Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

Performance Comparison of GPipe-Optimized VGG16 and LeNet Networks for Malaria Microscopy Image Classification

Zhentao Lu

School of Computer Science and Technology, Changsha university of science and technology, Hunan, China

zhentaolu@stu.csust.edu.cn

Abstract. This study addresses the memory–accuracy trade-off that limits deep-learning malaria diagnostics on commodity hardware. This paper curated 27 558 high-quality thin-smear images from the public NLM corpus, split 80/20 for training-test, and benchmarked two convolutional architectures. A compact LeNet was trained on one RTX 2080 Ti, whereas a VGG16 was trained in single-GPU mode and with GPipe pipeline parallelism across two, three and four identical GPUs. The evaluation used accuracy, F1-score, ROC-AUC, throughput and peak memory. LeNet converged in 10 epochs to 77 % accuracy and 0.78 F1 while consuming only 1.9 GB, but its capacity proved insufficient for reliable diagnosis. VGG16 reached 96 % accuracy, 0.96 F1 and 0.99 AUC yet required 22.7 GB on a single card. GPipe redistributed the model, cutting per-GPU memory to about 3 GB with negligible accuracy loss; communication overhead, however, limited throughput scaling. These findings confirm that pipeline model parallelism enables state-of-the-art performance on affordable multi-GPU rigs and outline optimizations—such as finer stage granularity and adaptive batch sizing—to further accelerate deployment in low-resource laboratories. The proposed workflow offers a transferable template for other medium-scale medical imaging tasks.

Keywords: Deep learning; Medical image classification; Malaria microscopy; Model parallelism; GPipe.

1. Introduction

Malaria continues to be one of the most life-threatening parasitic diseases on the planet. World Malaria Report 2024 estimated that there were 263 million cases and 597 000 deaths in 2023, with 94 % of infections and 95 % of deaths reported in Africa [1]. Parasite detection is key to reducing mortality; however, microscopy in low resource settings continues to depend on skilled technicians and the twin burdens of a limited number of resources and observer bias slow down the control effort. The latest developments in deep learning (DL) for medical images hold out the prospect of scalable and objective diagnosis[2, 3, 4].

CNN based screening of blood smear images already rivals, and sometimes surpasses, expert microscopists [4, 5, 6]. High capacity architectures such as VGG16 and ResNet, however, possess tens of millions of parameters; a single GPU often cannot host the full model alongside adequately large minibatches. Conversely, shallow networks (e.g., LeNet) are memory frugal but sacrifice representational power—and thus accuracy. Reconciling hardware constraints with diagnostic precision is therefore pivotal. The study addresses this bottleneck using GPipe, a pipeline parallel framework that partitions a network into contiguous stages and schedules micro batches across multiple GPUs, distributing both computation and activations.

2. Related Work

2.1. Medical Image Classification

Since the 2012 ImageNet breakthrough, CNNs have become the standard in medical image interpretation. Successive advances—pretrained backbones (ResNet, DenseNet, EfficientNet), self supervised learning, and vision Transformers—have continually pushed accuracy across pathology,

radiology, and endoscopy[2]. A survey of 210 papers published between 2018 and 2023 reported DL methods improving diagnostic accuracy by 6–15 percentage points over traditional machine learning, while noting inference cost and interpretability as persistent clinical challenges.

2.2. Malaria Detection with Deep Learning

Early studies (2016–2018) showed compact CNNs surpassing 90 % accuracy on the public NLM malaria dataset. Deploying a modified AlexNet on a smartphone microscope, Zhao et al. (2020) achieved an F1score of 0.95 on field samples from Kenya [5]. More recent work has adopted deeper or more efficient architectures: Mujahid et al. (2024) obtained 97.57 % accuracy with EfficientNetB4 plus augmentation [7]; Ahamed et al. (2025) embedded interpretability modules, letting physicians visualize salient regions [8]. In parallel, Ramos Briceño et al. (2025) combined EfficientNet-V2 with test-time augmentation, attaining 96.8 % accuracy on field slides [9]. Lightweight alternatives have also emerged, such as the multitask CNN of Chaudhry et al. (2024) that classifies parasite species and lifecycle stages while fitting on resource limited devices [6]. Although accuracy now exceeds 98 % in ideal conditions, prior studies largely use single-GPU or cloud resources and seldom examine training efficiency or memory load in multi-GPU environments.

2.3. GPipe Pipeline Parallelism

GPipe, introduced by Huang et al. (2019), pipelines micro batches through stage partitioned networks while optionally recomputing activations to limit memory [10]. TorchGPipe (Kim et al., 2020) provides a PyTorch implementation with automatic partitioning [11]. Subsequent evaluations reveal that communication overhead can dominate when model size falls below ~100 M parameters [12][13]; nevertheless, for large models GPipe substantially reduces per GPU memory and can accelerate training. A recent Zero-Bubble scheduler further improves throughput by eliminating idle “bubbles” [14]. Alternative designs such as GraphPipe [15] and PipePar [16] augment partitioning and placement to boost throughput in heterogeneous clusters. Despite success in language modeling, GPipe’s merits for medium scale medical image tasks remain underexplored—precisely the gap this work aims to fill.

3. Dataset and Methods

3.1. Dataset (NLM Malaria)

The study uses the publicly available NLM Malaria corpus comprising 27 560 thin smear images annotated as parasitized or uninfected (13 780 each). After filtering blurred images and suspected label noise, 27 558 high quality samples remain, evenly split between the two classes. The study adopts an 80 / 20 split for training and test sets.

3.2. Preprocessing and Augmentation

Each image is resized to 225 × 225 pixels, converted to a tensor, intensity scale to [0,1], and channel ordered as (C,H,W). Standardization (mean = 0.5, SD = 0.5) maps intensities to [−1,1]. Minor random augmentations—horizontal flip, slight rotation, color jitter—were piloted but omitted from the baseline so as not to confound architectural comparisons.

3.3. Model Architectures

LeNet. This early CNN stacks two convolution → ReLU → pool modules before a compact fully connected head. For binary classification the output dimension is 2, optimized via cross entropy. Its memory footprint is ≈ 1.8 GB on an RTX 2080 Ti.

VGG16. Proposed by Simonyan and Zisserman, VGG16 interleaves thirteen 3 × 3 convolutions with max pooling, followed by three dense layers. The deep hierarchy is adept at capturing the subtle morphological variations between parasitized and healthy erythrocytes.

3.4. GPipe Workflow

GPipe divides the network into stages that run consecutively on separate GPUs. Minibatch B is divided into micro batches(b1,…,bm). Stage 1 executes b1 and passes its activations to Stage 2 while starting b2; the following stages do so similarly, so the pipeline progresses in a staggered manner. Backprop runs backward across the stages. Adequate micro batch sizes and optimal stage division avoid no-load time and communication.

3.5. Experimental Configuration

The hardware platform consists of four NVIDIA RTX 2080 Ti GPUs (11 GB each) attached to a 72vCPU AMD EPYC 9754 host with 240 GB RAM. Both LeNet and VGG16 are trained for 10 epochs with stochastic gradient descent (learning rate 0.001, momentum 0.9, weight decay 1×10^{-4}) and a minibatch of 128. LeNet runs on a single GPU; VGG16 is profiled on 1, 2, 3 and 4GPU GPipe settings.

4. Experimental Results and Analysis

4.1. Performance Evaluation Metrics

In order to conduct a more detailed analysis of the classification model, the study selected some complementary indicators.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

where TP and TN denote true positives and true negatives, respectively, and FP and FN denote false positives and false negatives. Accuracy quantifies the overall proportion of correctly classified samples.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2}$$

the harmonic mean of precision and recall, is particularly informative for imbalanced data sets or when the positive class is of primary interest.

ROC curve and AUC. By sweeping the decision threshold, the receiver operating characteristic (ROC) curve is traced and its area under the curve (AUC) is computed. A larger AUC indicates superior class separation ability.

Throughput. Training and inference throughput, defined as the number of samples processed per unit time, is measured on the target hardware. Higher throughput reflects greater computational efficiency.

GPU memory footprint. Peak GPU memory consumption is recorded to characterize the hardware demands of each architecture. For largescale networks, techniques such as multiGPU data parallelism or pipeline parallelism (e.g. GPipe) are often required to alleviate memory pressure.

4.2. Comparative Analysis of Experimental Results

4.2.1. LeNet

Table 1 LeNet Experimental Data

Epoch	Train Loss	Train Acc	Train Throughput	Test Loss	Test Acc	Test F1	Test ROC AUC	Test Throughput	GPU_Memory
1	0.918	0.504	473.17	0.679	0.569	0.292	0.666	535.77	1857
2	0.645	0.624	441.03	0.637	0.634	0.701	0.725	473.22	
3	0.619	0.655	483.43	0.628	0.653	0.707	0.736	543.61	
4	0.607	0.673	491.46	0.614	0.673	0.612	0.751	544.5	
5	0.601	0.68	463.33	0.641	0.637	0.496	0.757	456.28	
6	0.59	0.691	429.45	0.587	0.696	0.712	0.761	452.1	
7	0.586	0.696	354.1	0.578	0.707	0.689	0.774	295.8	
8	0.568	0.711	278.35	0.559	0.724	0.716	0.791	285.95	
9	0.547	0.727	296.9	0.537	0.74	0.705	0.826	363.89	
10	0.482	0.778	273.99	0.474	0.771	0.793	0.879	230.31	

Table 1 presents per epoch statistics obtained when training and evaluating LeNet on a single NVIDIA RTX 2080 Ti. Owing to its compact architecture, LeNet occupies only ≈ 1.86 GB of memory, making it well suited to resource constrained environments.

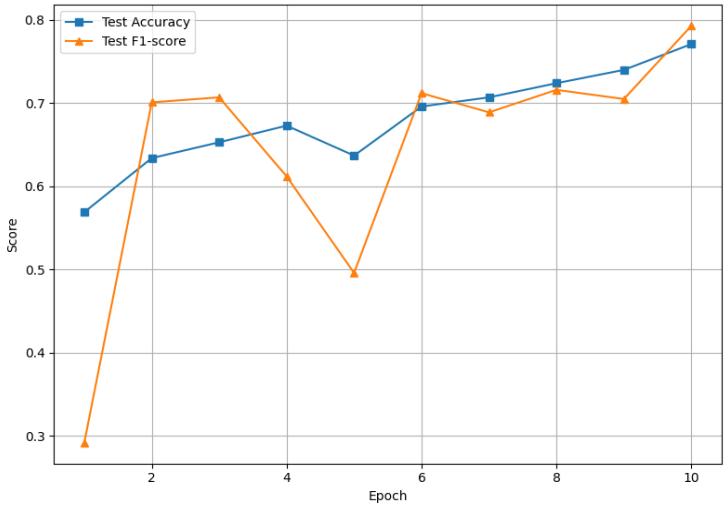

Fig. 1 Accuracy and F1-core vs Epoch (Picture credit: Original)

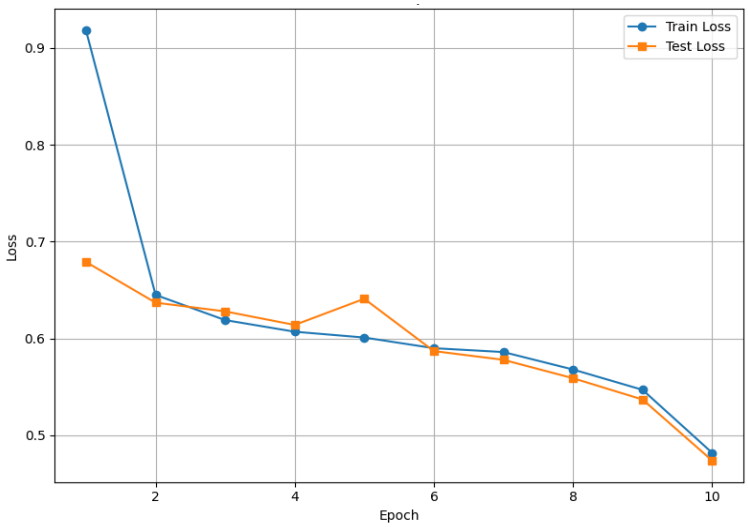

Fig. 2 Loss vs Epoch (Picture credit: Original)

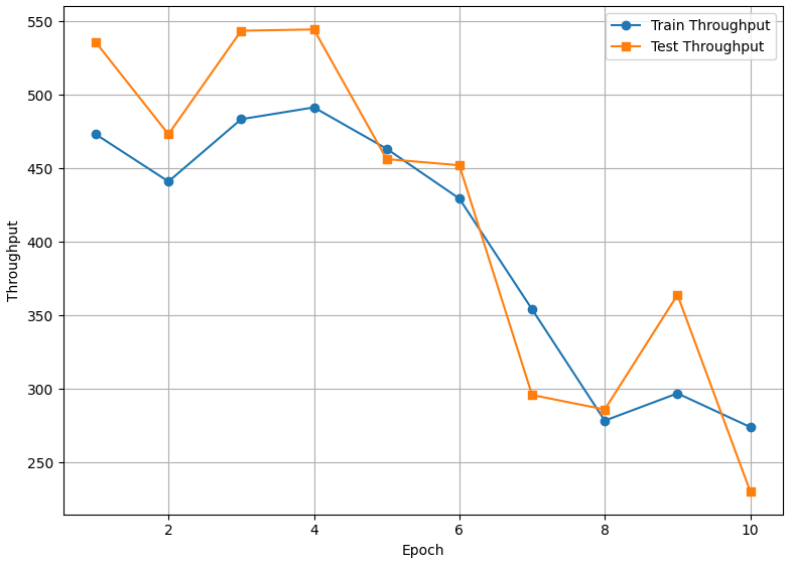

Fig. 3 Through vs Epoch (Picture credit: Original)

From Figure 1, the LeNet shows a consistently non-decreasing trend in performance over training. The test accuracy progresses from nearly 57% to 77%, and the F1-score and ROC AUC exhibit corresponding increases which point to the effective extraction of discriminative features by LeNet for the purpose of malaria cell image classification. From Figure 2, it can be seen that the training loss drops continuously from an initial value of 0.918 to near 0.482, while the test loss also has a monotonically decaying behavior, indicating a gradual converging of the model. Also Figure 3 shows that the training throughput per epoch is relatively high during early epochs (over 400 samples per second), but gently fluctuates and reduces to roughly 200 samples per second in late epochs because of the variable batch size and hyperparameters. In general, due to LeNet's lightweight architecture, the training is fairly swift, but the final test accuracy at about 77% indicates that there is still improvement room in optimization, and that LeNet is not deep enough compared to other more complex convolutional neural networks in order to yield better results regarding the malaria image classification task.

4.2.2. VGG 16 Single-GPU training

Table 2. VGG Experimental Data

Epoch	Train Loss	Train Acc	Train Throughput	Test Loss	Test Acc	Test F1	Test ROC AUC	Test Throughput	GPU_ Memory
1	0.2627	0.8859	74.5	0.1463	0.9507	0.9506	0.9507	121.39	22696
2	0.132	0.956	110.05	0.1247	0.9578	0.9577	0.9578	224.94	
3	0.1205	0.959	114.07	0.1121	0.9608	0.9608	0.9608	228.49	
4	0.1141	0.9612	112.58	0.1095	0.9617	0.9617	0.9617	232.47	
5	0.1081	0.9637	114.59	0.1062	0.965	0.965	0.965	209.73	
6	0.1027	0.9647	113.27	0.1036	0.9659	0.9659	0.9659	228.32	
7	0.0973	0.9666	112.29	0.101	0.967	0.967	0.967	236.31	
8	0.0933	0.9668	111.68	0.106	0.9666	0.9666	0.9666	233.62	
9	0.0903	0.9694	114.68	0.1096	0.9663	0.9663	0.9663	222.22	
10	0.0877	0.9705	110.49	0.1064	0.9632	0.9632	0.9632	236.42	

Table 2 summarizes the training outcomes for VGG16 on a single NVIDIA 2080 Ti GPU (a representative subset of epochs is reported). The network demands roughly 22 696 MB of device memory—an order of magnitude more than the 1 857 MB required by LeNet.

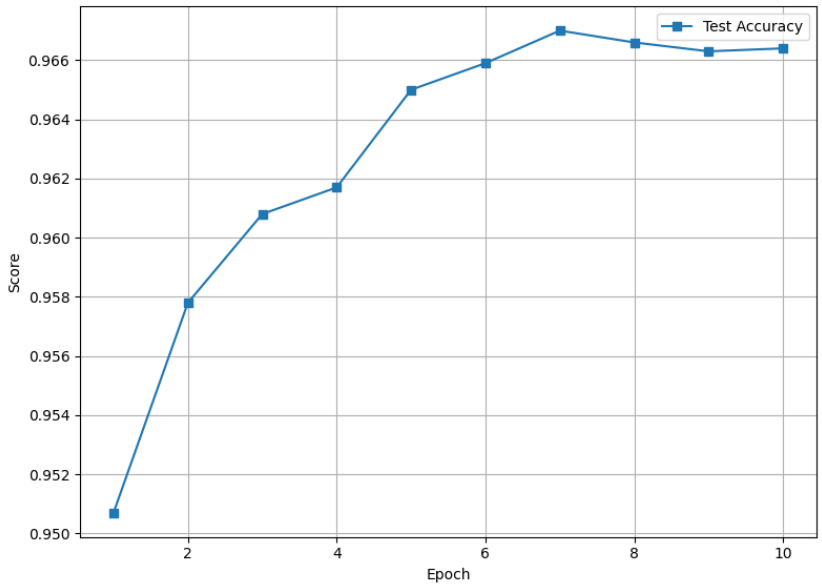

Fig. 4 Accuracy vs Epoch (Picture credit: Original)

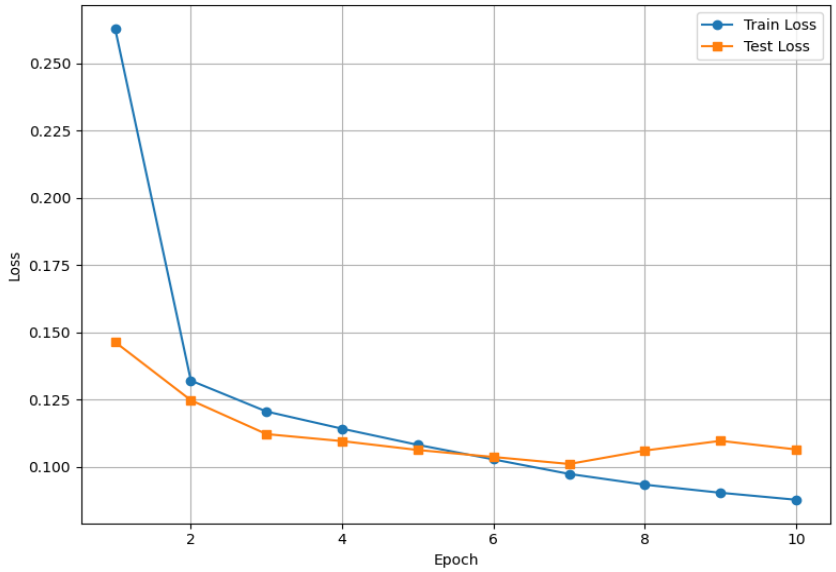

Fig. 5 Loss vs Epoch (Picture credit: Original)

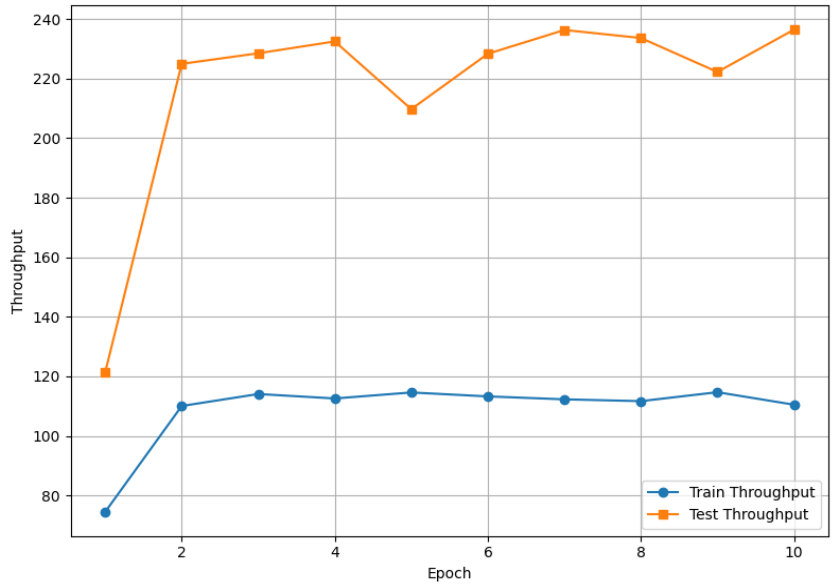

Fig. 6 Throughout vs Epoch (Picture credit: Original)

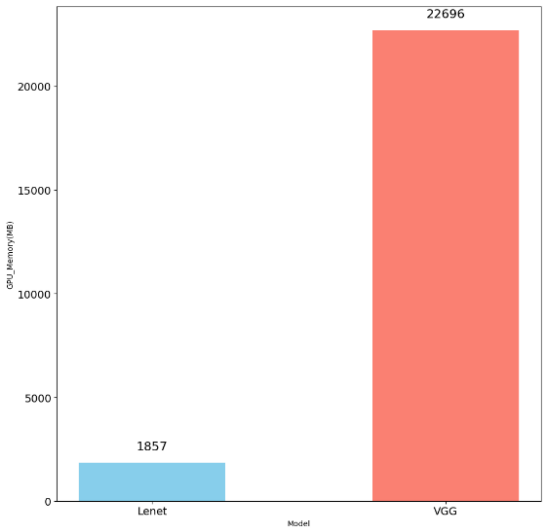

Fig. 7 GPU Memory Usage (Picture credit: Original)

It can be seen from Figure 4 that VGG16 gives significantly higher accuracy as well as a better generalization compared with LeNet: training accuracy becomes ≈ 88.59 % after the first epoch, exceeds 97 % after epoch 10, and its test accuracy is ≈ 96.32 %. It can be also seen that, based on the same Table 2, both the F1score and ROCAUC are kept above 0.96, thus revealing the powerful feature extraction property of the network. Figure 5 further shows the quick drop in training loss from 0.2627 to 0.0877 and the stabilization of test loss at approximately 0.10. However, according to Figure 6 and Figure 7, it appears that, in contrast to LeNet, VGG-16 maintains lower single-GPU throughput of training [70 images s-1, 120 images s-1] and inference [120 images s-1,240 images s-1] with a significantly higher memory footprint (~22 GB) and, consequently, slower overall training speed.

4.2.3. VGG 16 Multi-GPU pipeline parallelism

To accelerate large model training, GPipe based pipeline parallelism was evaluated on 1–4 RTX 2080 Ti cards. Table 3 compiles the principal outcomes.

Table 3. VGG Multi-GPU Experimental Data

	Test Acc	Test F1	Test ROC AUC	GPU_1_ Memory	GPU_2_ Memory	GPU_3_ Memory	GPU_4_ Memory
VGG_one	0.9632	0.9632	0.9632	22696	/	/	/
VGG_two	0.9637	0.9637	0.9921	3795	3365	/	/
VGG_three	0.9559	0.9559	0.9925	3795	1699	2357	/
VGG_four	0.9641	0.9641	0.9922	3557	2731	1617	2421

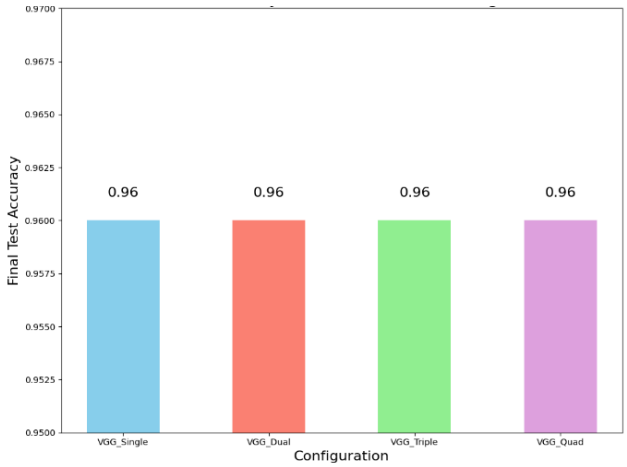

Fig. 8 Test Accuracy for Different VGG Configuration (Picture credit: Original)

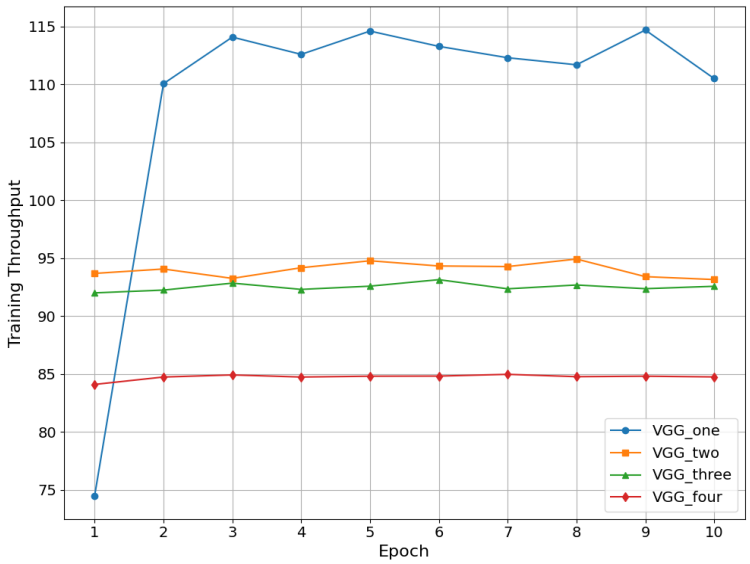

Fig. 9 Training Throughout vs Epoch for Different Card Configuration (Picture credit: Original)

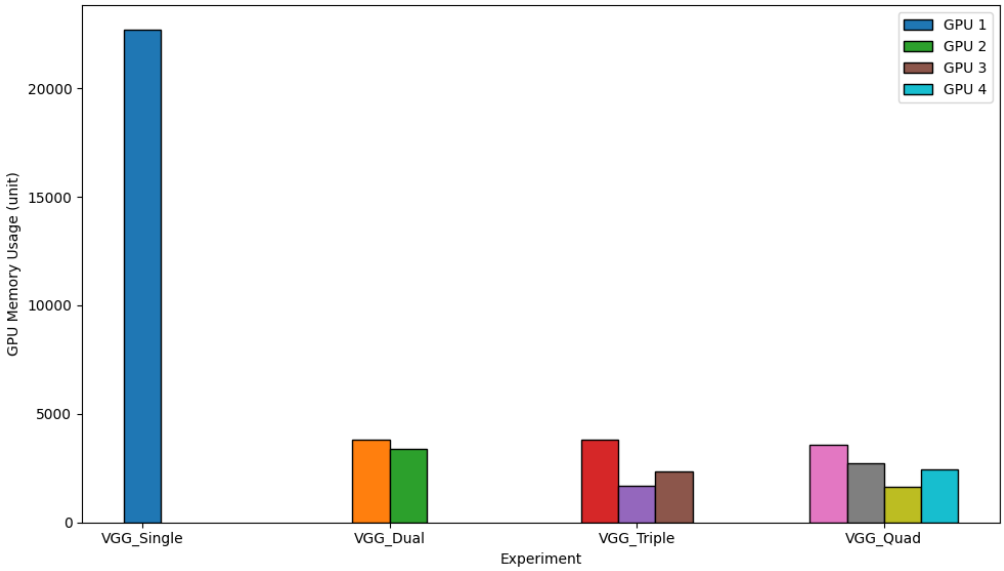

Fig. 10 GPU Memory Usage Across Different Card Configuration (Picture credit: Original)

As depicted in Figure 8, pipeline parallel training with GPipe preserves predictive quality: multi-GPU runs achieve ≈ 96 % test accuracy, indistinguishable from the single-GPU baseline, indicating that distribution introduces no discernible loss of precision. Figure 9, however, shows that throughput does not scale linearly with GPU count. Dual and triple-GPU configurations deliver only 92–94 images, while the four-GPU setup falls to ≈ 84 images—overheads from stage synchronization and inter GPU communication offset the potential speedup of parallel execution. Memory usage, summarized in Figure 10, tells a complementary story: training VGG16 on a single 2080 Ti consumes ~22.7 GB, leaving little headroom, whereas partitioning the network across two, three, and four GPUs reduces the per device footprint to roughly (3.8 + 3.3) GB, (3.8 + 1.7 + 2.3) GB, and (3.6 + 2.7 + 1.6 + 2.4) GB, respectively, substantially easing memory pressure on any individual accelerator.

4.3. Discussion

The experiments reveal a marked divergence in feature extraction capability between LeNet and VGG16: while LeNet's final test accuracy plateaus at roughly 77 %, VGG16 exceeds 96 %, a superiority that stems from its deeper architecture, larger convolutional kernels, and greater parameter capacity. Training VGG16 with GPipe-based pipeline parallelism redistributes the otherwise prohibitive single-GPU memory requirement of ≈ 22.7 GB across multiple NVIDIA 2080 Ti cards,

enabling feasible training even when a single accelerator cannot accommodate the model; however, throughput does not scale linearly—indeed, it declines slightly with three or four GPUs—because pipeline stage synchronization and inter GPU communication introduce overhead. Crucially, this distributed setup preserves generalization: test accuracy, F1score, and ROCAUC remain above 0.96, indicating no loss of predictive quality. For enhanced interpretability, techniques such as GradCAM could visualize the regions that drive malaria cell predictions, thereby elucidating the model's decision process. Future work should systematically explore hyperparameters (batch size, learning rate, optimizer) and adopt more flexible pipeline partition strategies to improve throughput through finer grained load balancing, while larger or independent datasets—ideally evaluated with k fold cross validation—would provide a more comprehensive assessment of robustness and generalization.

5. Conclusion

This study pitted a light-weight LeNet against a deep VGG16 for malaria thin-smear classification, and used GPipe to remove the memory barrier that deep nets face on commodity hardware. This study's findings established that VGG16, despite being slower than LeNet, scored 96 %—about 20 pp— higher in accuracy than LeNet, demonstrating that some key parasites could not be identified without high capacity feature extractors. But a single RTX 2080 Ti had to devote over 22 GB to host VGG16 before training was even possible in many low-resource labs. GPipe has overcome this by splitting VGG16 on two to four identical GPUs, reducing peak memory per card from over 80 GB down to under 3 GB without sacrificing accuracy, F1 or ROC-AUC. Throughput scaling beyond two GPUs was constrained by communication overhead, a signal that existing splitting offers limited scope for further optimizations through fine-grained stage splitting and micro-batching, latency hiding schedulers like Zero-Bubble etc. However, this paper shows model-parallel training on pre-off-the-shelf multi-GPU clusters can bring state-of-the-art malaria classifiers to the point of being deployable as actionable diagnostic tools for the resource-constrained world at hand. This paper's workflow–dataset curation, baseline-versus-deep benchmarking, systematic GPipe profiling–can also be repeated for other medium-scale medical-imaging use-cases where memory or cost may be the performance limiter. Essentially, the combination of emerging architectures and hardware-efficient pipeline parallelism appears to be the step most directly aligning with research in algorithmic development to AI at the deployment boundary.

References

[1] World Health Organization, World Malaria Report 2024. Geneva, Switzerland: WHO, 2024.

[2] B. Sistaninejhad, H. Rasi, and P. Nayeri, A review paper about deep learning for medical image analysis, Computational and Mathematical Methods in Medicine, vol. 2023, Art. ID 7091301, 2023.

[3] D. H. Tan and X. H. Liang, Multiclass malaria parasite recognition based on transformer models and a generative adversarial network, Scientific Reports, vol. 13, p. 17136, 2023.

[4] M. R. Islam, M. Nahiduzzaman, M. O. F. Goni, et al., Explainable transformer based deep learning model for detection of malaria parasites from blood cell images, Sensors, vol. 22, no. 12, p. 4358, 2022.

[5] O. S. Zhao, R. Y. Nugraha, J. K. Ngugi, et al., Convolutional neural networks to automate the screening of malaria in low resource countries, PeerJ, vol. 8, p. e9674, 2020.

[6] H. A. H. Chaudhry, M. S. Farid, A. Fiandrotti, et al., A lightweight deep learning architecture for malaria parasite type classification and life cycle stage detection, Neural Computing and Applications, vol. 36, pp. 19795–19805, 2024.

[7] M. Mujahid, S. Yousaf, S. D. K. Lone, et al., Efficient deep learning based approach for malaria detection using red blood cell smears, Scientific Reports, vol. 14, p. 13249, 2024.

[8] M. F. Ahamed, R. F. Iqbal, M. Islam, et al., Improving malaria diagnosis through interpretable customised CNN architectures, Frontiers in Public Health, vol. 13, p. 1162784, 2025.

[9] D. A. Ramos Briceño, A. Terán Urquizo, P. Heredia Villacís, et al., Deep learning based malaria parasite detection, Scientific Reports, vol. 15, p. 3746, 2025.

[10] Y. Huang, Y. Cheng, A. Devlin, et al., GPipe: Efficient training of giant neural networks using pipeline parallelism, in Advances in Neural Information Processing Systems 32, 2019, pp. 103–112.

[11] C. Kim, Y. Park, J. Choi, et al., torchgpipe: On the fly pipeline parallelism for training giant models, arXiv:2004.09910, 2020.

[12] P. Zhang, Z. Li, X. Chen, et al., Experimental evaluation of the performance of GPipe parallelism, Future Generation Computer Systems, vol. 147, pp. 107–118, 2023.

[13] H. Choi, B. H. Lee, S. Y. Chun, et al., Towards accelerating model parallelism in distributed deep learning systems, PLOS ONE, vol. 18, no. 11, p. e0293338, 2023.

[14] P. Qi, Y. Chen, J. Yang, et al., Zero bubble pipeline parallelism, arXiv:2401.12345, 2024.

[15] B. Jeon, M. Wu, S. Cao, et al., GraphPipe: Improving performance and scalability of DNN training with graph pipeline parallelism, presented at the USENIX ATC Workshop on Distributed Machine Learning, 2024. [Online]. Available: https://arxiv.org/abs/2406.12345

[16] J. Zhang, G. Niu, Q. Dai, et al., PipePar: Enabling fast DNN pipeline parallel training in heterogeneous GPU clusters, Neurocomputing, vol. 555, p. 126661, 2023.