Assessing the Performance of Analog Training for Transfer Learning

Omobayode Fagbohungbe*[‡], Corey Lammie[†], Malte J. Rasch*, Takashi Ando*, Tayfun Gokmen*, Vijay Narayanan*
*IBM Research - Yorktown Heights, NY USA [†]IBM Research Europe, 8803 Rüschlikon, Switzerland

[‡]Email: omobayode.fagbohungbe@ibm.com

Abstract—Analog in-memory computing is a next-generation computing paradigm that promises fast, parallel, and energyefficient deep learning training and transfer learning (TL). However, achieving this promise has remained elusive due to a lack of suitable training algorithms. Analog memory devices exhibit asymmetric and non-linear switching behavior in addition to device-to-device variation, meaning that most, if not all, of the current off-the-shelf training algorithms cannot achieve good training outcomes. Also, recently introduced algorithms have enjoyed limited attention, as they require bi-directionally switching devices of unrealistically high symmetry and precision and are highly sensitive. A new algorithm chopped TTv2 (c-TTv2), has been introduced, which leverages the chopped technique to address many of the challenges mentioned above. In this paper, we assess the performance of the c-TTv2 algorithm for analog TL using a Swin-ViT model on a subset of the CIFAR100 dataset. We also investigate the robustness of our algorithm to changes in some device specifications, including weight transfer noise, symmetry point skew, and symmetry point variability.

Index Terms—Deep Learning, Hardware Implemented Neural Network, Analog Device, Additive White Gaussian Noise

I. INTRODUCTION

Transfer Learning (TL) [1] is a training paradigm that leverages insights obtained from one task to accelerate the training of related but less complex tasks. As the process of fine-tuning a foundational model to many downstream tasks of choice is essentially TL [2], it plays a leading role in the wide application of Large Language Models (LLMs) to areas such as education [3], medicine [4], autonomous driving [5], and finance [6]. This process can significantly reduce the cost of training large models, as the foundation model can be re-used many times. Furthermore, TL makes it possible to train models with high accuracy, even with limited training data [7], [8]. These benefits have greatly improved the application of LLMs to tasks such as language modeling [7], [9]. Additionally, it also offers potential for model deployment at the edge particularly in scenarios where model training in the cloud is not possible due to data security and privacy concerns [10], [11].

However, training TL models using conventional hardware is very energy inefficient, particularly at the edge, where devices operate under a strict power budget [12]. This inefficiency is due to the von Neumann architecture, creating a physical separation between memory and computing units [13]–[15]. One significant consequence of this separation is a need for continuous data transfer, causing increased power

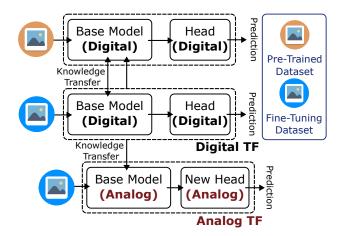


Figure 1: Key conceptual differences between digital TL and our analog TL implementation. We obtain the best results when the base model is converted from digital to analog and a new analog head is introduced.

consumption and processing time [16]. Analog In-Memory Computing (AIMC) [13], [17] is a new computing paradigm that represents weight stationary matrices in resistive crossbar arrays and leverages physical laws such as Kirchhoff's and Ohm's laws to compute matrix-vector multiplications inmemory which has been touted as a possible replacement for current Deep Learning (DL) acceleration hardware [13], [18].

Training Neural Networks (NNs) to achieve high-accuracy on AIMC-based hardware is challenging. First, the standard Stochastic gradient descent (SGD) training algorithm can not be used with AIMC as the computing devices exhibit asymmetric and non-ideal switching behavior [19], [20]. Moreover, the asymmetric nature of the memory device also makes gradient accumulation and update of weights represented in the conductances of the memory elements difficult [21]. Conducting some operations, like a full reset to a typical target conductance, is also costly, as the conductance can only be updated in small increments. Device materials also suffer from saturation to minimal and maximum conductance values. Lastly, the inherent presence of device-to-device variation means that an existing or new algorithm that assumes translational invariance cannot be used to train the model [20], [22].

Many efforts have been made to find an optimal algorithm for training NN models on AIMC-based hardware. The work in [23] proposed a mixed-precision algorithm that computes the gradient accumulation and weight update stages in digital and the forward and backward passes in analog, sacrificing speed and energy efficiency [20]. The authors in [24] proposed a fast and highly efficient parallel in-memory method to compute the outer-product and weight update operations, leveraging the coincidence of voltage pulse trains. However, this method requires a bi-directionally switching device of unrealistically high symmetry and precision, which is difficult to achieve in practice [24]-[26]. The use of the current asymmetric devices with realistic device-to-device variation for gradient accumulation causes devices to drift towards a different conductance value, even in the case when random fluctuations with zero mean are accumulated [20], [26]. To solve this problem, the authors in [25], [26] proposed various ideas to relax the symmetric and precision requirement, such as the use of separate non-volatile devices to accumulate the weight and gradients and represent pre-determined reference value, the introduction of a low-pass digital filtering stage, etc.

Despite statically correcting for device-to-device variations on the gradient accumulation and the update stages and achieving good time complexity, the improved training algorithm, known as TTv2, is very challenging to use in practice. The challenge stems from the sensitivity of the training algorithm to even slight changes from programmed reference values to their theoretical values – it is unlikely that the reference value remains unchanged over a long period of time [26], [27]. The authors in [20] proposed a method leveraging a chopped technique [28] to manage any offsets inflicted by an erroneous reference value in the gradient accumulation process by periodic or random sign changes. This new algorithm, called c-TTv2, achieved state-of-the-art results for analog training from scratch on toy models. The details about the c-TTv2 model can be obtained in [20].

However, the validation and demonstration of TL using the c-TTv2 algorithm has yet to be reported. Hence, this paper aims to demonstrate the use of the c-TTv2 algorithm for TL purposes for AIMC-based hardware using a medium-sized DL model. We also seek to validate the robustness of the c-TTv2 algorithm to changes in device specifications. Although hardware and software simulation of analog TL was performed in [12], the TTv2 algorithm algorithm was used. Furthermore, a 3-layer NN model and a 4-layer CNN-based model were used for the hardware demonstration and software simulation, respectively. To our knowledge, this is the first paper to simulate training of AIMC-based hardware for a Vision Transformer (ViT) model.

II. METHODOLOGY

In this section, we discuss our method used to validate the effectiveness of the c-TTv2 training algorithm for analog TL. Fig. 1 depicts the block diagram of digital and analog TL, highlighting the differences and similarities between the two approaches. The uniqueness of analog TL stems from using the analog-equivalent base model of the pre-trained model in the fine-tuning phase of the model training. The digital TL

differs from the analog TL in that both the pre-training and fine-tuning phases are done in digital.

Data augmentation was performed during the pre-training step. The softmax layer, the last layer of each model, reflects the number of classes in the pre-trained dataset. In preparation for the fine-tuning stage, the softmax layer of the pre-trained model was changed to reflect the number of classes of the fine-tuning task. Model training was performed for 100 epochs to achieve convergence. The weights and bias of the pre-trained model were then saved. For digital TL, the pre-trained model was then fine-tuned using the dataset selected for the fine-tuning phase for a further 100 epochs until convergence was achieved.

To perform analog TL, the modified pre-trained model was converted to its analog equivalent, and the fine-tuning process was performed. Due to the difference in precision between analog and digital hardware, it is impossible to program the digital weights to analog conductance values perfectly. To account for this, a weight transfer noise is modeled. This paper defines the weight transfer noise as an additive Gaussian noise of zero mean and unit variance. The mathematical equation defining the relationship between analog weight and pretrained weight is

$$W_{noise} = W_{pre_trained} + \tau \mathbb{N}(0, 1), \tag{1}$$

where W_{noise} is the analog weight, $W_{pre_trained}$ is the digital weight, τ is the noise factor, and $\mathbb{N}(0,1)$ is the Gaussian noise. The noise factor is used to increase or decrease the noise magnitude. A noise factor is a non-zero value, as a noise factor value of zero means that the weight transfer noise is not applied. It should be noted that the weight transfer noise is only added once before the start of the fine-tuning process. The fine-tuning process for the analog TL model was then repeated for different magnitudes of the weight transfer noise to measure its impact on the resulting analog model inference performance. Furthermore, analog and digital model training from scratch was also performed to establish baselines to validate the effectiveness of analog TL. These experiments are equivalent to performing the fine-tuning process without the pre-trained weights. To investigate the robustness of the c-TTv2 algorithm against selected device specifications, the value of the device specification of interest is increased, and the training process is repeated until convergence is achieved. The learning rate is also varied to select the model that achieves the best model inference performance with the varied device specifications.

We perform simulations using the Swin-ViT [29] model, which has six blocks and 26,591,906 parameters and 26,594,213 parameters for the 2-class task model and 5-class task model respectively. The Swin-ViT [29] model belongs to a class of ViT [30], [31], which is a class of attention-based architecture models that are widely used in Natural Language Processing (NLP) due to their computational efficiency and scalability. ViT operates by splitting an image into patches and providing the sequence of linear embeddings of these patches (equivalent to tokens) to the attention layers of the transformer.

Table I: Selected hyperparameters of the c-TTv2 algorithm for Analog Training and Stochastic Gradient descent Algorithm for Digital Training.

Hyperparameter Description	Value
Digital Training	
Learning rate	0.01
Batch size	8
Analog Tra	ining
Learning rate	0.01
Batch size	8
Transfer_every	1.0
Autogranularity	10000
Device momentum	0
Auto_scale	True
In_chop Probability	0.10
Units in mbatch	False
Forget Buffer	True
Auto_Momentum	0.99

The uniqueness of the Swin-ViT model stems from the usage of a shifted windowing scheme to process patches, bringing greater efficiency by limiting self-attention computation to non-overlapping local windows while also allowing for cross-window connection [29].

We use the CIFAR10 dataset for the model pre-training and a subset of the CIFAR100 dataset for the downstream/fine-tuning task [32]. The CIFAR10 dataset is selected for the pre-training task as it contains more images per class (5000) than the CIFAR100 dataset (500). TL requires the complexity of the fine-tuning task to be less than that of the pre-trained task. Hence, in our simulations, 2-class or 5-class subsets of the CIFAR100 dataset are used in the fine-tuning phase.

Simulation is performed using Pytorch and the IBM Analog hardware acceleration kit (aihwkit) [17]. The aihwkit is an open-source toolkit integrated within Pytorch to enable the simulation of analog crossbar arrays. It is centered around the concept of analog tile, a building block that captures the computation performed on a crossbar array. The toolkit is designed to use customized unit cell configurations and advanced optimization algorithms like tiki-taka algorithms. The toolkit can simulate analog tiles of different device materials such as ReRAM, PCM ECRAM, etc. We simulate HfOx-based ReRAM devices, as they have many desirable properties, such as non-volatility, energy efficiency, high density, and ability to scale. A detailed insight into the device structure can be found here [27]. We list all relevant hyperparameters used in Table I.

III. RESULTS

A. Comparison between Analog TL and Digital TL

Fig. 2 shows the test error trace for analog and digital TL for a 2-class and 5-class task for the Swin-ViT model. The figure also shows the test error trace when the models under consideration are trained from scratch. For the ViT model, the performance of the analog TL model is better than the results for regular analog training for both fine-tuning tasks. It is also shown that the analog TL model outperforms the digital

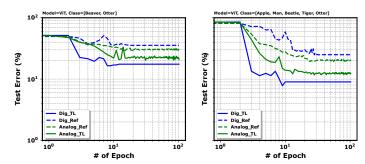


Figure 2: Training results of the analog and digital Swin-ViT model for TL and regular training (referred to as Ref) performed on the 2-class task (left) and 5-class task (right) using c-TTV2 algorithm

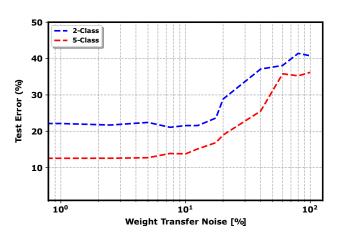


Figure 3: Relationship between the inference accuracy and weight transfer noise for Swin-ViT analog TL model for the 2-class and 5-class tasks.

model's performance. These observations can be explained by the presence of additional noise in the analog model training process. Introducing additional noise during training is equivalent to performing some form of noise injection during the forward pass of the training process. Noise injection during the model training process can be beneficial as it is used for some form of model regularization, which can help improve the model inference performance [33]. This form of model regularization can be highly beneficial in a TL scenario where the amount of training data is limited. However, the inherent presence of noise in analog model training can be problematic if the training process is sensitive to these noise sources.

For the 2-class and 5-class fine-tuning tasks, digital TL outperforms the analog TL by about 2%. The performance difference increases with respect to the task complexity. This difference is expected as the increase in task complexity increases the computation's noisy nature, leading to lower analog model inference performance. However, the performance difference is insignificant compared to the earlier algorithm [20], validating the effectiveness of the c-TTv2 algorithm.

B. Analog TL and Weight Transfer Noise

In Fig. 3, we determine the effect of the weight transfer noise on analog TL inference performance. For a 2-class task, the test error is mostly constant until the weight transfer noise exceeds $\approx 15\%$. For larger noise magnitudes, the degradation in model performance becomes more substantial. Similar behavior is also noticed for the 5-class task, except that the elbow point was $\approx 10\%$. These observations imply that the weight transfer noise can negatively affect model inference performance. At weight transfer noise below a certain point (or elbow), called the critical weight transfer noise, the effect of weight transfer noise on the analog TL model inference performance is negligible. The critical weight transfer noise point is affected by the complexity of the task and the model itself.

C. Analog TL and Device parameters

One major requirement of the proposed training algorithm to solve the problem of non-linear switching behavior in ReRAM devices is the need for a bi-directional switching device of unrealistically high symmetry and precision. Improvement in the algorithm further relaxes the requirement for high symmetry, but the need for it still exists. Hence, there is a need to investigate the robustness of the training algorithm in relation to the presence/absence of symmetry for analog TL. The behavior of the ReRAM device for this work, which is modeled using the softbound device model, is used to simulate the ReRAM device used in this work as this model is more representative of the device behavior. Hence, the impact of symmetric point skewness (SpS) and symmetric point variability (SpV) is discussed in this section. The effect of symmetric point skewness on the model inference accuracy is shown in Fig. 4. The device skewness is a measure of how far the maximum weight W_{max} is from the ideal maximum weight value of 1 (or how far W_{min} is from the ideal minimum weight). It is calculated in terms of a percentage (%),

$$SpS = \frac{W_{max} * 100}{W_{max} - W_{min}}. (2)$$

The symmetric point variability is calculated as a percentage of the weight range of 2 (+1,-1). A 5% symmetric point variability is equal to a variability value of 0.10. For the 2-class task, the symmetric point variability has minimal impact on the model performance for the symmetric point variability values under consideration, showing the robustness of the training algorithm. For a 5-class task, the analog model inference performance was not affected until a symmetric value of about 10%. At symmetric point variability beyond this point, the model inference error grows, meaning that the training algorithm has limited robustness. This observation implies that the robustness of the training algorithm to symmetric point variability is affected by the complexity of the task. Fig. 5 investigates the impact of pulse update noise and mean pulse device-to-device variability on model performance. It can be observed that the analog model inference performance remains mostly the same despite the increase in the value of the mean

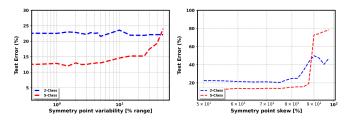


Figure 4: Relationship between the model inference accuracy and the symmetry point skew for Swin-ViT analog TL model for the 2-class and 5-class tasks trained using the c-TTv2 algorithm.

pulse device-to-device (DtoD) variability and the pulse update noise. The same observation can be made for the 2-class and the 5-class fine-tuning tasks, confirming the robustness of the c-TTv2 training algorithm to the noise in the pulse response.

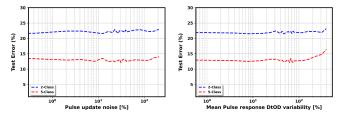


Figure 5: Relationship between the model inference accuracy, pulse update noise (left), and mean pulse device-to-device variation for Swin-ViT analog TL model for the 2-class and 5-class tasks trained using the c-TTv2 algorithm.

IV. CONCLUSION

In this paper, we assessed the performance of the c-TTv2 algorithm on an Analog Swin-ViT model for an analog TL task. We performed additional simulations to investigate the robustness of the c-TTv2 training algorithm to changes in the material specifications. Results demonstrated that c-TTv2 is suitable for training the analog TL models as it achieves better results than analog training from scratch and digital training from scratch, as expected. We also demonstrated that the c-TTv2 algorithm achieves competitive results with digital TL. Critically, the c-TTv2 algorithm was very tolerant to changes in device material specification, such as weight transfer noise, symmetry point variability, symmetry point skew, pulse update noise, and mean pulse DtoD variability. In future work, we aim to assess the performance of the c-TTv2 algorithm using more complex models and tasks such as LLM models, diffusion models, etc. Furthermore, there is a need to benchmark the performance of the training algorithm against the AGAD algorithm [20]. Lastly, there is a need to propose solutions that help narrow the inference performance gap between analog TL and digital TL models. Our intention for this paper is to lay the initial groundwork required for these future research directions.

REFERENCES

- [1] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in Artificial Neural Networks and Machine Learning-ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27. Springer, 2018, pp. 270–279.
- [2] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill et al., "On the opportunities and risks of foundation models," arXiv preprint arXiv:2108.07258, 2021.
- [3] M. S. Orenstrakh, O. Karnalim, C. A. Suarez, and M. Liut, "Detecting llm-generated text in computing education: Comparative study for chatgpt cases," in 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2024, pp. 121–126.
- [4] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting, "Large language models in medicine," *Nature medicine*, vol. 29, no. 8, pp. 1930–1940, 2023.
- [5] L. Chen, O. Sinavski, J. Hünermann, A. Karnsund, A. J. Willmott, D. Birch, D. Maund, and J. Shotton, "Driving with Ilms: Fusing objectlevel vector modality for explainable autonomous driving," in 2024 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024, pp. 14093–14100.
- [6] Y. Li, S. Wang, H. Ding, and H. Chen, "Large language models in finance: A survey," in *Proceedings of the fourth ACM international* conference on AI in finance, 2023, pp. 374–382.
- [7] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler et al., "Emergent abilities of large language models," arXiv preprint arXiv:2206.07682, 2022.
- [8] Z. Zhao, L. Alzubaidi, J. Zhang, Y. Duan, and Y. Gu, "A comparison review of transfer learning and self-supervised learning: Definitions, applications, advantages and limitations," *Expert Systems with Appli*cations, p. 122807, 2023.
- [9] S. Yin, C. Fu, S. Zhao, K. Li, X. Sun, T. Xu, and E. Chen, "A survey on multimodal large language models," arXiv preprint arXiv:2306.13549, 2023.
- [10] B. Yang, O. Fagbohungbe, X. Cao, C. Yuen, L. Qian, D. Niyato, and Y. Zhang, "A joint energy and latency framework for transfer learning over 5g industrial edge networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 531–541, 2021.
- [11] O. Fagbohungbe, S. R. Reza, X. Dong, and L. Qian, "Efficient privacy preserving edge intelligent computing framework for image classification in iot," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 4, pp. 941–956, 2021.
- [12] F. F. Athena, O. Fagbohungbe, N. Gong, M. J. Rasch, J. Penaloza, S. Seo, A. Gasasira, P. Solomon, V. Bragaglia, S. Consiglio *et al.*, "Demonstration of transfer learning using 14 nm technology analog reram array," *Frontiers in Electronics*, vol. 4, p. 1331280, 2024.
- [13] G. W. Burr, A. Sebastian, T. Ando, and W. Haensch, "Ohm's law + kirchhoff's current law = better ai: Neural-network processing done in memory with analog circuits will save energy," *IEEE Spectrum*, vol. 58, no. 12, pp. 44–49, 2021.
- [14] S. Jain, H. Tsai, C.-T. Chen, R. Muralidhar, I. Boybat, M. M. Frank, S. Woźniak, M. Stanisavljevic, P. Adusumilli, P. Narayanan et al., "A heterogeneous and programmable compute-in-memory accelerator architecture for analog-ai using dense 2-d mesh," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 1, pp. 114–127, 2022.
- [15] C. Frenkel, D. Bol, and G. Indiveri, "Bottom-up and top-down approaches for the design of neuromorphic processing systems: tradeoffs and synergies between natural and artificial intelligence," *Proceedings of the IEEE*, vol. 111, no. 6, pp. 623–652, 2023.
- [16] A. Sebastian, M. Le Gallo, and E. Eleftheriou, "Computational phase-change memory: Beyond von neumann computing," *Journal of Physics D: Applied Physics*, vol. 52, no. 44, p. 443002, 2019.
- [17] M. Le Gallo, C. Lammie, J. Büchel, F. Carta, O. Fagbohungbe, C. Mackin, H. Tsai, V. Narayanan, A. Sebastian, K. El Maghraoui et al., "Using the ibm analog in-memory hardware acceleration kit for neural network training and inference," APL Machine Learning, vol. 1, no. 4, 2023.
- [18] A. Okazaki, P. Narayanan, S. Ambrogio, K. Hosokawa, H. Tsai, A. Nomura, T. Yasuda, C. Mackin, A. Friz, M. Ishii et al., "Analog-memory-based 14nm hardware accelerator for dense deep neural networks includ-

- ing transformers," in 2022 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2022, pp. 3319–3323.
- [19] C. Lee, K. Noh, W. Ji, T. Gokmen, and S. Kim, "Impact of asymmetric weight update on neural network training with tiki-taka algorithm," Frontiers in neuroscience, vol. 15, p. 767953, 2022.
- [20] M. J. Rasch, F. Carta, O. Fagbohungbe, and T. Gokmen, "Fast and robust analog in-memory deep neural network training," *Nature Communica*tions, vol. 15, no. 1, p. 7133, 2024.
- [21] Y. Kim, T. Gokmen, H. Miyazoe, P. Solomon, S. Kim, A. Ray, J. Doevenspeck, R. S. Khan, V. Narayanan, and T. Ando, "Neural network learning using non-ideal resistive memory devices," *Frontiers* in Nanotechnology, vol. 4, p. 1008266, 2022.
- [22] M. J. Rasch, T. Gokmen, and W. Haensch, "Training large-scale artificial neural networks on simulated resistive crossbar arrays," *IEEE Design & Test*, vol. 37, no. 2, pp. 19–29, 2019.
- [23] S. Nandakumar, M. Le Gallo, I. Boybat, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Mixed-precision architecture based on computational memory for training deep neural networks," in 2018 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2018, pp. 1–5.
- [24] G. Tayfun and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices," CoRR, 2016.
- [25] T. Gokmen and W. Haensch, "Algorithm for training neural networks on resistive device arrays," *Frontiers in neuroscience*, vol. 14, p. 103, 2020.
- [26] T. Gokmen, "Enabling training of neural networks on noisy hardware," Frontiers in Artificial Intelligence, vol. 4, p. 699148, 2021.
- [27] N. Gong, M. J. Rasch, S.-C. Seo, A. Gasasira, P. Solomon, V. Bragaglia, S. Consiglio, H. Higuchi, C. Park, K. Brew et al., "Deep learning acceleration in 14nm cmos compatible reram array: device, material and algorithm co-optimization," in 2022 International Electron Devices Meeting (IEDM). IEEE, 2022, pp. 33–7.
- [28] O.-A. Imperfections, "Circuit techniques for reducing the effects of op-amp imperfections: autozeroing, correlated double sampling, and chopper stabilization," *Proceedings of the IEEE*, vol. 84, no. 11, 1996.
- [29] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on* computer vision, 2021, pp. 10012–10022.
- [30] A. Vaswani, "Attention is all you need," Advances in Neural Information Processing Systems, 2017.
- [31] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu et al., "A survey on vision transformer," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 1, pp. 87–110, 2022.
- [32] A. Krizhevsky, G. Hinton et al., "Learning multiple layers of features from tiny images," 2009.
- [33] O. Fagbohungbe and L. Qian, "Impact of 1 1 batch normalization on analog noise resistant property of deep learning models," in 2022 International Joint Conference on Neural Networks (IJCNN). IEEE, 2022, pp. 1–9.