National College of Ireland

A Deep Learning Feature-Ranked Backpropagation Framework for Sustainability

MSc Research Project
Artificial Intelligence

Gaurav
Student ID: x23189487

School of Computing
National College of Ireland

Supervisor: Paul Stynes

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Gaurav
Student ID:	x23189487
Programme:	Artificial Intelligence
Year:	2024
Module:	MSc Research Project
Supervisor:	Paul Stynes
Submission Due Date:	12/12/2024
Project Title:	A Deep Learning Feature-Ranked Backpropagation Framework for Sustainability
Word Count:	5369
Page Count:	21

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	29th January 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

A Deep Learning Feature-Ranked Backpropagation Framework for Sustainability

Gaurav
x23189487

Abstract

Backpropagation method is used for training neural networks, it adjusts weights through error feedback from layer wise predictions, improving the accuracy of the model over epochs. Feature-ranked backpropagation prioritizes features during training by adjusting the weights based on their influence while masking the weights with less influence on the output to improve learning efficiency. Sustainability in deep learning refers to the process of developing artificial intelligence (AI) models that reduce computational resource requirements such as reducing computational time requirements(from 1 hour to 45 minutes). This paper proposes a novel framework combining Feature-Ranked Backpropagation with established deep learning architectures such as AlexNet, ResNet-18, and VGGNet-19 to reduce the use of resources such as training time, computational requirements(RAM, GPU, and Processors), energy consumption and CO_2 emissions. The proposed method dynamically prioritizes the most critical weights for updates, guided by feature-ranking techniques derived from gradient-based saliency maps. The models trained on the subset of ImageNet dataset containing 66,433 images across 50 categories for image classification. The framework achieves notable reductions up to 20% savings in energy usage and 15% in training times compared to standard training approaches, with comparable accuracy metrics. This paper lays foundation for training deep learning models efficiently and sustainably.

Keywords - Feature Ranking, Backpropagation, Deep Learning, AlexNet, ResNet, VGGNet, Image Classification, Sustainability, Training Efficiency.

1 Introduction

Numerous fields have revolutionized with deep learning at its back, particularly in computer vision where convolutional neural networks (CNNs) have become the de facto standard for image classification tasks. First of its kind architectures such as AlexNet (Krizhevsky et al., 2012), ResNet (He et al., 2016), and VGGNet (Simonyan and Zisserman, 2014) have shown exceptional performance on benchmark datasets like ImageNet. However, the long training times, high energy consumption, environmental impacts, leading to high computational costs pose significant challenges. Recent studies have emphasized on the need for reducing the carbon footprint of machine learning training process and development of energy-efficient AI practices (Henderson et al., 2020) and (Schwartz et al., 2020).

Backpropagation is the cornerstone of neural network optimization and responsible for successful training of the model, involves updating all weights in the network which often leads to redundant computations. That's where feature-ranking based selective backpropagation takes an edge, which only updates a subset of weights, it emerges as a promising solution to mitigate this inefficiencies.

However, existing heuristic-based weight selection methods, such as those by (Vakil-Baghmisheh and Pavešić, 2004) and (Jiang et al., 2019), can compromise model performance. This research addresses these limitations, it integrates feature-ranking techniques into the selective backpropagation process, hence, prioritizing critical weights during an epoch based on saliency maps derived from gradient information.

The aim of this research is to optimize the training of deep learning models like AlexNet, ResNet, and VGGNet for image classification by integrating feature ranking into the backpropagation process and evaluate its effects on training time, memory requirements, and environmental impact while maintaining performance. To address the research question, the following specific objectives were derived:

Investigate the state of the art in selective backpropagation and feature-ranking techniques for optimizing deep learning training processes. Design a feature-ranking based selective backpropagation framework to prioritize important weights for updates. Implement the proposed framework on CNN architectures, specifically AlexNet, ResNet-18, and VGGNet-19. Evaluate the framework's effectiveness in reducing training time, energy consumption, and CO_2 emissions while maintaining model performance.

By demonstrating how selective backpropagation can optimize training processes without compromising model accuracy, this paper contributes to the growing field of sustainable AI. It addresses key gaps in the literature by providing a scalable and principled approach to reduce the resource costs of deep learning.

This paper is structured as follows: Section 2 reviews the relevant literature on deep learning, feature ranking, and selective backpropagation. Section 3 outlines how the process was thought and what steps were taken to develop the proposed framework. Section 4 describes design specifications of the selective backpropagation framework. Section 5 walks you through how the framework is implemented. Section 6 presents experimental results, including performance metrics, energy analysis and discusses the findings in the context of existing research, highlighting the contributions and limitations. Finally, Section 7 concludes the study and suggests directions for future research.

2 Related Work

The field of optimizing deep learning models for sustainability, computational efficiency, and feature prioritization has become a crucial one in recent years. This section focuses on existing literature while focusing on advancements in deep learning architectures, selective backpropagation, feature ranking, and sustainable AI practices and provides a comprehensive review.

(Krizhevsky et al., 2012) introduced a revolutionary deep convolutional neural network known as AlexNet for image recognition which utilized ReLU activations, dropout and GPU acceleration. The model achieved a top-5 error rate of 15.3% on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), surpassing traditional machine learning methods. However, this highlighted a key inefficiencies in the backpropagation, which was a result of high computational costs of training AlexNet were substantial. This

study uses AlexNet as a baseline model, with selective backpropagation to pacify these training inefficiencies.

(He et al., 2016) proposed ResNet a deep convolutional model architecture, which introduced residual connections to address the vanishing gradient problem. ResNet's scalability enabled the successful training of extremely deep networks, achieving a top-5 error rate of 3.57% on ImageNet. Despite its efficiency in handling gradient flow, the computational cost of training ResNet remained high. This study uses ResNet-18 as a baseline model, with selective backpropagation to improve energy efficiency during training.

(Simonyan and Zisserman, 2014) developed VGGNet architecture, which demonstrated the benefits of deeper architectures with small convolutional filters. Achieving a top-5 accuracy of 92.7% on ImageNet, VGGNet became a standard benchmark for convolutional networks. However, due to its extreme computational demands, the need for more efficient training methods was uncovered. This research introduces VGGNet-19 as a baseline model, applying selective weight updates to dynamically reduce training costs.

(Shrikumar et al., 2017) introduced a gradient-based method for computing feature importance by comparing neuron activations to reference values called DeepLIFT and its computational efficiency and high interpretability makes it a valuable tool to understand predictions of neural networks. This study uses gradient-based saliency maps to rank weights for selective updates during backpropagation, based on the principles of DeepLIFT.

(Fong and Vedaldi, 2017) developed a method to assess feature importance based on perturbations, which took advantage of Taylor expansion to approximate the impact of feature perturbations on predictions. While this process was effective for increased interpretability, it incurred additional computational overhead. The proposed method avoids such inefficiencies by adopting gradient-based saliency methods for weight prioritization.

(Vakil-Baghmisheh and Pavešić, 2004) introduced selective backpropagation for Radial Basis Function (RBF) networks in which they selected weights for updates based on output error. While this approach was effective for small-scale models, it lacked stability and was not applicable to modern large-scale architectures. These foundational findings are extended in this research.

(Jiang et al., 2019) proposed giving importance to weight updates for features with the highest training loss in order to accelerate deep learning. While their approach resulted in reduced training times, but the feature importance was calculated only once. This research addresses this gap by ranking weights using gradient-based saliency maps during each epoch.

(Henderson et al., 2020) focused on importance of sustainability of AI and standardized reporting of enegy consumption and CO_2 emissions while training but it lacked any actionable methodology for reducing the said energy costs. The proposed framework demonstrates significant energy savings through selective backpropagation, aligning with their goals.

Core contributions such as AlexNet (Krizhevsky et al., 2012), ResNet (He et al., 2016), and VGGNet (Simonyan and Zisserman, 2014) have established benchmarks for performance but often at high computational costs. (Vakil-Baghmisheh and Pavešić, 2004; Jiang et al., 2019) developed early phases of selective backpropagation methods but they lacked stability and weight prioritization criteria. More emphasis is on the need for energy-efficient training methodologies and recent efforts were made by (Henderson et al., 2020) to address sustainability of AI but they fall short of providing concrete

solutions.

The proposed research bridges these gaps by integrating feature-ranking based selective backpropagation into AlexNet, ResNet-18, and VGGNet-19. By dynamically updating the importance of a feature during an epoch. The proposed framework reduces training time, energy consumption and memory usage all while maintaining model performance. This work advances the state of the art in energy-efficient deep learning, contributing to the broader goals of sustainable AI development.

Author(s)	**Focus**	**What They Did**	**Results**	**Relevance**
(Krizhevsky et al., 2012)	AlexNet for image recognition	Introduced ReLU, dropout, and GPU acceleration	Achieved 15.3% top-5 error rate on ImageNet	Baseline model for selective backpropagation
(He et al., 2016)	ResNet and residual connections	Enabled deeper networks with residual learning	Achieved 3.57% top-5 error rate on ImageNet	Baseline model with selective updates
(Simonyan and Zisserman, 2014)	VGGNet for image recognition	Developed deep networks with small filters	Achieved 92.7% accuracy on ImageNet	Baseline model for selective updates
(Henderson et al., 2020)	Energy and carbon tracking	Advocated for standardized metrics	Highlighted sustainability challenges	Demonstrates energy savings
(Shrikumar et al., 2017)	Feature ranking (DeepLIFT)	Proposed gradient-based feature importance	Provided efficient and interpretable attributions	Inspired saliency-based ranking
(Fong and Vedaldi, 2017)	Feature importance via perturbation	Used Taylor expansion for interpretability	Improved interpretability at computational cost	Demonstrates feature-ranking benefits
(Jiang et al., 2019)	Selective backpropagation	Prioritized weights with high loss	Reduced training time	Builds upon selective updates with saliency
(Vakil-Baghmisheh and Pavešić, 2004)	Early selective backpropagation	Prioritized weights using output error	Reduced training time in RBF networks	Lays the groundwork for modern methods

Table 1: Summary of reviewed works detailing focus, methodologies, results, and relevance to the current study.

3 Methodology

This study revolves around evaluating a selective backpropagation mechanism based on feature-ranking which aims to improve computational efficiency during training without compromising performance of the model. This section is structured in five phases namely: data gathering, pre-processing, transformation, modeling and implementation, and evaluation and results.

During phase one: A widely recognized dataset used for benchmarking state-of-the-

art models in image classification research known as ImageNet[1] dataset. (Deng et al., 2009) was selected as its diversity ensures the representativeness and generalizability of findings (Simonyan and Zisserman, 2014). A subset of 66,433 images across 50 classes was used from a corpus of 13,31,167 images across 1000 classes, the classes were selected at random to include variety of categories such as animals, objects, and scenes. This was done in order to reduce the computational overhead while maintaining the complexity in classification tasks as training on the entire dataset would require extensive computational resources and time (Russakovsky et al., 2014). This subset was organized into class-labelled directories for compatibility with PyTorch's 'ImageFolder' API to ensure accessibility of the dataset during training and evaluation phases.

During phase two: Data pre-processing is important to prepare data in order to be used effectively for deep learning models (Goodfellow et al., 2016). The dataset was split into training, validation, and testing subsets with 60%, 20%, and 20% ratio respectively to ensure balanced representation across splits. **Table 2** shows the sample of dataset after the split along with their encrypted and real class names.

Real Class Name	**Encrypted Class Name**	**Train Images**	**Test Images**	**Validation Images**
Trimaran	n04483307	810	270	270
English Foxhound	n02089973	482	162	160
Pickelhaube	n03929855	810	270	270
Dishrag, Dishcloth	n03207743	810	270	270
Meerkat, Mierkat	n02138441	810	270	270
Bloodhound, Sleuthhound	n02088466	810	270	270
Common Newt, Triturus vulgaris	n01630670	810	270	270

Table 2: Sample of dataset statistics with real class names, encrypted class names, and data splits for training, testing, and validation.

The images had gone through a series of pre-processing tasks for maximum compatibility with AlexNet, ResNet-18, and VGGNet-19; this included resizing images to 224×224 pixels and normalization technique(min-max normalization) was performed to scale pixel values to the range[0,1] using:

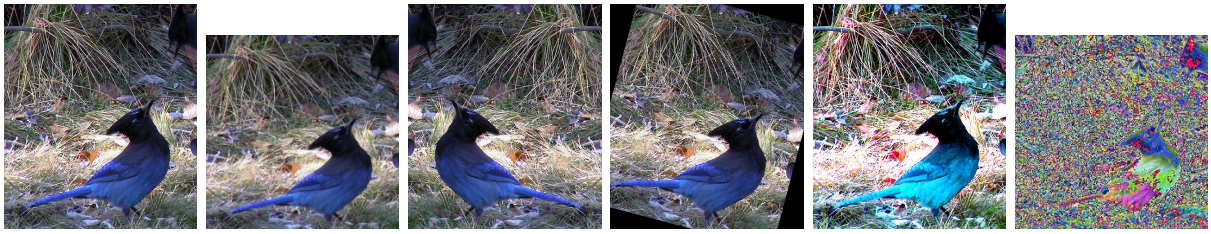
$$\textbf{Min-Max Normalization:} \qquad I' = \frac{I - I_{\min}}{I_{\max} - I_{\min}} \tag{1}$$

This standardization reduces variance and improves gradient descent convergence. Along with that, to create a more diverse dataset and potentially reduce overfitting (Fong and Vedaldi, 2017), multiple data augmentation techniques such as horizontal flip, random rotation, color jitter, and random zoom were applied. **Figure 1** shows all the augmentations each image went through.

During phase three: Two different backpropagation techniques were implemented.

1. **Standard Backpropagation:** gradients of the loss function (L) with respect to all weights (w_i) are computed:
$$\nabla_{w_i} = \frac{\partial L}{\partial w_i}. \tag{2}$$

[1]ImageNet-1k: `https://huggingface.co/datasets/ILSVRC/imagenet-1k`



(a) Original (b) Cropped (c) Flip (d) Rotation (e) Color Jitter (f) Normalized

Figure 1: Visualization of Original and Augmented Images

These gradients are then applied to update all weights during each iteration/epoch:

$$w_i \leftarrow w_i - \eta \nabla_{w_i}. \tag{3}$$

where η is the learning rate.

As this approach updates all the parameters irrespective of their contribution to the loss, it ensures an exhaustive training, therefore, it is computationally expensive.

2. **Selective Backpropagation with He Initialization:**

Weights were initialized using:

$$w_i \sim \mathcal{N}\left(0, \sqrt{\frac{2}{n_{\text{in}}}}\right) \tag{4}$$

where n_{in} is the number of inputs to the layer.

Feature ranking prioritizes weights with high gradient magnitudes for updates, as calculated by:

$$\text{Importance Rank}(w_i) = |\frac{\partial L}{\partial w_i}|. \tag{5}$$

Weights with higher importance ranks are updated, while weights with less influence on the conclusions are skipped. In terms of mathematics only the top $p(t)\%$ of the weights, ranked by the importance are updating during each epoch t.

Two configurations of selective backpropagation were implemented to select top $[p(t)\%]$ of the weights to be updated:

(a) **Linear Decay:** The percentage of weights updated decreases linearly over epochs:

$$p(t) = p_0 - kt, \tag{6}$$

where p_0 is the initial percentage, k is the decay rate, and t is the epoch.

(b) **Exponential Decay:** The percentage of weights updated decreases exponentially:

$$p(t) = p_0 \cdot e^{-kt}. \tag{7}$$

Weights are updated selectively:

$$w_i \leftarrow w_i - \eta \nabla_{w_i}, \quad \text{if } w_i \in \text{Top } p(t)\%. \tag{8}$$

whereas, non-updated weights remain unchanged.

Selective backpropagation reduces the overall computational requirements while retaining important learning updates as it reduces number of computations required per iteration/epoch. Linear decay offers gradual reductions, while exponential decay achieves faster savings in later epochs. While He initialization prevents vanishing or exploding gradients especially in cases of ReLU-based architectures (He et al., 2016) and ensure better convergence. An architectural comparison between standard and selective backpropagation can be seen by an example of ResNet in **Figure 2**.

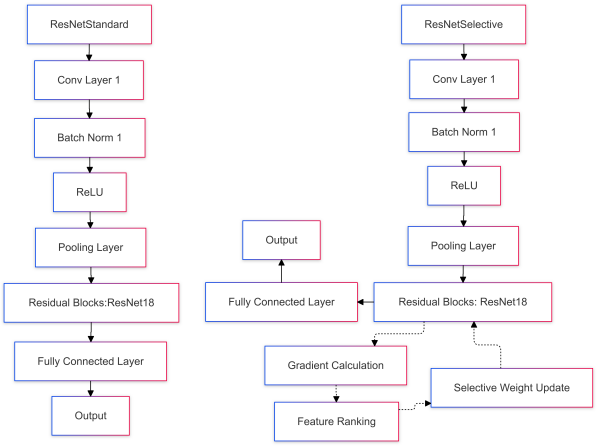


Figure 2: ResNet with Standard and with Selective Backpropagation example

During phase four: Three image classification deep learning models were selected to test these backpropagation approaches:

1. AlexNet: A shallow network with 8-Layers(5 convolutional and 3 fully connected layers) is known for its effectiveness on moderate datasets (Krizhevsky et al., 2012).

2. ResNet-18: A medium-depth network consisting of 18-Layers and with the help of residual blocks it gains the ability to skip connections in-turn addressing vanishing gradients (He et al., 2016).

3. VGGNet19: A deeper network with 19-Layers uses small convolutional filters(generally 3x3) which gives it robust feature extraction capabilities (Simonyan and Zisserman, 2014).

Each model was implemented in three configurations:

1. Standard backpropagation.

2. Selective backpropagation with linear decay.

3. Selective backpropagation with exponential decay.

This selection provides us a broad range of model architectures from small, medium, and large and enables a more robust evaluation of selective backpropagation methods across models of different complexities.

During phase five: A comprehensive comparison of standard backpropagation with selective backpropagation(linear and exponential decay). Evaluation metrics include accuracy, F1-score, training time and energy consumption. To attain a detailed understanding of the trade-offs between computational efficiency and model performance, both overall performance and epoch-wise trends were captured.

- **Performance Metrics:**

  Accuracy was calculated as:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \tag{9}$$

  Precision, **recall**, and **F1-score** were calculated as:

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad R = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad F1 = 2 \times \frac{P \times R}{P + R} \tag{10}$$

  Precision evalutes how many model's positive predictions are correct which makes it critical where false positives are expensive. **Recall** calculates the proportion of true positives correctly identified out of all positives, this makes it important identifying all true positives is critical. **F1-Score** F1-score is the harmonic mean of precision and recall, it's important in cases where both false positives and false negatives are important.

  Confusion Matrix provides a detailed information about model predictions versus the actual labels for each class and helps identify specific classes where the performance of a model is poor.

- **Efficiency Metrics:**
  Training Time was recorded per epoch and for the overall model to complete training. This allowed us to make a comparison of computational efficiency between different backpropagation methods.

  Energy consumption was tracked using 'CodeCarbon', it provides estimates of energy usage (in kilowatt-hours) and CO_2 emissions.

  'Nvidia NVML' was used to monitor and assess how effectively the **GPU resources** are utilized.

  'CPU' and **'RAM'** usage was monitored using 'Psutil' to identify potential bottlenecks.

  Peak memory usage was recorded to ensure models fit within hardware constraints.

4 Design Specification

This section describes the architectural and operational specifications of the proposed framework in contrast with the old standard. We will highlight the similarities and differences of both architectures, that is, standard backpropagation and selective backpropagation(linear and exponential decay) as shown in Table 3.

4.1 Framework Architecture

To enable data preprocessing, model training, feature ranking, selective backpropagation, and evaluation, the proposed framework integrates these multiple models as shown in Figure 3.

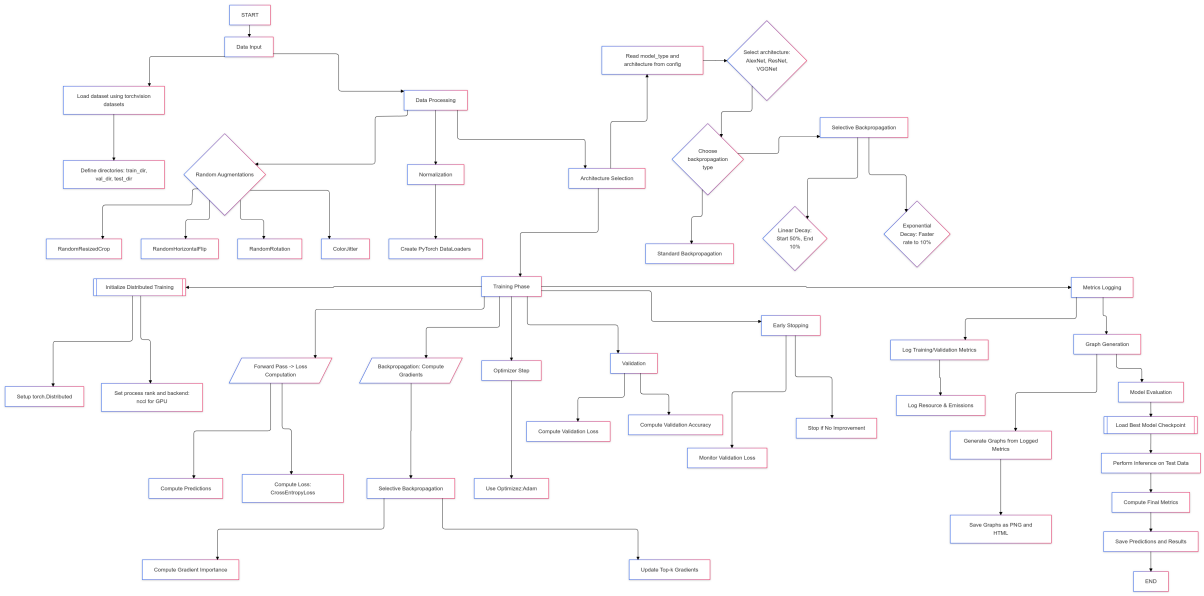


Figure 3: Overall system workflow, showing the interaction between modules from data input to model evaluation.

1. **Data Preprocessing Module:** Data loading, augmentations and pre-processing of the data is handled.
2. **Model Module:** Architectures of all the different neural networks(Alexnet, ResNet, and VGGNet) are defined.
3. **Feature Ranking Module:** Calculations of the feature ranking based on gradients magnitudes during each iteration/epoch are handled.
4. **Selective Backpropagation Module:** Based on the selected decay function(linear or exponential), selective updates of weights mechanism are handled.
5. **Evaluation Module:** Training and testing logs are kept along with that performance and efficiency metrics are calculated.

4.2 Standard Backpropagation Architecture

Traditional neural networks uses standard backpropagation architecture while training deep learning models, in which all the parameters weights are updated during each iteration/epoch. The complete process is as follows:

Workflow:

1. **Input Layer:** Receives pre-processed images.
2. **Forward Propagation:** Data is passed from the input layer in a forward direction, going through hidden layers to the output layer and activations are computed through the network layers to produce predictions.
3. **Loss Computation:** Calculates the difference between predictions and ground truth using the loss function.

4. **Backward Propagation:** Computes gradients for all parameters.

5. **Weight Update:** Updates all weights and biases using the computed gradients in order to minimize the loss.

Standard backpropagation is simple and easy to implement, it provides updates to all the intermediate parameters and mostly leads to thorough learning, but comes with high computational cost and it may include unnecessary updates to weights which has minimal impact on loss reduction.

4.3 Selective Backpropagation Architecture

Our new selective backpropagation architecture overrides the standard backpropagation process and introduces feature ranking mechanism to prioritize and update weights selectively.

Workflow:

1. **Weights Initialization:** Weights are initialized.

2. **Input Layer:** Receives pre-processed images.

3. **Forward Propagation:** Same as standard backpropagation.

4. **Loss Computation:** Same as standard backpropagation.

5. **Backward Propagation:** Computes gradients for all parameters and the Feature Ranking Module ranks the weights based on magnitudes of these gradients.

6. **Selective Weight Update:** Selects top $p(t)\%$ of weights to update based on the selected decay function and updates only the selected weights.

Selective backproagation reduces the overall computational requirements as this approach focuses only on parameters which are significant(parameters which have higher feature ranking) and 'He Initialization' ensures that gradients remain within a manageable range, resulting in faster convergence as it provides optimal inital weights. This process potentially results in faster training times and reduced energy consumption. On the other hand, implementation complexity increases due to introduction of feature ranking and selection mechanisms.

4.4 Training Setup

Training was conducted on an AWS **g4dn.12xlarge** instance with:

- **Hardware:**
    1. 4 NVIDIA Tesla T4 GPUs.
    2. 48 vCPUs.
    3. 192 GB RAM.

Feature	**Standard Backpropagation**	**Selective Backpropagation**
Weight Updates	All weights updated	Only top $p(t)\%$ weights updated
Computational Cost	High	Reduced
Energy Consumption	Higher	Lower
Training Time	Longer	Shorter
Implementation Complexity	Simpler	Slightly more complex
Performance Impact	Baseline	Comparable or potentially improved

Table 3: Comparison of Standard and Selective Backpropagation

- **Software:**
  1. 'PyTorch' for model implementation.
  2. 'Torchvision' for data pre-processing and augmentation.
  3. 'codecarbon' (Anthony et al., 2020) for energy consumption monitoring and estimating CO_2 emissions.
  4. NVIDIA NVML for GPU utilization tracking.
  5. 'psutil' for CPU and RAM usage monitoring.
  6. 'Plotly' for visualizing results.

Models were trained for 1000 epochs using the Adam optimizer (Kingma and Ba, 2014) ($\beta_1 = 0.9, \beta_2 = 0.999$), a learning rate of 0.0001, and a batch size of 32.

The cross-entropy loss function, defined as:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}), \tag{11}$$

was used to optimize classification accuracy. Early stopping was enabled based on validation loss to prevent overfitting.

4.5 Design Justification

1. By updating only the most impactful weights, the framework reduces unnecessary computations, leading to faster training and lower energy consumption.
2. Prioritizing significant weight updates enhances learning by emphasizing critical features.
3. The framework is more scalable to larger models and datasets due to reduced computational requirements.
4. He Initialization helps in supressing vanishing gradients. Deeper models such as VGGNet-19 often face gradient decay, this is where 'He Initialization' helps.
5. Addresses the need for sustainable AI practices by minimizing energy usage and computational resources (Strubell et al., 2019).

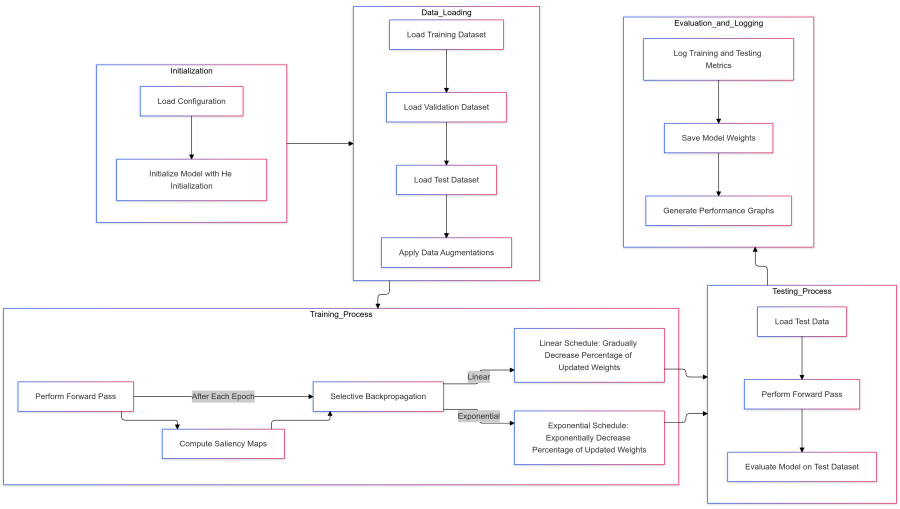


Figure 4: Selective Backpropagation workflow

5 Implementation

This section details the final implementation of three widely-used architectures: AlexNet, ResNet-18, and VGGNet-19 using the proposed selective backpropagation framework.The implementation involves combining feature-ranking based selective weight update technique into these model's backpropagation process while preserving their core architectural integrity. For evaluating the performance of selective approach, the standard backpropagation as a baseline.

5.1 Standard Backpropagation

AlexNet, ResNet-18, and VGGNet-19 were implemented and trained using the traditional procedure to establish a baseline. PyTorch's default techniques were used in order to initialize the weights which vary based on the layer type.

All the weights were updated uniformly at each epoch using backpropagation during the training procedure. Prediction errors were quantified using cross-entropy loss function and the weights were adjusted based on their gradients across all layers using Adam optimizer. No feature-ranking or masking of weight updates was applied.

The training process produced fully trained baseline models with standard backpropagation , serving as a reference for comparing accuracy, computational efficiency, and sustainability metrics against selective backpropagation.

5.2 Selective Backpropagation

The selective backpropagation framework introduced a feature-ranking based weight updates mechanism during training. The focus of this approach was to reduce training costs and energy consumption by dynamically selecting a subset of critical weights for optimization during each epoch while maintaining performance.

He initialization(also known as Kaiming Normal Initialization) was used for initializing weights of all the models in the selective framework to ensure stable variance propagation through layers with ReLU activations (He et al., 2016). The issues such as vanishing gradients or exploding gradients were diminished through this initialization, particularly

in deep architectures like VGGNet-19. The workflow of selective backpropagation is illustrated in **Figure 4**

The process of performing forward pass in order to compute predictions remained same as that of standard process. During backpropagation, gradients of the loss function with respect to the model weights were computed which were then used to generate saliency maps, which quantified the importance of each weight:

$$S_{ij} = \left| \frac{\partial L}{\partial W_{ij}} \right| \tag{12}$$

where S_{ij} denotes the saliency score of weight W_{ij}, and L represents the loss function. These saliency scores highlighted the weights that contributed significantly to reduce the loss during that epoch.

The weights were ranked from most to least important parameters during epoch, producing an ordered list based on these saliency scores. It provided the foundation for prioritizing weight updates during selective backpropagation.

A binary mask was applied to gradients by the selective backpropagation process, allowing only a subset of weights to be updated. The percentage of weights selected varied dynamically according to the chosen update schedule:

1. Linear Schedule: The percentage of updated weights decreased linearly over epochs, starting at 50% in the initial epoch and reducing to a predefined minimum percentage by the final epoch based on the total number of epochs the model is training for. By introducing aggressive selective updates as training went on, it allowed for a balanced accuracy and efficiency.

2. Exponential Schedule: The percentage of weights updated decreased exponentially over epochs based on the total number of epochs the model is training for, enabling more aggressive reduction of weight updates in later epochs. Although it prioritized computational efficiency but careful tuning was required to avoid performance degradation.

The gradients of masked weights were zeroed out, effectively freezing their updates. Only the unmasked weights were optimized during backpropagation using the Adam optimizer.

This process was implemented into AlexNet, ResNet-18, and VGGNet-19 model architectures. Where each model underwent two training runs, one with a linear schedule and another with an exponential schedule. These configurations allowed for an in-depth evaluation of the trade-offs between accuracy, training efficiency, and energy consumption.

5.3 Tools and Technologies

`Python` was used as the primary programming languages in harmony with `PyTorch` for implementing both standard and selective backpropagation frameworks.

PyTorch's `Torchvision` was used for performing data transformations along with plotly for visualizations.

`psutil` and `pyNVML` were used to monitor CPU and GPU resource usage respectively during training.

`codecarbon` was used to track energy consumption and estimated CO_2 emissions during training.

6 Evaluation

An in-depth evaluation of experimental results is discussed in this section. Comparison between Selective Backpropagation strategies with (Linear and Exponential decay schedules) with their respective Standard Backpropagation strategy(Baseline Model) across three models: AlexNet, ResNet, and VGGNet is carried out.

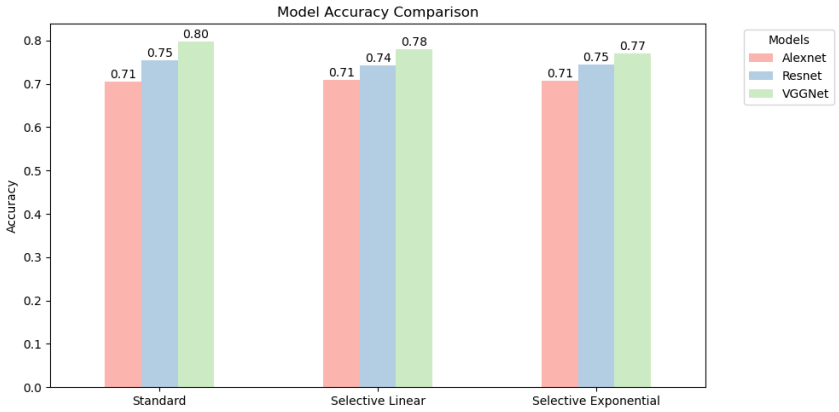


Figure 5: Test Accuracy of all models

6.1 Experiment 1: AlexNet

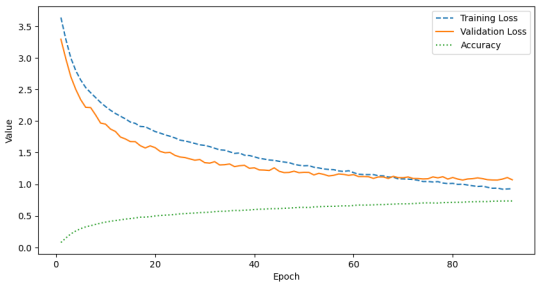
Strategy	**Test Accuracy (%)**	**F1 Score (%)**	**ROC AUC**	**Training Time (hh:mm:ss)**	**Energy Consumption (kWh)**	CO_2 **Emissions (kg)**
Standard	70.60	70.45	0.9818	2:01:39	0.67	0.25
Selective Linear	70.84	70.75	0.9821	1:42:46	0.62	0.23
Selective Exponential	70.63	70.47	0.9816	1:42:31	0.62	0.23
Standard: 1000 classes	49.25	48.54	0.9887	21:17:18	8.054	2.97
Selective Exponential: 1000 classes	47.40	46.51	0.9870	15:24:34	6.22	2.29

Table 4: AlexNet metrics comparison

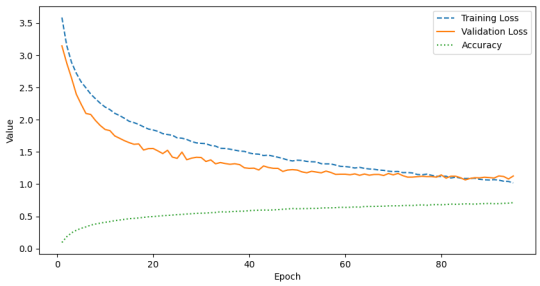
The aim of this experiment is to evaluate the effectiveness of feature-ranked selective backpropagation in improving training efficiency and reducing energy consumption on a foundational architecture like AlexNet.

As shown in **Table 4** in terms of test performance both the selective decay schedules demonstrated strong alignment with the standard approach while achieving noteworthy gains in training efficiency, **Figure 6** shows the comparison of training metrics of all backpropagation techniques used. Compared to 70.60% test accuracy of standard backpropagation, selective backpropagation with linear decay schedule outperformed with

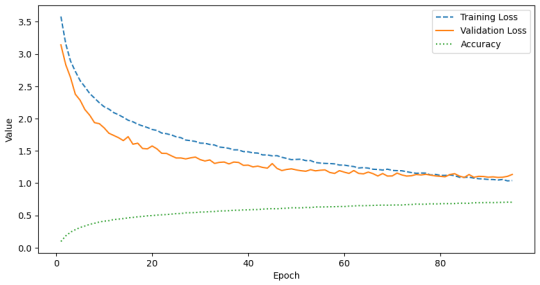
70.84% as shown in **Figure 5**. Similarly, the exponential schedule maintained comparable performance, with a marginal accuracy reduction to 70.63%.



(a) AlexNet Standard



(b) AlexNet Selective Linear



(c) AlexNet Selective Exponential

Figure 6: AlexNet: Training metics

Both selective schedules achieved substantial reductions in training time and energy consumption. A 15.5% improvement in training time in linear schedule with reduced training time of 1:42:56 over the standard approach with training time of 2:01:39 as shown in **Figure 8**. The exponential schedule demonstrated similar efficiency, achieving a 15.6% reduction in training time. Energy consumption for both selective strategies decreased from 0.67 kWh to 0.62 kWh, with corresponding reductions in CO_2 emissions from 0.25 kg to 0.23 kg as shown in **Figure 7**.

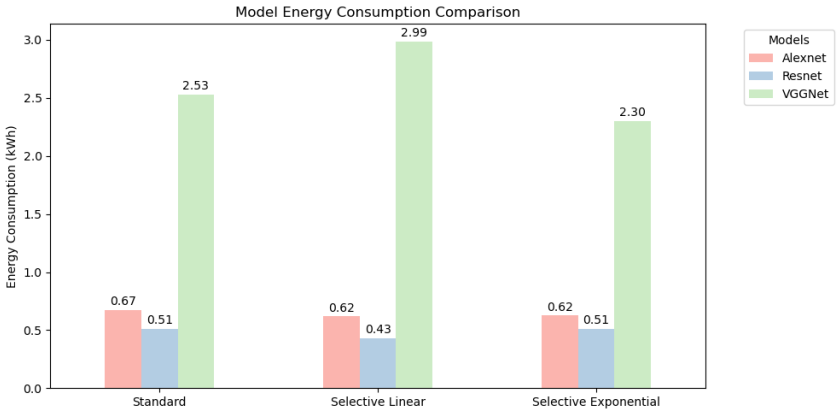


Figure 7: Average energy consumption(KWh) of all models

Despite having aggressive weights reductions in exponential schedule, accuracy degradation was minimal. The linear schedule model was shown to have stability under selective updates with marginally better performance in terms of F1 Score (70.75% vs. 70.45%) and ROC AUC (0.9821 vs 0.9818).

Alexnet tested with complete ImageNet dataset with 1000 classes showed results consistent with the ones tested on the subset with only about 2% drop in accuracy while reducing training time by ∼6 hours.

These results highlight AlexNet's ability to handle selective weights updation based on feature importances which makes it an efficient candidate for feature importance based selective backpropagation in resource-constrained environments due to its shallow architecture.

Figure 8: Training time(seconds) of all models

AlexNet's shallow architecture limits scalability assessments, motivating a transition to ResNet-18, which introduces residual connections to handle deeper networks effectively.

6.2 Experiment 2: ResNet

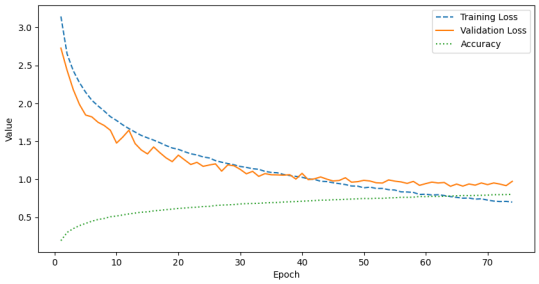
Strategy	**Test Accuracy (%)**	**F1 Score (%)**	**ROC AUC**	**Training Time (hh:mm:ss)**	**Energy Consumption (kWh)**	CO_2 **Emissions (kg)**
Standard	75.45	75.39	0.9883	1:20:48	0.51	0.19
Selective Linear	74.22	74.40	0.9868	1:08:33	0.43	0.16
Selective Exponential	74.54	74.61	0.9865	1:20:03	0.51	0.19

Table 5: ResNet metrics comparison

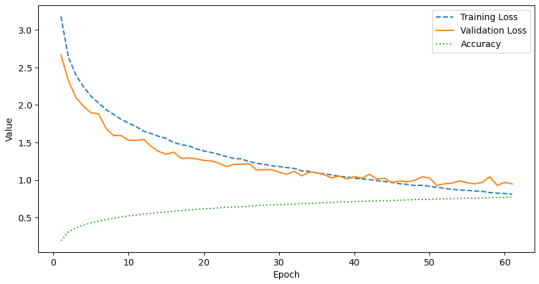
The aim of this experiment is to assess the scalability of the proposed selective backpropagation framework in deeper architectures like ResNet-18.

As shown in **Table 5**, the selective strategies revealed nuanced trade-offs between accuracy and efficiency which is depicted by the training metrics in **Figure 9**. A modest decrease in test accuracy from 75.45% with standard backpropagation to 74.22% with linear schedule was seen. Relatively higher accuracy of 74.54% was seen with exponential schedule, reflecting its capacity to retain critical weight updates effectively. **Figure 9** shows the complete training metrics comparison.

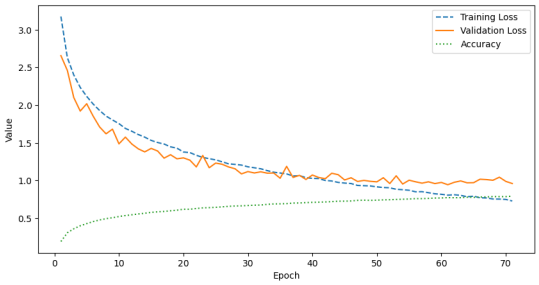
Efficiency gains were more noticeable in the linear schedule which reduced training time by 15.2% (from 1:20:48 to 1:08:33) and energy consumption by 15.7% (from 0.51



(a) ResNet Standard



(b) ResNet Selective Linear



(c) ResNet Selective Exponential

Figure 9: ResNet: Training metics

kWh to 0.43 kWh), a more detailed comparison of training times can be seen in **Figure 8**. Whereas, the exponential schedule, achieved a smaller reduction in training time (0.9%), but, energy consumption levels remain similar to that of standard approach.

The linear schedule indicated a significant reduction in CO_2 emissions, decreasing from 0.19 kg to 0.16 kg. In contrast, as seen in the **Figure 7** the exponential schedule did not improve sustainability metrics significantly, retaining emissions at 0.19 kg.

ResNet's deeper architecture made it more sensitive to selective schedules, with saliency-based updates effectively preserving key feature representations. However, the minor accuracy reductions and the trade-offs in sustainability metrics underscore the need for adaptive scheduling mechanisms tailored to such architectures.

ResNet-18's moderate depth provides scalability insights of the framework, but transitioning to the deeper and more uniform VGGNet-19 allows evaluation in highly complex architectures.

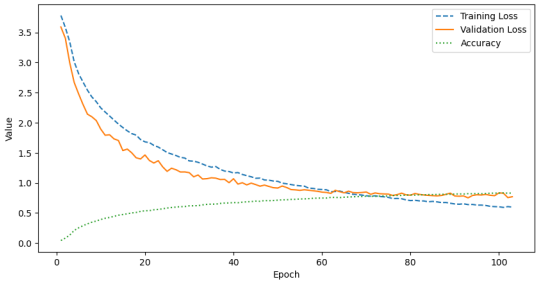
6.3 Experiment 3: VGGNet

Strategy	**Test Accuracy (%)**	**F1 Score (%)**	**ROC AUC**	**Training Time (hh:mm:ss)**	**Energy Consumption (kWh)**	CO_2 **Emissions (kg)**
Standard	79.84	79.75	0.9912	5:51:45	2.53	0.93
Selective Linear	78.03	78.24	0.9891	6:53:59	2.99	1.10
Selective Exponential	76.92	76.89	0.9887	5:18:38	2.30	0.85

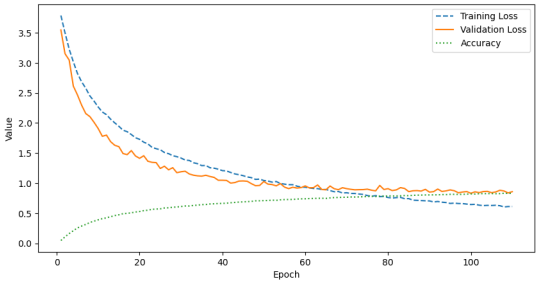
Table 6: VGGNet metrics comparison

The aim of this experiment is to examine the applicability of selective backpropagation in a highly complex and deep architecture like VGGNet-19.

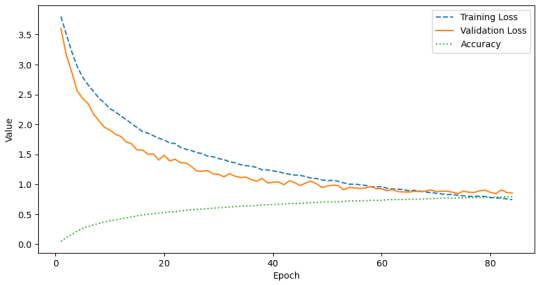
As shown in **Table 6**, greater variation in performance and efficiency was seen in selective schedules compared to standard approach. The linear schedule achieved a test accuracy of 78.03%, a 1.81% reduction from the 79.84% accuracy of the standard approach. Whereas, exponential schedule endured a more drastic accuracy drop to 76.92%, reflecting to challenges of aggressive weight reduction in deeper architectures, which can be found in **Figure 5**. Also, complete training metrics can be seen in **Figure 10** for all techniques.



(a) VGGNet Standard



(b) VGGNet Selective Linear



(c) VGGNet Selective Exponential

Figure 10: VGGNet: Training metics

Efficiency gains were mixed. The exponential schedule reduced training time by approximately 9% (from 5:51:45 to 5:18:38), depicted in **Figure 8** and energy consumption by 9.1% (from 2.53 kWh to 2.30 kWh) seen in **Figure 7**.

In contrast, the linear schedule increased the training time and energy consumption by 17.3% and 18.2% respectively as seen in **Figure 8** and **Figure 7**, resulting in higher CO_2 emissions (1.10 kg compared to 0.93 kg with the standard approach).

These results suggest that the exponential schedule is better suited for computationally expensive models like VGGNet, where reducing training time and energy consumption outweighs the minor dip in accuracy. Inefficiencies of the linear schedule highlight the limitations of fixed weight reduction strategies for deeper networks, emphasizing the need for adaptive or depth-aware approaches.

6.4 Discussion

This section evaluates the results of Feature-Ranking Based Backpropagation(Linear and Exponential Decay Schedules) in contrast to Standard Backpropagation, focusing on their strengths, drawbacks, potential and challenges of integrating it in real-world deep learning applications.

Both linear and exponential schedules demonstrates the applicability of selective backpropagation across diverse architectures as they gave importance to critical weights using feature ranking. A minimal accuracy degradation was seen in shallow architecture like AlexNet which exhibits its robustness to selective updates. While in deeper architectures, a greater sensitivity was to be seen, particularly under exponential schedule.

A balanced performance and energy savings were depicted by AlexNet and ResNet under linear schedule. While reducing training time and energy consumption these architectures achieved accuracy retention close to the standard approach. For deeper models like VGGNet, exponential schedule was successful in reducing training time but at the cost of reduced accuracy. The dependence of deeper architectures on iterative refinement in later layers is the likely cause of these challenges.

The potential of selective backpropagation to achieve comparable performance with reduced computational costs makes it a viable option for deploying AI models not only in resource-constrained environments, such as edge devices or low-power systems but also in larger architectures.These results roots towards developing more sustainable AI practices to reduce the environmental impact of AI is validated by the reduction in CO_2 emissions observed across all models. These results aligns with (Henderson et al., 2020), where they emphasize on developing energy-efficient machine learning.

A flexible framework and adaptable to various model depths and complexities. This makes the selective backpropagation suitable for both shallow and deep networks. Feature-Ranking based selective weight updates retained critical parameters which enabled the models to maintain high classification performance even with selective weight updates. This demonstrates the robustness of gradient-based ranking methods (Simonyan and Zisserman, 2014).

Feature-Ranking based selective weight updates reduced the overall training costs, an additional computational overhead was introduced due to computation of saliency maps which was more evident for deeper architectures. In future, more efficient ranking mechanisms, such as attention-based approaches can be explored to reduce computational complexity.

One-size-fits-all strategy is suboptimal for selective weight updates strategy, this is evident with the varying impacts of selective backpropagation on AlexNet, ResNet, and VGGNet. For a broader applicability of the framework, tailored approaches that account for architectural depth and feature hierarchies are necessary.

Extend the study to larger datasets, such as the full ImageNet dataset or other benchmarks like `CIFAR-100` or `COCO`, to validate the scalability and robustness of the approach.

This study extends the previous work where effectiveness of saliency-based optimization for identifying and prioritizing critical weights was highlighted by (Simonyan and Zisserman, 2014) and (He et al., 2016).

This is done by systematically evaluating the impact of feature ranking based selective weight updation across architectures and schedules, introducing reduced training times, reduced computational requirements as well as sustainability metrics as a novel dimension.

7 Conclusion and Future Work

The aim of this research was to reduce computational costs, energy consumption and carbon footprint of AI models by optimizing the training process of deep learning models for image classification by integrating feature-ranking based selective backpropagation.

This research proposed a novel framework that dynamically prioritizes important weights for update using saliency maps, applied to well-established architectures such as AlexNet, ResNet-18, and VGGNet-19. The objectives included investigating the state-of-the-art models, designing and implementing our own framework, and evaluating impact of the framework on training efficiency and model performance against state-of-the-art. Additionally, this framework was tested on the entire ImageNet-1K dataset for AlexNet, confirming consistent performance gains, although similar testing was not conducted for ResNet-18 and VGGNet-19 due to resource constraints.

Results showed that the proposed method was able to achieve substantial reductions in training time by 15% and energy consumption while maintaining on-par performance against state-of-the-art, with AlexNet showing consistent improvements on both the subset and full ImageNet-1K dataset.. The energy usage and CO_2 emissions were seen to be reduced by up to 20% when compared to standard approaches. The dynamic feature importance weighting of the framework proved to be effective across all three architectures and ten models, offering a scalable and sustainable solution for deep learning optimization.

From both academic and professional point of view, the impacts of this research are significant. By addressing the inefficiencies of standard backpropagation, it contributes to the broader field of sustainable AI paving a way forward for developing more energy-efficient deep learning training methods. However, currently, this work has limitations, while AlexNet was tested on the full ImageNet-1K dataset; its applicability remained untested for other larger datasets, models and other domains.. Additionally, certain details of feature importance in more complex architectures may not be captured by gradient-based saliency maps.

Extending the framework to different datasets and exploring its applications for other deep learning tasks such as object detection or natural language processing could be done in future. To enhance the efficacy of selective backpropagation more can include exploring the integration of advanced interpretability methods, such as attention mechanisms or hybrid feature ranking techniques. This framework can enable widespread adoption of sustainable AI practices as it could be refined to be deployed in energy-constrained environments, applications are numerous such as edge devices or mobile platforms.

This research demonstrates the potential of sustainable and scalable AI solutions while maintaining the performance. Further advancements in this area contributes to global efforts towards sustainability.

References

Anthony, L., Kanding, B. and Selvan, R. (2020). Carbontracker: Tracking and predicting the carbon footprint of training deep learning models.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database, *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255.

Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation, *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3449–3457.

Goodfellow, I. J., Bengio, Y. and Courville, A. (2016). *Deep Learning*, MIT Press, Cambridge, MA, USA. `http://www.deeplearningbook.org`.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Henderson, P., Hu, J., Romoff, J., Brunskill, E., Jurafsky, D. and Pineau, J. (2020). Towards the systematic reporting of the energy and carbon footprints of machine learning, *J. Mach. Learn. Res.* **21**(1).

Jiang, A. H., Wong, D. L. K., Zhou, G., Andersen, D. G., Dean, J., Ganger, G. R., Joshi, G., Kaminksy, M., Kozuch, M., Lipton, Z. C. and Pillai, P. (2019). Accelerating deep learning by focusing on the biggest losers.
URL: *https://arxiv.org/abs/1910.00762*

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization, *CoRR* **abs/1412.6980**.
URL: *https://api.semanticscholar.org/CorpusID:6628106*

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks, *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, Curran Associates Inc., Red Hook, NY, USA, p. 1097–1105.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. and Fei-Fei, L. (2014). Imagenet large scale visual recognition challenge, *International Journal of Computer Vision* **115**.

Schwartz, R., Dodge, J., Smith, N. A. and Etzioni, O. (2020). Green ai, *Commun. ACM* **63**(12): 54–63.
URL: *https://doi.org/10.1145/3381831*

Shrikumar, A., Greenside, P. and Kundaje, A. (2017). Learning important features through propagating activation differences, *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, JMLR.org, p. 3145–3153.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition, *CoRR* **abs/1409.1556**.
URL: *https://api.semanticscholar.org/CorpusID:14124313*

Strubell, E., Ganesh, A. and McCallum, A. (2019). Energy and policy considerations for deep learning in NLP, *in* A. Korhonen, D. Traum and L. Màrquez (eds), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Florence, Italy, pp. 3645–3650.
URL: *https://aclanthology.org/P19-1355*

Vakil-Baghmisheh, M.-T. and Pavešić, N. (2004). Training rbf networks with selective backpropagation, *Neurocomput.* **62**(C): 39–64.
URL: *https://doi.org/10.1016/j.neucom.2003.11.011*