RCV2023 Challenges: Benchmarking Model Training and Inference for Resource-Constrained Deep Learning

Rishabh Tiwari[*1,2], Arnav Chavan[*1,3], Deepak Gupta[*1,4], Gowreesh Mago[1], Animesh Gupta[1], Akash Gupta[3], Suraj Sharan[1], Yukun Yang[5], Shanwei Zhao[5], Shihao Wang[5], Youngjun Kwak[6,7], Seonghun Jeong[7], Yunseung Lee[7], Changick Kim[6], Subin Kim[6], Ganzorig Gankhuyag[8], Ho Jung[9], Junwhan Ryu[9], HaeMoon Kim[9], Byeong H. Kim[10], Tu Vo[11], Sheir Zaheer[11], Alexander Holston[11], Chan Park[11], Dheemant Dixit[12], Nahush Lele[12], Kushagra Bhushan[12], Debjani Bhowmick[1], Devanshu Arya[13], Sadaf Gulshad[14], Amirhossein Habibian[15], Amir Ghodrati[15], Babak Bejnordi[15], Jai Gupta[2], Zhuang Liu[16], Jiahui Yu[17], Dilip Prasad[4], Zhiqiang Shen[18]

[1]Transmute AI Lab (Texmin Hub) [2]Google Research [3]NyunAI [4]UiT Tromso
[5]Ant Group [6]KakaoBank Corp. [7]KAIST [8]KETI [9]Hanwha Systems
[10]Korea Institute of Industrial Technology [11]KC Machine Learning Lab [12]IIT ISM Dhanbad
[13]Serket BV [14]University of Amsterdam [15]Qualcomm AI Research [16]Meta AI Research
[17]Google Brain [18]MBUZAI

Abstract

This paper delves into the results of two resource-constrained deep learning challenges, part of the workshop on Resource-Efficient Deep Learning for Computer Vision (RCV) at ICCV 2023, focusing on memory and time limitations. The challenges garnered significant global participation and showcased a range of intriguing solutions. The paper outlines the problem statements for both tracks, summarizes baseline and top-performing approaches, and provides a detailed analysis of the methods used. While the presented solutions constitute promising initial progress, they represent the beginning of efforts needed to address this complex issue. We conclude by emphasizing the importance of sustained research efforts to fully address the challenges of resource-constrained deep learning.

1. Introduction

Deep learning has revolutionized numerous fields, including visual recognition, language understanding, and healthcare [21, 4, 35, 13]. This success is largely attributed to the model's ability to automatically learn complex hierarchical features from raw data. However, this complexity has given rise to models with millions, if not billions, of parameters, making them computationally intensive and memory-hungry. Consequently, this has led to significant computational challenges, especially in the realms of model training and inference.

As deep learning models continue to grow in complexity and size, the need for resource-efficient solutions becomes paramount. Traditional computing resources struggle to keep up with the escalating demands of these tasks, leading to longer training times, increased energy consumption, and limited accessibility to state-of-the-art models in resource-constrained environments. This paper delves into the critical role of efficiency in both the training and inference phases of deep learning.

The importance of efficient training has become ever more evident in 2023. Many research groups struggle with a significant challenge: the formidable resource requirements demanded by these large models. Training large models becomes an arduous task, often exceeding the computational capacity available. This concern is further exacerbated when dealing with huge datasets from domains like medical imaging, aerial surveillance, or high-energy physics, where the sheer volume of data necessitates substantial memory allocation, straining even traditional deep learning models [1, 2].

Consider ChatGPT, a prime example of state-of-the-art

[*]Authors contributed equally.

large language models [6, 3]. Training a model of this scale demands an enormous amount of computational resources. This process also involves hyperparameter optimization and architecture tuning. As a result, only organizations with access to industry-leading computational infrastructure could undertake such a task.

Similarly, in medical imaging, the development of deep learning models for tasks like diagnosing complex diseases from scans presents a challenge. Enormous volumes of high-resolution images are needed for training, demanding significant memory and computational power [42, 5, 40]. This not only restricts the accessibility of technology to well-funded institutions but also hinders its application in resource-constrained environments like rural clinics or underfunded hospitals.

While the efficiency of training is crucial, the efficiency of inference is as important, particularly in applications that require real-time responses. Many deep learning models find application in scenarios where low-latency responses are crucial, such as autonomous driving, real-time video analysis, and industrial automation [4, 20]. In these contexts, the model's ability to provide rapid predictions directly influences its practical utility.

Efficient inference is not merely about fast predictions. It also has significant implications for energy consumption and cost-effectiveness [53, 58]. As deep learning integrates into edge and IoT devices, available computational resources are often limited. Models delivering accurate results with fewer operations contribute to extended battery life and cost-efficient deployment. In critical domains like healthcare, efficient inference can yield timely, potentially life-saving insights. Therefore, while training efficiency is important, optimizing inference is key to fully realizing deep learning's potential in real-world applications.

In recent years, dedicated events have been hosted to address the issues outlined above. Some examples include the NeurIPS Efficient Deep Learning (EDL) workshop, NeurIPS 2022 Efficient Natural Language and Speech Processing (ENLSP) workshop, CVPR Efficient Deep Learning for Computer Vision (ECV) workshop, ICLR workshop on Energy-Efficient Deep Learning, ECCV workshop on Efficient Deep Learning for Visual Recognition and TinyML summit. This is only a limited list of events related to efficient deep learning, among many others.

While the community is actively exploring multiple approaches to improve the efficiency of deep learning, there is no single solution that has fully addressed this problem. Further, more recent issues such as efficient fine tuning of large vision and language models, improving the inference of generative models for real-world applications, etc, are also mostly unaddressed. To address these pending issues, we are hosting the Resource-Efficient Deep Learning for Computer Vision (RCV) workshop, the first in its series that primarily aims at discussing innovation towards practical implementations of efficient deep learning for computer vision. RCV aims at bringing together researchers and industry practitioners who work towards building efficient computer vision models with deep learning, serving as a platform for discussion.

To encourage research efforts in efficient deep learning, we organized two challenges focusing on resource-efficient model training and inference where participants are required to optimize model training under computational memory constraint, and inference under latency constraint. In this paper, we discuss the details related to the two challenges hosted as part of the RCV 2023 workshop of the ICCV conference. The two challenges are *Budgeted Model Training Challenge* and *Budgeted Model Inference Challenge*. We provide details related to the problem statement of the two challenges, as well as discuss the best solutions submitted to tackle the posed problem statements. We also discuss recent development and potential future directions for efficient deep learning.

2. Recent Development

Community efforts to enhance the efficiency are through multiple approaches. In this section, we outline recent developments by key topics and discuss notable contributions.

Model Quantization. Reducing the numerical precision of deep learning models has become a common technique in both training and deployment phases. Notably, recent studies such as [11, 56] have shown it is possible to quantize large language models to 8-bit precision for both weights and activations without compromising performance. [36] addresses the challenges encountered when quantizing vision transformers by using auxiliary loss functions. [37] introduces a method for quantizing large language models in the absence of data. [33] explores the possibility of post-training quantization of billion-scale diffusion models to 8-bit formats, thereby supporting the general applicability of these precision-reduction techniques.

Parameter Efficient Fine-tuning. Recent works like [23, 25, 7, 12] have made progress in fine-tuning large models effectively in settings with limited resources. [23], for example, takes advantage of the low-rank nature of weight matrices by applying low-rank decomposition during fine-tuning. Extending this, [12] introduces quantization for pre-trained models, followed by high-precision fine-tuning focused on these low-rank matrices. Additionally, [7] provides a holistic view on efficient fine-tuning, combining principles from works like [23, 25, 34] within a unified search-based optimization framework.

Network Pruning. Network pruning [38, 45] is an effective

technique for reducing the complexity of neural networks. Structured pruning methods, like those in [14], optimize for modern GPU architectures, achieving significant computational gains. Unstructured methods, on the other hand, focus on reducing the number of parameters [51].

For large language models, recent works [15, 46] have offered pruning methods that avoid the need for retraining, although they may suffer some performance loss. [33] extends pruning techniques to diffusion models using time-based Taylor scoring. Given the inherent redundancy in transformer architectures, numerous strategies have been explored for making them more efficient in both language and vision tasks [41, 8, 54, 45, 29]. A notable development by [57] shows that quantization and pruning can be combined within a single optimization framework, leading to practical performance improvements on GPUs.

Knowledge Distillation. Knowledge Distillation [22] is a well-known method for improving smaller models. For example, [50] used signals from a larger model to speed up the learning of vision transformers, enhancing both training speed and efficient inference. Recently, [26] developed a method to transfer knowledge from closed-source large language models to smaller, open-source versions. This opens up possibilities for knowledge transfer across different model architectures. In addition to parameter distillation [27], time-step distillation has also been introduced in works like [39, 43] to reduce the computational burden during the inference process of diffusion models.

3. Proposed Challenges

In this section, we describe the two challenges that form part of the RCV workshop.

3.1. Budgeted Model Training Challenge

Description. In the context of the budgeted model training challenge, we present the task of ImageNet100 classification. ImageNet100 is a subset of data created from Imagenet-1K [10]. The objective was to develop a classifier for categorizing ImageNet100 samples into predefined classes. The training and evaluation process was constrained by a 6GB GPU memory limit and a time restriction of 9 hours. This challenge utilized a V100 GPU card along with a 4-core CPU. The competition took place in two phases, detailed as follows.

Phase I. This phase was a standard classification problem where the goal was to maximize the accuracy on the test set and improve the ranking on the leaderboard. Labels of the test set were hidden, and an evaluation engine was used to evaluate the submissions. This phase of the competition was hosted on Kaggle, and the leaderboard of the Kaggle platform was used to rank the solutions.

Phase II. We posted a baseline solution, and all solutions that obtained an accuracy higher than the baseline accuracy were eligible to submit their solutions for Phase II. This phase was run, and the leaderboard was hosted at the competition website. We keep updating the leaderboard on a rolling basis. However, every solution can be expected to be reflected on the leaderboard within 5 days from the day of submission. The submissions were trained and evaluated offline on our servers, with the hardware constraints mentioned in the challenge description, on a separate subset of data then released publicly.

3.2. Budgeted Model Inference Challenge

Description. In the scope of this constrained computational challenge, we present the problem of classifying UltraMNIST digits [17]. This task requires handling during test time with constrained GPU computational memory and time constraints. The UltraMNIST dataset used in this challenge is an adapted version of the UltraMNIST dataset and includes images with 3-5 digits per image. Each of these digits is sourced from the original MNIST dataset. The objective is to predict the sum of the digits in each image, a number that can range from 0 to 27. For the final evaluation, each submitted inference script along with the trained model's weights will be used to evaluate performance on a separate, undisclosed test set that is different from the public test set. The inference speed and accuracy on this private test set are used to determine the model's final score. All models are tested on an RTX 8000 GPU with a memory limitation of 16 GB.

The competition was conducted in 2 phases:

Phase I. This stage was structured as a conventional classification challenge with the aim of improving test set accuracy and advancing one's position on the leaderboard. Conducted on Kaggle, the phase used Kaggle's leaderboard to rank the solutions. It is crucial to emphasize that accuracy was the primary metric for evaluation at this stage. Although considerations such as GPU memory and inference speed were not mandatory requirements, participants were advised to create efficient models. This was recommended to facilitate a seamless transition to the next phase. The use of overly complex networks, which are known for slow inference speeds, was discouraged. While such networks might elevate rankings in this phase, they were unlikely to be suitable for Phase II.

Phase II. Phase II was conducted in an offline mode, and the resulting models were evaluated on a separate private test set. Every team had the opportunity to submit their two best-performing models in one instance. The evaluation procedure for this phase was as follows:

$$\mathbf{I}_{\text{score}} = \frac{\mathbf{P}_{\text{acc}}^{\mathbf{2}}}{\mathbf{T}_{\text{infer}}}$$

Table 1: Peformance scores for the top 5 teams in the budgeted model training challenge. All solutions were developed with a training time and GPU memory limits of 9 hours and 6 GB, respectively. All solutions were evaluated on a V100 GPU with 4 CPUs.

Team	Accuracy (%)
RABS	91.38
xNN	91.34
AndrewG	91.30
yuanxi	90.42
TuVo	85.30
Baseline	83.10

where $\mathbf{P}_{\text{acc}}$ denotes the classification accuracy in percentage and $\mathbf{T}_{\text{infer}}$ denotes inference time in minutes. This scoring metric is designed to evaluate models based on an empirical balance of performance and inference time.

4. Results

4.1. Training track

We present here the baseline solution as well as the summary of the solutions presented by the top participating teams for the budgeted training challenge track.

Baseline. For baseline, we followed [55] to train a ResNet50 model with Mixup augmentation and cosine decay with a warmup of 5 epochs as the learning rate scheduler respecting the GPU memory and time constraints posed in the challenge description to get a test accuracy of 83.10%.

Team RABS. The team employed a resource-aware backbone search (RABS), consisting of profile and instantiation stages. The objective was to identify optimal models that efficiently utilize either automatic mixed precision (AMP) or single precision floating point format (FP32). Secondly, their proposed ensemble strategy harnessed multi-inferences with randomly flipped multi-resolution images. This new ensemble solution not only boosted accuracy but also addressed the challenges of time and memory constraints.

In addressing the problem description centered on the ImageNet-100 subset, the team's model focused on maximizing accuracy while adhering to the limitations of GPU memory (6 GB) and training time (9 GPU hours).

Their method included critical adjustments such as reducing training time from 9 to 3 hours on RTX 3090, selecting ResNest50d_1s4x24d [59] as their backbone, and configuring parameters like batch size and max epochs. They used AdamW optimizer and a cosine learning rate scheduler for optimization.

The team explored mixed precision training to reduce gpu memory usage. Leveraging AMP led to an increase in batch size from 56 to 96, consequently accelerating training speed and expanding the maximum epochs from 46 to 72. This approach exhibited a significant 3% higher validation accuracy compared to the model without AMP. Furthermore, the incorporation of half-precision floating point format calculation for learnable parameters contributed to enhanced throughput.

To adhere to GPU memory constraints, they strategically employed asymmetric image sizes of 160 and 224 for training and deployment. Their multi-inference ensemble methodology adeptly combined model outputs based on regular and flipped test images, yielding consistent performance improvement. Notably, this approach harnessed high-resolution images to capitalize on abundant information, while flipped images introduced the desired generalization into their trained model.

The team's resource-aware backbone exploration further enhanced their strategy. By presenting candidate models that optimized batch size and training epochs, they derived adaptive learning rates and identified ResNest50d_1 as the optimal backbone. This backbone selection was pivotal for their two primary methods: augmenting batch size using AMP and employing asymmetric image sizes. Notably, the team's comprehensive evaluation highlighted the collective contributions of larger batch sizes, image sizes, and epochs to their overall performance improvement.

Team xNN. The approach introduced by the team focuses on the optimal use of training resources. It also emphasizes refining training techniques to guarantee efficient model outcomes. They identify the crucial factors of training efficiency, given the limitations on training time and GPU memory budget. The team highlights the importance of not only high accuracy but also efficient training when selecting models for the task. To objectively evaluate different models, they utilized the timm library's model list, and conducted comprehensive evaluations using a V100 GPU with a 6GB memory limit. After analyzing the training efficiency of several model families, they finalized ese_vovnet39b [31, 30].

The team further delved into the optimization of training strategies that are independent of the model structure.

Table 2: Peformance scores for the top 5 teams for the budgeted model inference challenge. All solutions were evaluated using $\mathbf{I}_{\text{score}}$ metric on a RTX8000 GPU with a GPU memory constraint of 16 GB.

Team	Model	Image Size	Inference Time (in minutes)	Accuracy (%)	Score
xNN	EfficientNetv2 B0	768	0.6	92.35	14142.69
FTL	EfficientNetv2 B0	512	0.73	84.57	9693.48
ganzoo	MobileNetV3	1,024	0.73	82.35	9229.36
HSC	YOLOv6m	512	0.99	79.82	6422.56
IIT Dhanbad	EfficientNet B3	512	1.16	82.89	5901.78
Baseline	MobileNetV2	1,024	1.63	35.25	759.71

Mixed precision training emerged as a key technique, enabling reduced memory usage and potentially faster training without compromising accuracy. Another strategy employed was gradient accumulation, which involved aggregating gradients over multiple mini-batches before updating weights. This increased the effective batch size without straining GPU memory, allowing for larger batch sizes and potentially accelerated convergence.

In addition, they explored the impact of lower training resolution and higher testing resolution, resulting in consistent accuracy gains. The choice of optimizers was further evaluated, transitioning from the Adam optimizer to both AdamW and NovoGrad. The impact on accuracy was mixed: while the adoption of AdamW showed an enhancement in results, the use of NovoGrad led to a decline.

By strategically approaching both model selection and training strategy optimization, the team's methodology underscores the importance of balancing accuracy and efficiency within the constraints of training resources.

Team TuVo. The team's approach is centered around the development of a versatile learning rate scheduler tailored to resource-constrained scenarios, particularly in budgeted training. They base their scheme on optimization iterations, representing resources, and ensure its parameter-free nature for wide applicability across varying constraints. For their model selection, given the limitations of 6GB GPU memory and a total training time of 9 hours without pretrained weights, they opt for the seresnext26t_32x4d [24] model, chosen from various lightweight candidates. Their training scheme includes resizing input to 160×160, applying mixup during training, and excluding label smoothing.

A significant highlight is their learning rate scheduler, an integral component in budgeted training. Drawing inspiration from the idea of tuning learning rates for specific budgets rather than employing early-stopping, they calculate training time for one epoch and estimate total epochs within the challenge's given budget. Their parameter-free scheduler decays learning rates linearly from a predefined value (0.001 in their setting) to 0 as the epoch progresses, aiming to maximize performance under budget constraints.

The team also employs data quality enhancement strategies. After training a baseline model, they identify and remove noisy data points by evaluating the confidence scores of classification results. This ensures a cleaner training set. Furthermore, they employ test-time augmentation, wherein they flip and rotate test images, pass them through the model, and average the predictions for improved accuracy.

4.2. Inference track

In this section, we present the baseline solution as well as the summary of top participating teams for the Inference Challenge Track.

Baseline. The baseline model was chosen to be MobileNetV2 [44] which was trained using a cosine decay learning rate scheduler with a warmup of 2 epochs. The model was trained for 23 epochs and then inferenced on RTX 8000 respecting the challenge constraints. The baseline model got 35.25% accuracy and an inference time of 1.16 minutes thereby getting a final score of 759.71.

Team xNN. The team used *Data Processing*, *Training strategies*, *Engineering acceleration* and *Model Selection* techniques to achieve first place in the competition. During the training phase, they meticulously optimized hyperparameters using grid search to impact training outcomes significantly. They used the Adam optimizer with a base learning rate of 0.002 and a cosine learning rate decay. Training took place over 120 epochs, with 5 epochs dedicated to warm-up, distributed across 8 P100 GPUs. Each GPU had a minibatch size of 16.

The team was particularly concerned about overfitting and undertook multiple strategies to address it. They leveraged data augmentation, utilized pre-trained weights from ImageNet-1K for initialization, applied regularization techniques including dropout, weight decay, and L2-Norm, and implemented label smoothing with a coefficient of 0.05. They trained an EfficientNet-B0 [47] model over 120 epochs as their baseline and evaluated the various strategies' effectiveness based on this model.

Furthermore, in the engineering acceleration domain, the team employed several techniques to reduce inference time. They used half-precision inference, which proved compati-

ble with FP16 in PyTorch, reducing inference time by 35%-65% while maintaining accuracy. They optimized memory scheduling by setting "pin_memory=True" for the dataloader and utilizing non-blocking memory transfers from CPU to GPU. Additionally, they shifted data preprocessing tasks from CPU to GPU. These engineering-level accelerations collectively contributed to substantial reductions in inference time while preserving classification accuracy.

They selected the best model based on the public leaderboard, and found EfficientNetv2_B0 [48] trained with image resolution of 1024 to be working best.To further optimize the inference time they tried out different inference resolutions used 512 resolution in the final submission.

Team HSC. The team's approach for the Budgeted Model Inference challenge focused on optimizing the trade-off between accuracy and inference time using limited resources. To achieve this, they initially considered lightweight backbone models that strike a balance between accuracy and speed, specifically YOLOv6 [32] and ReXNet [19]. They evaluated these models based on inference speed and accuracy, utilizing an NVIDIA GeForce RTX 3080 GPU for speed measurements. After evaluating the trade-off, they selected the YOLOv6m model as it yielded the highest score.

To enhance accuracy, the team employed various methods. They utilized augmentation techniques, particularly the bitwise not operation for color inversion, which proved effective given the challenge's constraints. Unlike traditional augmentations, this unique approach suited their dataset and resources. To prevent overfitting, augmented data was separated into a validation set, ensuring accurate performance measurement. Moreover, the team leveraged Automatic Mixed Precision (AMP) to optimize training speed and memory usage. By employing AMP, they increased image size and batch size, leading to more stable convergence. They also considered half-precision for inference, aligning their approach with the eventual deployment scenario.

In order to further enhance the inference time of their approach, the team implemented several strategies: They harnessed the advantages of the Pin_memory option available in the PyTorch framework's DataLoader. This technique led to a notable improvement in data loading speed, resulting in a roughly 13% reduction in inference time. The team focused on enhancing the speed of image reading by switching from the PILLOW library to the cv2 function. This transition led to a 2% improvement in the overall image reading process. Leveraging half-precision for inference was another pivotal move. This decision brought about a substantial reduction in inference time of approximately 29%.

Team FTL. The team experimented with images to understand their basic properties. They undertook experiments involving image resizing. Their observations indicated a trade-off between image size and accuracy: as image size increased, accuracy improved, but inference time also increased. As the provided data consisted of black and white 1-channel images, they transitioned the input to 1 channel. This modification, resulted in significantly reduced inference time by almost half while maintaining similar accuracy. After careful evaluation, the team settled on the 512 image resolution model due to its substantial reduction in inference time at that size.

For augmentation, the team approached this aspect with great care, recognizing challenges related to label transformation and sensitivity to noise. To overcome these concerns, they employed bitwise augmentation, a technique that randomly replaces black and white pixels in each iteration. The results demonstrated that bitwise augmentation led to an overall increase in accuracy, thereby validating its efficacy.

Another strategy the team employed was knowledge distillation, a technique commonly utilized to enhance lightweight model performance. They applied knowledge distillation in this context and observed slight improvements in accuracy.

Team Ganzoo. The team introduces a downsizing method that capitalizes on a straightforward yet effective pixel unshuffle technique and 1×1 convolutions for the classification task. This approach is augmented by integrating the MobileNet V3 [28] large model to classify the sum of digits. Their downsizing procedure is noteworthy for its ability to reduce the input image dimensions by half, employing a tensor rearrangement process. By converting the tensor from shape $(*, C, H \times r, W \times r)$ to $(*, C \times \hat{r}, H, W)$, where '$r$' represents the downsizing factor, they achieve notable image dimension reduction. This method, similar to the pixel shuffle process, is versatile and works well in many areas, also highlighted in efficient super-resolution tasks [16].

Moreover, the team's decision to utilize only the Y channel, instead of the RGB image, showcases their effective solution for handling input data. They start with an input image size of (1, 1,024, 1,024), and after applying the pixel unshuffle method, the image dimensions transform into (4, 512, 512). Subsequently, the incorporation of a 1×1 convolution layer in the downsizing process effectively reduces the channel count to (3, 512, 512).

To tackle overfitting concerns, the team employs the weight decay and embraces comprehensive model training using the entire dataset. Their model training unfolds in two distinct steps: the scratch training step and the fine-tuning step. In the scratch training step, they train the model from the ground up, downsizing input images by a factor of 2 and employing an Adam optimizer with a learning rate of

1e-3. Cross-entropy loss drives the training process, complemented by a cosine warm-up scheduler and 300 total epochs. In the second step, they fine-tune the model using weights obtained from the first step. They used cosine warm-up scheduler with 4 cycles with an initial learning rate of 1e-4 spanning 100 epochs.

Furthermore, to reduce the inference time, the team experimented with structure pruning using DepGraph. While this approach aimed to prune network parameters by 50%, the team noted a decline in accuracy, prompting them to maintain the training settings from the second step.

Team IIT Dhanbad. The team's initial methodology involved the generation of a dataset utilizing MNIST digits, coupled with the concurrent extraction of bounding box information to facilitate YOLOv7 [52] model training. This yielded a notable accuracy of 90.11%. However, the potential for improved classification accuracy became apparent. Subsequently, a dedicated re-classification module was introduced. This module required training a specialized re-classifier on the MNIST dataset, enhancing the classification of digits identified by the YOLO classifier. Remarkably, employing EfficientNet B1 [49] achieved an impressive 96.75% accuracy, while ResNet50 [21] attained 96.6%. Despite the improvements in accuracy, the increased time taken during inference prompted a study into optimization methods.

An alternative strategy emerged - training a classifier on all 28 classes - which effectively reduced inference time, resulting in a 72.82% accuracy using EfficientNet B3. This unanticipated triumph underscored the potential of unconventional methodologies in UltraMNIST inference optimization. To further address inference time challenges, the team resorted to resizing images to (512, 512) dimensions and fine-tuning for accuracy, capitalizing on EfficientNet B3. This tactical adjustment sought to strike a harmonious balance between computational efficiency and accuracy alignment, yet revealed opportunities for further enhancement.

Subsequently, the team implemented network slimming, involving a 50% pruning of EfficientNet B3. Pruning was executed based on the rescaling factor (gamma) of each channel in BatchNorm layers, with the lowest 50% pruned. The approach led to a notable decrease in time and achieved an accuracy of 83%, making it the most effective strategy. Additionally, the team explored EfficientNet B2 and B1 but the results were unsatisfactory.

5. Discussions

In the competition, we noticed participants employed various strategies to enhance their leaderboard scores. Models became more efficient through a set of focused methods. For instance, dynamic learning rate scheduling was used to speed up convergence. Other techniques, like early stopping based on validation performance, helped to avoid overfitting while saving time. Data augmentation techniques such as rotations and flips were also common, aiding in model generalization. Simplified architectures and batch normalization have further streamlined training.

Transfer learning was frequently used to accelerate model training, especially by employing various pre-trained backbones. It was observed that in resource-constrained environments, the most effective backbones may not be those highly rated in academic literature. Instead, a balance needs to be struck between accuracy and other performance metrics like memory usage and training speed. Multi-resolution training and hyperparameter optimization were also leveraged, along with gradient checkpointing and knowledge distillation, to make training more efficient.

While submissions have shown a commendable level of novelty, most still rely on traditional techniques for improving model performance in resource-limited settings. The utilization of emerging, more advanced techniques remains largely unexplored. For future challenges, it is crucial to incorporate these newer methods into the problem set. Doing so will diversify the range of solutions and offer participants an avenue to experiment with groundbreaking techniques, possibly leading to advancements that could transform resource-constrained deep learning.

5.1. Future Directions

Newer methods have emerged for training deep learning models in resource constrained settings. We discuss several promising future directions.

Training Large Models on Smaller GPUs. The use of patchwise training schemes, illustrated by methods like Patch Gradient Descent [18], opens up possibilities for improving deep learning model performance. Instead of processing an entire image in one go during training, this method divides the image into smaller patches, which are then used for model updates. This alteration from the usual practice offers several advantages for model training.

A key benefit of patchwise training lies in its ability to manage memory constraints, particularly for large models and extensive datasets. Traditional training methods require the entire image and its corresponding gradients to be stored in memory for each training iteration. In contrast, focusing on individual patches considerably reduces memory usage. This allows for larger batch sizes within the same memory limits, enhancing both training speed and model performance. By allowing for larger batches, this method also adds diversity to gradient estimation, which could potentially improve the model's ability to generalize.

Leveraging Enhanced Fine-Tuning Methods. Recent de-

velopments in Model Adaptation techniques, such as Parameter Efficient Fine-Tuning (PEFT), offer effective solutions to overcome time and memory constraints when adapting large deep learning models [23, 12, 7]. These methods diverge from traditional weight-updating mechanisms by incorporating adapters—supplementary modules—into the existing model structure. These adapters are trained to specialize the model for new datasets through transfer learning, without altering the core model parameters. This approach effectively mitigates memory limitations, thereby facilitating the deployment of advanced architectures in resource-constrained environments.

A notable advantage of model adaptation techniques like PEFT is their fast convergence. These methods utilize the existing knowledge encapsulated in the core model parameters and fine-tune them via adapters for specialized tasks or datasets, resulting in a quicker convergence time. This feature is particularly useful in resource-limited scenarios, where traditional fine-tuning approaches often require extended periods to converge.

Developing Efficient Models. VanillaNet is an example of a promising approach to crafting efficient deep learning models specifically designed for environments with memory and time constraints [9]. In a landscape dominated by increasingly complex neural architectures, VanillaNet emphasizes the importance of simplicity and efficiency. It avoids the use of excessive depth and intricate operations, such as self-attention modules, focusing instead on streamlined designs that tackle resource limitations effectively. The proposed "deep training" approach, which systematically eliminates non-linear layers while maintaining the networks' performance, embodies adaptability and optimization tailored for constrained settings.

5.2. Conclusion

The challenges presented at the RCV 2023 workshop have uncovered an intriguing landscape. The solutions submitted to date only begin to address the variety of ways to tackle the problem of resource-limited model training and inference. The diversity of approaches demonstrates the complexity of the issue and highlights the varied strategies employed by researchers. However, as noted in the future directions section, much remains unknown, leaving numerous opportunities unexplored.

Reflecting on the current state of the challenge, it becomes clear that arriving at a comprehensive solution will require a long-term commitment with creativity from the entire community. Future iterations of the challenge will focus more on this aspect, encouraging participants to dive deeper into areas yet to be explored. Through subsequent RCV challenges, the aim is to draw closer to resolving the difficulties associated with resource-limited model training and inference, thereby advancing our collective understanding of this significant issue.

References

[1] Bilge Acun, Matthew Murphy, Xiaodong Wang, Jade Nie, Carole-Jean Wu, and Kim Hazelwood. Understanding training efficiency of deep learning recommendation models at scale, 2020. 1

[2] Kim Albertsson, Piero Altoe, Dustin Anderson, Michael Andrews, Juan Pedro Araque Espinosa, Adam Aurisano, Laurent Basara, Adrian Bevan, Wahid Bhimji, Daniele Bonacorsi, et al. Machine learning in high energy physics community white paper. In *Journal of Physics: Conference Series*, volume 1085, page 022008. IOP Publishing, 2018. 1

[3] Mohammad Bavarian, Heewoo Jun, Nikolas Tezak, John Schulman, Christine McLeavey, Jerry Tworek, and Mark Chen. Efficient training of language models to fill in the middle, 2022. 2

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars, 2016. 1, 2

[5] Nadia Brancati, Giuseppe De Pietro, Daniel Riccio, and Maria Frucci. Gigapixel histopathological image analysis using attention-based neural networks. *IEEE Access*, 9:87552–87562, 2021. 2

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. 2

[7] Arnav Chavan, Zhuang Liu, Deepak Gupta, Eric Xing, and Zhiqiang Shen. One-for-all: Generalized lora for parameter-efficient fine-tuning, 2023. 2, 8

[8] Arnav Chavan, Zhiqiang Shen, Zhuang Liu, Zechun Liu, Kwang-Ting Cheng, and Eric Xing. Vision transformer slimming: Multi-dimension searching in continuous optimization space. 2022. 3

[9] Hanting Chen, Yunhe Wang, Jianyuan Guo, and Dacheng Tao. Vanillanet: the power of minimalism in deep learning, 2023. 8

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[11] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm. int8 (): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022. 2

[12] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023. 2, 8

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[14] Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16091–16101, 2023. 3

[15] Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. 2023. 3

[16] Ganzorig Gankhuyag, Kihwan Yoon, Jinman Park, Haeng Seon Son, and Kyoungwon Min. Lightweight real-time image super-resolution network for 4k images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1746–1755, June 2023. 6

[17] Deepak K Gupta, Udbhav Bamba, Abhishek Thakur, Akash Gupta, Suraj Sharan, Ertugrul Demir, and Dilip K Prasad. Ultramnist classification: A benchmark to train cnns for very large images. *arXiv preprint arXiv:2206.12681*, 2022. 3

[18] Deepak K. Gupta, Gowreesh Mago, Arnav Chavan, and Dilip K. Prasad. Patch gradient descent: Training neural networks on very large images. In *arxiv*, 2023. 7

[19] Dongyoon Han, Sangdoo Yun, Byeongho Heo, and YoungJoon Yoo. Rethinking channel dimensions for efficient model design. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 732–741, 2021. 6

[20] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A. Horowitz, and William J. Dally. Eie: Efficient inference engine on compressed deep neural network, 2016. 2

[21] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 7

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[23] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. 2, 8

[24] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 5

[25] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022. 2

[26] Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. Lion: Adversarial distillation of closed-source large language model. *arXiv preprint arXiv:2305.12870*, 2023. 3

[27] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. On architectural compression of text-to-image diffusion models. *arXiv preprint arXiv:2305.15798*, 2023. 3

[28] Brett Koonce and Brett Koonce. Mobilenetv3. *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pages 125–144, 2021. 6

[29] François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. Block pruning for faster transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10619–10629, 2021. 3

[30] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and gpu-computation efficient backbone network for real-time object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019. 4

[31] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13906–13915, 2020. 4

[32] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, et al. Yolov6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022. 6

[33] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. *arXiv*, 2023. 2, 3

[34] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2

[35] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, dec 2017. 1

[36] Shih-yang Liu, Zechun Liu, and Kwang-Ting Cheng. Oscillation-free quantization for low-bit vision transformers. 2023. 2

[37] Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023. 2

[38] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2018. 2

[39] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023. 3

[40] Hans Pinckaers, Bram van Ginneken, and Geert Litjens. Streaming convolutional neural networks for end-to-end

learning with multi-megapixel images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1581–1590, mar 2022. 2

[41] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949, 2021. 3

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 2

[43] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 3

[44] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 5

[45] Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in Neural Information Processing Systems*, 33:20378–20389, 2020. 2, 3

[46] Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*, 2023. 3

[47] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 5

[48] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10096–10106. PMLR, 18–24 Jul 2021. 6

[49] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 7

[50] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 3

[51] Antoine Vanderschueren and Christophe De Vleeschouwer. Are straight-through gradients and soft-thresholding all you need for sparse training? In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3808–3817, 2023. 3

[52] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023. 7

[53] Yu Emma Wang, Gu-Yeon Wei, and David Brooks. Benchmarking tpu, gpu, and cpu platforms for deep learning, 2019. 2

[54] Zhenyu Wang, Hao Luo, Pichao Wang, Feng Ding, Fan Wang, and Hao Li. Vtc-lfc: Vision transformer compression with low-frequency components. *Advances in Neural Information Processing Systems*, 35:13974–13988, 2022. 3

[55] Ross Wightman, Hugo Touvron, and Hervé Jégou. Resnet strikes back: An improved training procedure in timm. *arXiv preprint arXiv:2110.00476*, 2021. 4

[56] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 38087–38099. PMLR, 23–29 Jul 2023. 2

[57] Chong Yu, Tao Chen, Zhongxue Gan, and Jiayuan Fan. Boost vision transformer with gpu-friendly sparsity and quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22658–22668, 2023. 3

[58] Chen Zhang, Di Wu, Jiayu Sun, Guangyu Sun, Guojie Luo, and Jason Cong. Energy-efficient cnn implementation on a deeply pipelined fpga cluster. In *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, August 2016. 2

[59] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2736–2746, 2022. 4