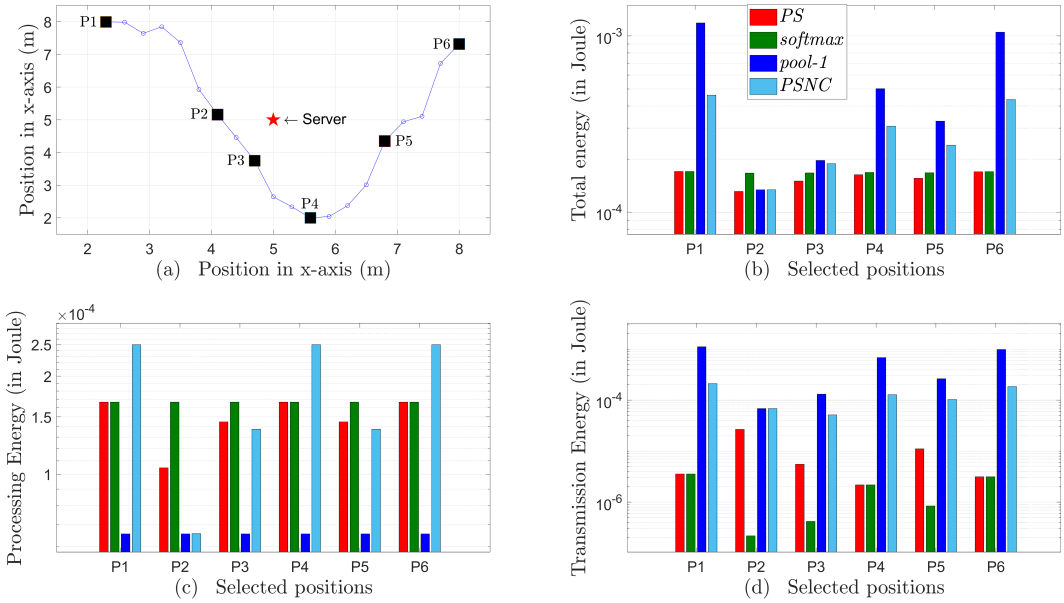

Fig. 4: Cut layer selection for different channel observations and its corresponding energy consumptions.

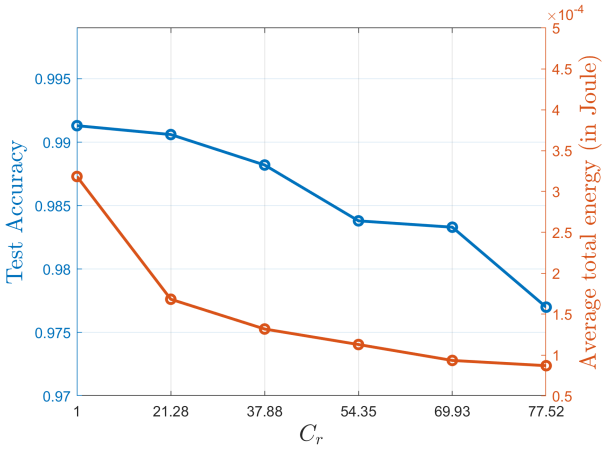

Fig. 5: Average total energy and test accuracy values for different compression ratios.

reduction in the number of model parameters defined as the ratio of the total number of non-zero parameters of the original model $\bar{\theta}^o$ with respect to the number of parameters of the compressed model $\bar{\theta}^c$. As the compression ratio increases, more energy expenditure can be saved, but at the cost of decreasing test accuracy. This is due to the fact that aggressive sparsification leads to eliminating more parameters which may decrease the accuracy.

Fig. 4 shows the performance of the proposed layer selection (*PS*) compared to the following baselines

- *softmax*: The client runs over all the model layers and transmits the labels.
- *Pool-1*: The client processes the data up to the first pooling layer. The pooling operation is used for dimensionality reduction of the feature maps.
- *PSNC*: The client selects the optimal layer for a given channel observation employing the original model (without compression).

Fig. 4(a) depicts a trajectory of the client in a 2-dimensional grid to simulate different channel conditions. In Fig. 4(b), we plot the total energy for the different selected locations. We notice that *PS* outperforms all the baselines. *pool-1* has the worst performance since the client is far from the server, and the transmit energy is the dominant part. On the other hand, when the channel is good (at P2), *softmax* performance is bad when processing energy is dominant. In Fig. 4(c), processing energy is plotted. We notice that *pool-1* performs better than all the other cases at the cost of consuming the highest transmission energy because of the large output size for *conv* layers, as seen in Fig. 4(d). The baseline *PSNC* uses up more processing energy seeing the high complexity of the original model. Fig. 4(d) illustrates the transmission energy results. We observe that *softmax* is always better than the other cases due to the small size of the labels, at the price of running through all the layers of the NN.

Table III contains different simulation results for different values of the bandwidth. We observe that *PS* outperforms all the other baselines in terms of total energy consumption and consequently CO_2 emission. For $W = 15KHz$, *softmax* performs slightly similar to *SP*. This is consequent to the high transmission energy at the low bandwidth regime. For $W = 15MHz$, the abundance of bandwidth enables the client to process few layers and transmit larger output sizes in order to save processing energy. This is observed by the

	POOL-1	SOFTMAX	PS	PSNC
		$W = 15KHz$		
$E\ (J)$	8.66×10^{-4}	1.69×10^{-4}	1.69×10^{-4}	4.36×10^{-4}
$E_p\ (J)$	0.66×10^{-4}	1.67×10^{-4}	1.64×10^{-4}	2.60×10^{-4}
$E_{tr}\ (J)$	7.98×10^{-4}	0.03×10^{-4}	0.05×10^{-4}	1.76×10^{-4}
$EU\ CO_2e\ (kg)$	0.66×10^{-10}	0.13×10^{-10}	0.13×10^{-10}	0.33×10^{-10}
		$W = 15MHz$		
$E\ (J)$	1.24×10^{-4}	1.67×10^{-4}	0.53×10^{-4}	0.61×10^{-4}
$E_p\ (J)$	0.66×10^{-4}	1.67×10^{-4}	0.23×10^{-4}	0.23×10^{-4}
$E_{tr}\ (J)$	0.58×10^{-4}	0.002×10^{-4}	0.31×10^{-4}	0.38×10^{-4}
$EU\ CO_2e\ (kg)$	0.1×10^{-10}	0.12×10^{-10}	0.04×10^{-10}	0.05×10^{-10}

TABLE III: Simulation results for different bandwidth values and baselines.

remarkable performance of *PSNC* since most NN parameters are positioned in the *fc* layers.

V. Conclusion

In this paper, we tackle the problem of collaborative inference where the goal is to reduce the total energy bill at the edge device by utilizing model compression and time-varying model split between the edge and remote nodes. Numerical results show that our proposed scheme outperforms the considered baselines in terms of the total energy consumption and CO_2 emission. In particular, our scheme maintains a robust performance for different channel conditions and bandwidth regime choices. For future work, we will extend our current system model to the multi-user case and investigate system latency and reliability under finite block-length regime.

References

[1] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[2] Y. Bounab, M. Oussalah, and A. Ferdenache, "Reconciling image captioning and user's comments for urban tourism," in *2020 Tenth International Conference on Image Processing Theory, Tools and Applications (IPTA)*. IEEE, 2020, pp. 1–6.

[3] M. K. Abdel-Aziz, C. Perfecto, S. Samarakoon, M. Bennis, and W. Saad, "Vehicular cooperative perception through action branching and federated reinforcement learning," *arXiv preprint arXiv:2012.03414*, 2020.

[4] F. Tariq, M. Khandaker, K.-K. Wong, M. Imran, M. Bennis, and M. Debbah, "A speculative study on 6g," 2019.

[5] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," 01 2013, pp. 1–9.

[6] A. Elgabli, J. Park, C. B. Issaid, and M. Bennis, "Harnessing wireless channels for scalable and privacy-preserving federated learning," *IEEE Transactions on Communications*, 2021.

[7] J. Liu, J. Liu, W. Du, and D. Li, "Performance analysis and characterization of training deep learning models on mobile devices," 2019.

[8] S. Zhang, Y. Liu, S. Li, Z. Tan, X. Zhao, and J. Zhou, "Fimpa: A fixed identity mapping prediction algorithm in edge computing environment," *IEEE Access*, vol. 8, pp. 17 356–17 365, 2020.

[9] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, and L. Tang, "Neurosurgeon: Collaborative intelligence between the cloud and mobile edge," *SIGARCH Comput. Archit. News*, vol. 45, no. 1, p. 615–629, Apr. 2017.

[10] T. Mohammed, C. Joe-Wong, R. Babbar, and M. D. Francesco, "Distributed inference acceleration with adaptive DNN partitioning and offloading," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, 2020, pp. 854–863.

[11] C. Hu, W. Bao, D. Wang, and F. Liu, "Dynamic adaptive DNN surgery for inference acceleration on the edge," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 1423–1431.

[12] L. Zhang, L. Chen, and J. Xu, "Autodidactic neurosurgeon: Collaborative deep inference for mobile edge intelligence via online learning," 2021.

[13] M. Jankowski, D. Gunduz, and K. Mikolajczyk, "Joint device-edge inference over wireless links with pruning," *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, May 2020.

[14] W. Shi, Y. Hou, S. Zhou, Z. Niu, Y. Zhang, and L. Geng, "Improving device-edge cooperative inference of deep learning via 2-step pruning," 2019.

[15] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2014, pp. 10–14.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[17] European Environment Agency, Data and maps, "Greenhouse gas emission intensity of electricity generation," Tech. Rep., Dec 2020.