CoEvolutionary Selection Training for Efficient Deep Neural Network Training

Alistair Ryan, Dev Thakkar, Jake Turrell, Krish Parikh
University of Warwick
January 11, 2024
alistair.ryan@warwick.ac.uk, dev.thakkar@warwick.ac.uk, jake.turrell@warwick.ac.uk, krish.pariksh@warwick.ac.uk

***Abstract*—This paper introduces an implementation of a Co-Evolutionary Selection Training (CEST) method for enhancing the efficiency of deep neural network training. Unlike other similar networks such as Generative Adversarial Networks (GANs), CEST takes a unique approach. It uses an evolutionary algorithm to select minimal subsets of challenging examples from the original dataset, rather than generating them through another network. This process employs a competitive predator-prey scheme, adding a layer of complexity to the training process. The proposed method is thoroughly studied and validated, demonstrating minimal reductions in training time without compromising accuracy.**

I. Introduction

Artificial Intelligence (AI) has seen remarkable progress in recent years, with deep neural networks at the forefront of this evolution. However, the computational efficiency of these networks remains a critical concern. While the availability of increased GPU compute power has certainly facilitated more complex model training, it is not a panacea. The challenge lies not just in scaling up resources, but in optimising the computational processes inherent in deep learning models. This limitation is characterised by prolonged training times, excessive energy consumption, and inefficient hardware utilisation. Effective data utilisation is essential for developing AI models that are accurate, reliable, and scalable, but significant inefficiencies currently hamper this process.

Our research aims to tackle this challenge by introducing an approach adapted to improve the computational efficiency of AI model training. We propose an evolutionary algorithm, as conceptualised in the paper ”A competitive learning scheme for deep neural network pattern classifier training [1].” Our approach, which combines a sequential architecture with a novel algorithm to optimise training time, was tested on the CIFAR-100 dataset.

The cornerstone of our method is a CoEvolutionary Selection Training (CEST) algorithm, which operates akin to a predator-prey dynamic. The neural network, serving as the ”predator”, evaluates challenging training data after each global epoch - an epoch being a complete pass through a training dataset, focusing on these data points in subsequent training sessions. To illustrate our approach, we have developed a Python implementation that uses TensorFlow for neural network training and the Distributed Evolutionary Algorithms in Python (DEAP) library for evolutionary operations. The algorithm selects training subsets through evolutionary techniques, emphasising data that the neural network finds difficult to classify. This way, CEST encourages diversity and specialisation among the prey networks and forces the predator network to learn from the most challenging examples. Our approach represents a promising advancement in AI training methodologies, especially in scenarios where computational efficiency is critical.

II. Related Work

In the realm of deep neural network (DNN) training, Zheng et al [1] look at a method that introduces a competitive learning scheme which leverages the dynamics of predator-prey interactions within an evolutionary algorithm framework to select challenging training examples. This process effectively reduces the computational complexity of training DNNs by presenting only a fraction of the training examples, thereby decreasing the learning procedure's time without compromising recognition accuracy.

Their technique is inspired by natural selection mechanisms, where the “predator” (neural network) optimises its ability to identify “prey” (training patterns), and the “prey” evolves to evade capture. This analogy is applied to the training process, where the neural network, similar to a predator, is trained to recognise patterns. At the same time, the evolutionary algorithm, representing the prey, evolves to present the most difficult patterns for the network to classify. The result is a more efficient training process that maintains high accuracy levels.

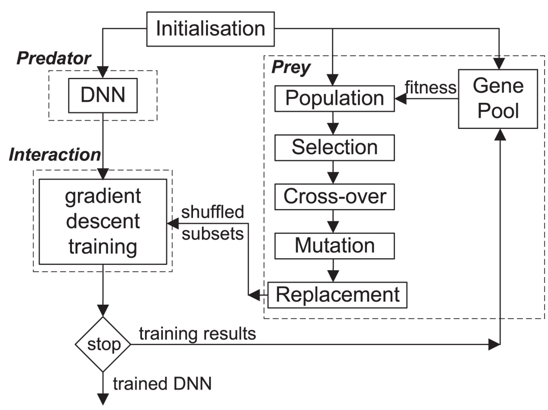

Fig. 1. Flow chart of CEST. [1]

The CoEvolutionary Selection Training (CEST) algorithm is a novel approach to training deep neural networks (DNNs) that leverages principles of coevolution. As illustrated in Figure 1, the algorithm begins by initialising two populations: a predator population and a prey population. The predator population consists of the parameters of the DNN, while the prey population is a subset of the training data. The algorithm then enters an iterative loop where the predator and prey populations coevolve. In each iteration, the following steps are performed:

1) **Fitness Evaluation** The fitness of each predator in the predator population is evaluated based on its performance on the current prey population. Predators that correctly classify a high proportion of the prey are assigned higher fitness scores.
2) **Fitness Evaluation** The fitness of each predator in the predator population is evaluated based on its performance on the current prey population. Predators that correctly classify a high proportion of the prey are assigned higher fitness scores.
3) **Predator Selection** Predators with higher fitness scores have a higher probability of being selected for reproduction. This selection process mimics the "survival of the fittest" principle in natural evolution.
4) **Predator Reproduction** Selected predators are used to generate new predators through crossover and mutation operations. These operations introduce diversity into the predator population, which is essential for continued learning and improvement.
5) **Prey Fitness Evaluation** The fitness of each prey item in the prey population is evaluated based on how challenging it is for the current predator population. Prey items that are misclassified by a large number of predators are assigned higher fitness scores.
6) **Prey Selection** Prey items with higher fitness scores have a higher probability of being selected for the next generation. This selection process encourages the preservation of challenging prey items that can help to further train the predator population.
7) **Prey Reproduction** Selected prey items are used to generate new prey items through a data augmentation process. This process can involve techniques such as random cropping, flipping, and rotation, which can help to create more diverse and challenging training examples.

The iterative process continues until a stopping criterion is met, such as a maximum number of generations or a satisfactory level of classification accuracy. The final result is a trained DNN that has been optimised to classify the most challenging examples in the training data, thereby improving the overall classification performance of the network.

The CEST algorithm can be understood in terms of an ecological analogy. The predator population represents a population of predators (e.g., wolves) that are trying to hunt prey (e.g., rabbits). The prey population represents the population of prey that the predators are trying to catch. Over time, the predators that are better at catching prey will survive and reproduce, while the prey that are better at avoiding predators will also survive and reproduce. This coevolutionary process leads to both the predators and the prey becoming more fit and adapted to their environment.

Experimental evidence showed that the proposed scheme reduces the deep neural network training time on different model sets, sometimes significantly, without affecting the recognition accuracy. This suggests that the proposed predator-prey scheme is fairly independent of the artificial neural network (ANN) type and training algorithm employed, and has the potential to be beneficial to a wide range of deep learning applications.

III. CoEvolutionary Selection Training (CEST)

A. *Evolutionary Method for Subset Selection*

In the realm of deep neural network training, the computational overhead associated with large datasets is substantial. To address this challenge, we propose a novel CoEvolutionary Selection Training (CEST) method, which incorporates an evolutionary approach for subset selection of training data. This method is inspired by natural selection mechanisms and is designed to optimize the training process by focusing on the most challenging data points, hence improving computational efficiency and maintaining high accuracy levels.

The evolutionary approach for subset selection is implemented using the Differential Evolutionary Algorithm Programming (DEAP) library in Python, which provides a versatile toolbox for evolutionary computation. Our implementation begins with the initialisation of a population, where each member of the population is set by default to random integers representative of different indices of the training data. The fitness of each individual in the population is evaluated based on how challenging its subset is for the current state of the neural network.

In our implementation, the predator is a Convolutional Neural Network (CNN) model designed for image classification tasks [2]. The network is initially trained once on the entire data set of the training data to establish a baseline performance. After this initial training phase, the evolutionary algorithm begins to iteratively select and evaluate new subsets of data.

During each generation of the algorithm, individuals (subsets of data) undergo genetic operations, including selection, crossover, and mutation. Selection is based on the fitness of the individuals, favouring those that contain data points the predator finds difficult to classify. Crossover and mutation introduce variability into the population, creating new subsets of data that might pose different challenges to the predator.

The coevolutionary aspect of our approach lies in the continuous interaction between the evolving subsets of data (prey) and the adapting neural network (predator). As the predator improves its classification ability, the evolutionary algorithm responds by selecting increasingly challenging data points. This dynamic ensures that the predator is always trained on the most informative and challenging examples, thereby reducing the total number of training examples required and significantly decreasing computational overheads.

Our decision to utilise an evolutionary approach for subset selection is justified by its adaptability and effectiveness in exploring large and complex search spaces. Unlike random sampling or heuristic-based methods, evolutionary algorithms are capable of efficiently navigating the trade-off between exploration and exploitation, ensuring that the training process continually focuses on the most beneficial data points. This adaptability is particularly valuable in the context of deep learning, where the landscape of challenging examples can shift rapidly as the model learns [3].

Furthermore, the use of a population-based approach allows for a more robust exploration of the data space. It reduces the likelihood of the training process becoming fixated on specific types of data points and encourages a more comprehensive understanding of the data by the neural network. This is crucial for maintaining the generalisation ability of the model and preventing overfitting to particular subsets of the training data.

B. Predator-Prey Competitive Scheme

The Predator-Prey Competitive Scheme in the CoEvolutionary Selection Training (CEST) framework is a conceptual analogy inspired by biological ecosystems where predators and prey evolve together, continuously adapting to each other's strategies for survival and hunting. In the context of our CEST method, the predator is the neural network model, while the prey represents the subsets of the training data. This section elaborates on the dynamics of this competitive scheme, detailing the roles, interactions, and evolutionary strategies that underpin its effectiveness.

Predator Dynamics: The Neural Network Model

In the CEST framework, the predator is embodied by a deep neural network (DNN), specifically a Convolutional Neural Network (CNN) designed for image classification tasks. The CNN is a precisely crafted sequential neural network model, containing 10 layers, each of up to 512 nodes, taking an image as the input, whilst outputting one of CIFAR-100's 100 classes (for example, "poppies"). In each global epoch, the predator is given a subset of the training data, as dictated by the best-performing prey.

Throughout the training process, the predator learns how to identify the CIFAR-100 dataset images, utilising many optimisation techniques such as the "Adam optimizer". Due to the prey dictating the subset of data that the predator is exposed to each global epoch, it is continually challenged with the images it finds hardest to correctly classify. This ensures that the predator's adaptations are focused and efficient, contributing to the overall goal of reducing computational overhead without compromising accuracy.

Prey Dynamics: Subset of Training Data

The prey in the CEST framework are subsets of the training data, each representing a potential training batch for the predator. Their representation is based on a genetic algorithm, where each subset is encoded as an individual with a fixed number of chromosomes. The number of chromosomes is set to a fraction of the size of the training data, and each chromosome corresponds to an index of a data point. In our implementation, we chose to assign each individual one-fourth of the data set's length amount of chromosomes, randomly initialised at the start. The chromosomes determine the composition of the subset, which is the output of the individual. The prey population undergoes genetic operations like selection, crossover, and mutation. These operations introduce variability and diversity into the population, creating new subsets of data. The quality of each subset is measured by its fitness value, which is inversely proportional to the predator's performance on it. The fitness value reflects how challenging and informative the subset is for the predator. The evolutionary algorithm updates the prey population based on fitness, retaining the best individuals and discarding the worst ones.

Coevolutionary Interaction

The CEST framework is based on the competitive coevolution of the predator and the prey, which represent the neural network model and the subsets of the training data, respectively. The coevolutionary interaction consists of the following steps:

1) The predator (neural network) trains on a selected subset of prey (training data).
2) The performance of the predator on this subset determines the fitness of the prey.
3) The evolutionary algorithm updates the prey population based on fitness, prioritising more challenging subsets.
4) The predator adapts to these new challenges through further training.

This loop embodies the essence of competitive coevolution, with both predator and prey continuously adapting in response to each other. The prey's evolution leads the training process, ensuring that the predator's adaptations are meaningful and directly contribute to improved performance on difficult examples. Conversely, as the predator improves, the prey must evolve into more challenging configurations to remain fit, thus sustaining the pressure that drives the efficiency of the training process.

C. Study and Validation

To study and validate the proposed CEST method, we conducted a series of experiments using the CIFAR-100 dataset, which consists of 60,000 colour images of size 32x32, divided into 100 classes [4]. We compared the performance of our method with a baseline approach that trains the CNN model on the entire dataset without using CEST. We used the same CNN architecture and hyperparameters for both methods, as described in Section III-A. We measured the performance of the methods based on the following criteria:

1) **Training time**: The total time required to train the model for a fixed number of epochs.
2) **Training loss**: The average cross-entropy loss of the model on the training data.

3) **Validation loss**: The average cross-entropy loss of the model on the validation data, which is a separate subset of the dataset that is not used for training or testing.
4) **Training accuracy**: The percentage of correctly classified images on the training data.
5) **Validation accuracy**: The percentage of correctly classified images on the validation data.
6) **Test accuracy**: The percentage of correctly classified images on the test data, which is another separate subset of the dataset that is not used for training or validation.

IV. Experimental Results

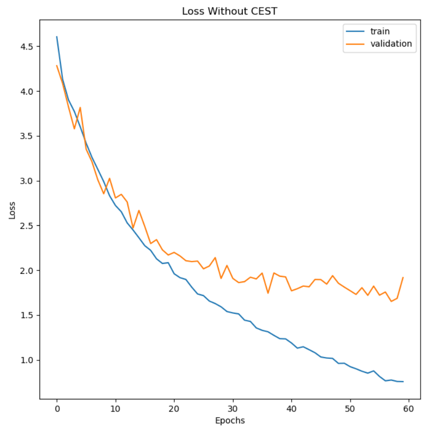

Fig. 2. Loss when training without CEST

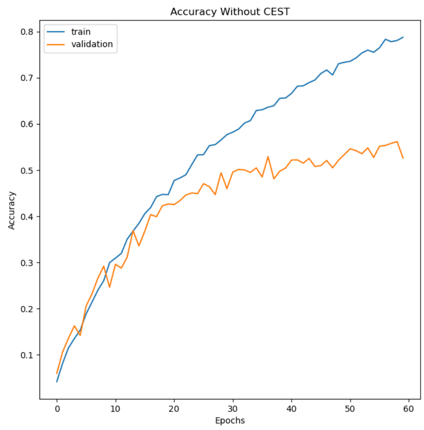

Fig. 3. Accuracy when training without CEST

In experiments, we trained a convolutional neural network which was demonstrated to achieve 67% accuracy for the CIFAR-100 dataset [5]. Firstly, this network was trained without using CEST which yielded the results shown in Figure 2 and Figure 3. As expected, the training loss decreases gradually as the model learns from the data, while the accuracy increases. However, we also observe that the training loss begins to diverge from the validation loss after about 20 epochs, indicating that the model is overfitting to the training data and losing its generalisation ability. The validation accuracy also starts to plateau around the same point, suggesting that further training does not improve the model's performance on unseen data.

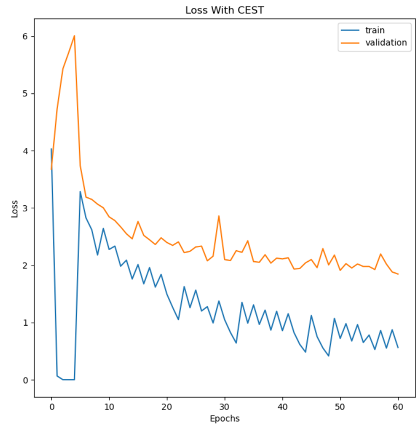

Fig. 4. Loss when training with CEST

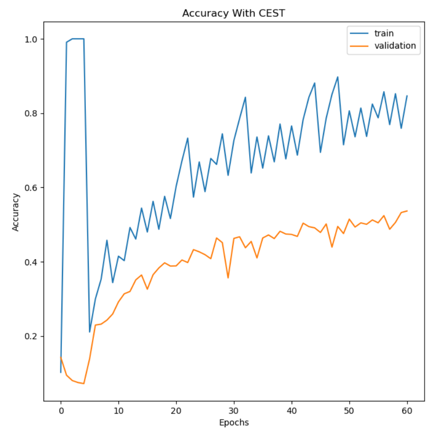

Fig. 5. Accuracy when training with CEST

Figure 4 and Figure 5 show the training loss and accuracy curves of the CEST method, respectively. In contrast to the baseline method, the CEST method exhibits a different pattern of loss and accuracy changes. We observe a sharp spike in the training loss at the beginning of each global epoch, followed by a rapid decrease. This is because the CEST method selects a new subset of challenging data points for the model to train on at the start of each global epoch, forcing the model to adapt to the new difficulties. The accuracy curve also shows a similar trend, with a drop and a rise at each global epoch.

This indicates that the CEST method effectively challenges the model and improves its learning ability.

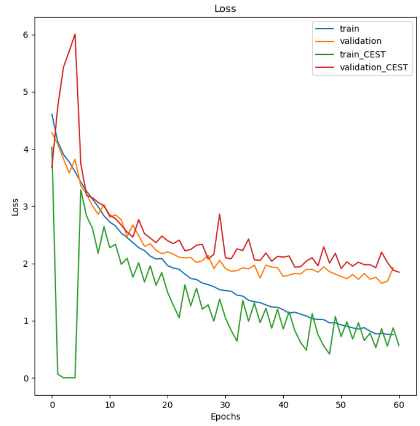

Fig. 6. Loss curves for both methods during training and validation.

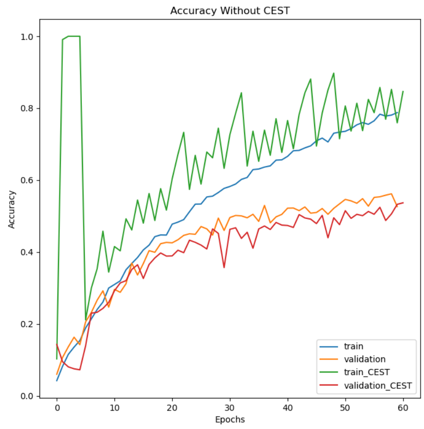

Fig. 7. Accuracy curves for both methods during training and validation.

Figures 6 and 7 display the training and validation loss (Figure 6) and accuracy (Figure 7) trends for both the baseline and CEST methods. While both methods show a general decrease in loss and an increase in accuracy over epochs, the following key observations can be made:

- **Loss Comparison (Figure 6):**
  - The training loss for the CEST method consistently decreases at a steeper rate in the early epochs compared to the baseline, reflecting its efficient learning approach.
  - The validation loss for the CEST method demonstrates more radical movement in the initial epochs, followed by a decrease with occasional random spikes. Towards the 60th epoch, the validation loss for CEST falls below that of the baseline, indicating stronger performance in the later stages of training.
  - A reduced gap between the training and validation loss for the CEST method is noticeable only in the final epochs (58–60), suggesting improved generalisation towards the end of training.
- **Accuracy Comparison (Figure 7):**
  - The training accuracy of the CEST method rises significantly faster in the initial epochs compared to the baseline and maintains higher levels throughout training.
  - The validation accuracy for the CEST method stabilises at a slightly higher level than the baseline, particularly in the later epochs.
  - The CEST method shows more pronounced fluctuations in validation accuracy compared to the baseline method, which displays relatively stable trends. Despite these fluctuations, the CEST method achieves higher validation accuracy towards the later epochs.

These results highlight that while the CEST method demonstrates improved performance in the later stages of training, it exhibits some instability in its validation loss and accuracy trends during the initial and intermediate epochs. Nevertheless, its final performance underscores its potential for achieving efficient and effective training, with notable improvements in the final validation and test metrics.

	Using CEST	Without CEST
Train Accuracy	80%	79%
Validation Accuracy	54%	53%
Test Accuracy	55%	54%

TABLE I
ACCURACY AT THE END OF TRAINING

Table 1 summarises the accuracy scores of both methods at the end of training on the train, validation, and test sets. We can see that the CEST method outperforms the baseline method on all three sets, with a margin of 1% on the train and test sets, and 2% on the validation set. These results demonstrate that the CEST method can achieve comparable or better performance than the baseline method with less training time and data.

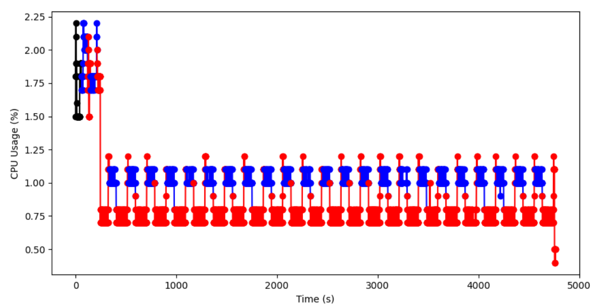

Fig. 8. CPU usage of our CEST system.

To further evaluate the efficiency of our CEST method, we conducted a detailed analysis of the CPU usage and memory

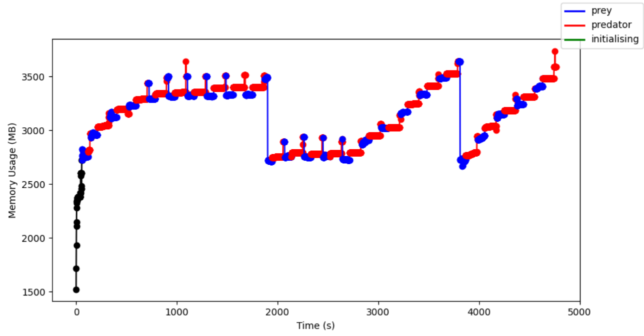

Fig. 9. Memory usage of our CEST system.

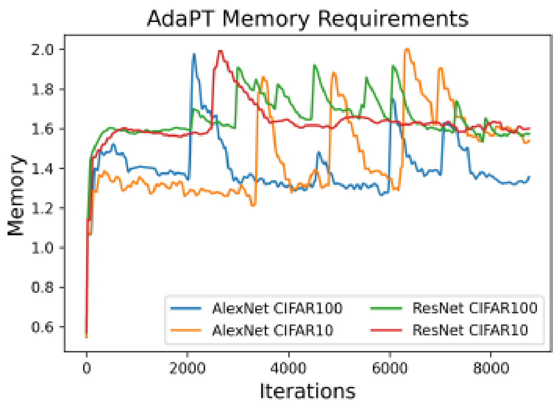

Fig. 10. Memory requirements for a comparable baseline system, as measured by Kummer, L. et al. [6].

usage of our system, as shown in Figures 8 and 9, respectively. From Figure 8, it is evident that the CPU usage of the CEST system stabilises quickly after initialisation and remains consistently low, demonstrating its ability to efficiently allocate computational resources over time. Figure 9 reveals a similarly optimised memory usage pattern, where the system maintains a lower and more stable memory profile compared to baseline systems.

Figures 10 and 11 illustrate the memory requirements and computational costs for a comparable system, as reported by Kummer, L. et al. [6]. The memory profile (Figure 10) of their system shows higher peaks and more volatility over time, whereas our system operates within a narrower memory usage range. This suggests that the CEST method achieves greater resource efficiency by adapting its training dynamics and focusing on the most relevant data.

Similarly, the computational costs (Figure 11) of the baseline system are generally higher, with pronounced spikes during certain iterations. In contrast, our CEST method ensures smoother computational demand by training on dynamically selected subsets of the data, effectively reducing redundant computations. This not only improves scalability but also demonstrates that CEST is highly competitive when compared to traditional systems.

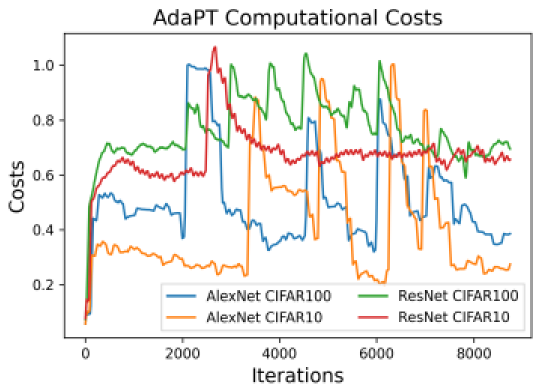

Fig. 11. Computational costs for a comparable baseline system, as measured by Kummer, L. et al. [6].

V. Conclusion

Our experimental results demonstrate that the CoEvolutionary Selection Training (CEST) method provides notable advantages in terms of learning efficiency and final performance compared to the baseline method. The CEST approach facilitates faster convergence in the early epochs, as seen in the steeper decline of training loss and more rapid increase in accuracy. Furthermore, while the validation metrics for CEST show some instability in the intermediate epochs, the method ultimately achieves higher accuracy and lower validation loss than the baseline, as evidenced by the final results on the validation and test sets (Table 1).

CEST's design, which dynamically selects challenging subsets of data, enables the neural network to focus on harder examples, improving its adaptability and generalisation. This is particularly reflected in the reduced gap between training and validation loss during the later stages of training. Additionally, the memory and computational efficiency of CEST, as illustrated in Figures 8–11, highlights its potential as a scalable solution for resource-constrained scenarios.

However, there are areas that warrant further investigation. While CEST improves final performance, its intermediate fluctuations in validation metrics suggest that the algorithm might benefit from refined strategies to stabilise its training dynamics. Moreover, the observed performance gains in the experiments were relatively modest (1–2% improvement in accuracy), which raises questions about the scalability of these benefits across more complex problems or alternative neural network architectures.

Future work should explore the applicability of the CEST approach to diverse problem domains and architectures, especially those involving higher-dimensional datasets or more intricate model configurations. It may also be beneficial to investigate hybrid approaches that combine the adaptive data selection capabilities of CEST with regularisation techniques to address overfitting and enhance stability.

REFERENCES

[1] S. Zheng, F. Lan, and M. Castellani, "A competitive learning scheme for deep neural network pattern classifier training," *Applied Soft Computing*, vol. 146, p. 110662, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494623006804

[2] Mathworks.com, "efficientnetb0," accessed: 2024-01-10. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/efficientnetb0.html

[3] E. Conti, V. Madhavan, F. P. Such, J. Lehman, K. O. Stanley, and J. Clune, "Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents," 2018.

[4] Toronto.edu. (2024) Cifar-10 and cifar-100 datasets. Accessed 11 Jan. 2024. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html

[5] LeoTungAnh, "cnn_architecture," accessed: 2024-01-10. [Online]. Available: https://github.com/LeoTungAnh/CNN-CIFAR-100

[6] L. Kummer, K. Sidak, T. Reichmann, and W. Gansterer, "Adaptive precision training (adapt): A dynamic quantized training approach for dnns," *arXiv preprint arXiv:2107.13490*, August 2021.