Exploring the Role of Optimizers and Architectures in Active Learning

Khadiza Sarwar Moury, Md Hasibul Hasan Shovo, Syed Abdullah Ali, Youssef Midra
Concordia University, Canada
Student IDs: 40343620, 40329539, 40333397, 40341729
khadizasarwar.moury, mdhasibulhasan.shovo,
syedabdullah.ali, youssef.midra@mail.concordia.ca
Code link: Github link

Abstract

Active Learning (AL) is a strategy for reducing the high cost of data annotation in deep learning. It is achieved by selecting the most informative samples for training using query strategies. This is a developing field where the interplay between the underlying model architecture, the optimization algorithm, and the active learning process remains underexplored. This paper presents a comprehensive comparative study of three ResNet (Residual Network) architectures (ResNet18, ResNet34, ResNet50) trained using four different optimizers (SGD, Adam, AdamW, RMSProp) within a Loss Prediction based active learning framework. Using the CIFAR-10 dataset and a labeling budget of 10,000 samples, we demonstrate that the choice of optimizer significantly dictates the learning trajectory. Our results reveal a distinct trade-off: adaptive optimizers like Adam provide superior sample efficiency in the "cold start" phase (low-data period), whereas SGD with momentum consistently achieves higher generalization accuracy once the labeled pool exceeds a critical threshold. Furthermore, we find that deeper architectures like ResNet-50 require a longer "warm-up" period but ultimately provide the highest performance when paired with SGD. These findings suggest that dynamic optimization strategies play an important role in maximizing the efficiency of active learning pipelines.

1 INTRODUCTION

Deep learning models have achieved remarkable success in image classification tasks, but this success often comes at the cost of requiring large amounts of labeled data. In many domains, labeling such data can be expensive, time consuming, and some domains; such as medical imaging or satellite imagery; require expert annotators. Active Learning (AL) offers a compelling solution to this problem by selecting the most informative or uncertain samples from a large pool of unlabeled data for annotation. Instead of randomly choosing data points, AL prioritizes instances that are expected to improve model performance the most when labeled. A key AL strategy is uncertainty sampling [1], where the model queries data points it is least confident about. This approach helps the model refine its decision boundaries, especially during the early stages of training.

In this research, we explore how different choices of optimizers (SGD, Adam, AdamW, RMSProp) and model architectures (ResNet18, ResNet34, ResNet50) [2] affect the performance and efficiency of an active learning pipeline. Using the CIFAR-10 dataset [3], we simulated a budget constrained labeling scenario by incrementally increasing the labeled dataset over 10 active learning cycles. Through systematic experimentation, we discovered how these fundamental design decisions influence model accuracy, sample efficiency and cost; key considerations for deploying image classification systems in practical environments with limited resources.

2 RESEARCH OBJECTIVES

- To assess the effectiveness of various optimizers (SGD, Adam, AdamW, RMSProp) in the context of active learning.
- To evaluate how different Residual Network (ResNet) architectures (ResNet18, ResNet34, ResNet50) impact the performance of active learning on the CIFAR-10 dataset.
- To compare the overall performance (accuracy, sample efficiency, and computational cost) of these models and optimizers under a fixed labeling budget.

3 METHODOLOGY

This research adopts an iterative active learning framework to evaluate how differing model architectures and optimizers impact learning efficiency and performance. The study uses the benchmark image classification dataset CIFAR-10 [3].

The active learning process is initialized without labeled data. In the first cycle, a "cold start" strategy is employed where 1,000 samples are selected and labeled randomly. In each subsequent cycle, the model queries an additional 1,000 samples from the unlabeled pool using the Loss Prediction strategy, adding them to the labeled training set. This iterative process continues over 10 cycles, resulting in a total labeling budget of 10,000 samples per run. To ensure consistent evaluation, the model is fully retrained using the updated labeled set after each cycle. Performance is assessed via accuracy on a held-out test set, sample efficiency, and computational efficiency. This methodology is applied systematically across all model–optimizer combinations outlined below.

3.1 Model Architectures

We utilized three variations of the ResNet family to analyze the trade-off between model complexity and active learning efficiency. These networks utilize residual learning through identity based skip connections, which mitigate the vanishing gradient problem and enable the stable optimization of deeper models without degradation in accuracy [2]. The architectural differences are visualized in Fig. 1.

- **ResNet18:** A light, yet powerful baseline. Its shallow depth, combined with the residual block structure, offers fast training and inference while maintaining strong generalization performance. This makes it well suited for resource constrained environments or for rapid prototyping when exploring data characteristics and hyperparameter configurations. Moreover, its reduced parameter count helps minimize overfitting when working with smaller datasets or when extensive data augmentation is required to enhance model robustness.
- **ResNet34:** ResNet-34 builds on ResNet-18 by increasing the network depth and representational capacity while maintaining the same basic residual block architecture. This architecture is particularly advantageous when the dataset is more complex and requires detailed feature extraction without a substantial increase in computational cost. The additional layers enable the model to learn more complicated features and patterns, improving task performance. Its moderate complexity provides a balanced trade-off between efficiency and accuracy.
- **ResNet50:** ResNet-50 introduces *bottleneck residual blocks*, which use a 1×1–3×3–1×1 convolutional pattern to greatly increase depth compared to ResNet-34 while controlling parameter growth. This structure yields a significantly more detailed model capable of capturing intricate patterns and high-level structures. This architecture is ideal for complex datasets or tasks that benefit from deep feature hierarchies. Despite its depth, the optimized bottleneck design ensures efficient gradient flow and manageable memory usage.

3.2 Optimizers

To evaluate the impact of optimization dynamics on active learning performance, we trained the architectures using four distinct optimizers. Each optimizer employs a different strategy for navigating the loss landscape, influencing convergence speed and generalization capability.

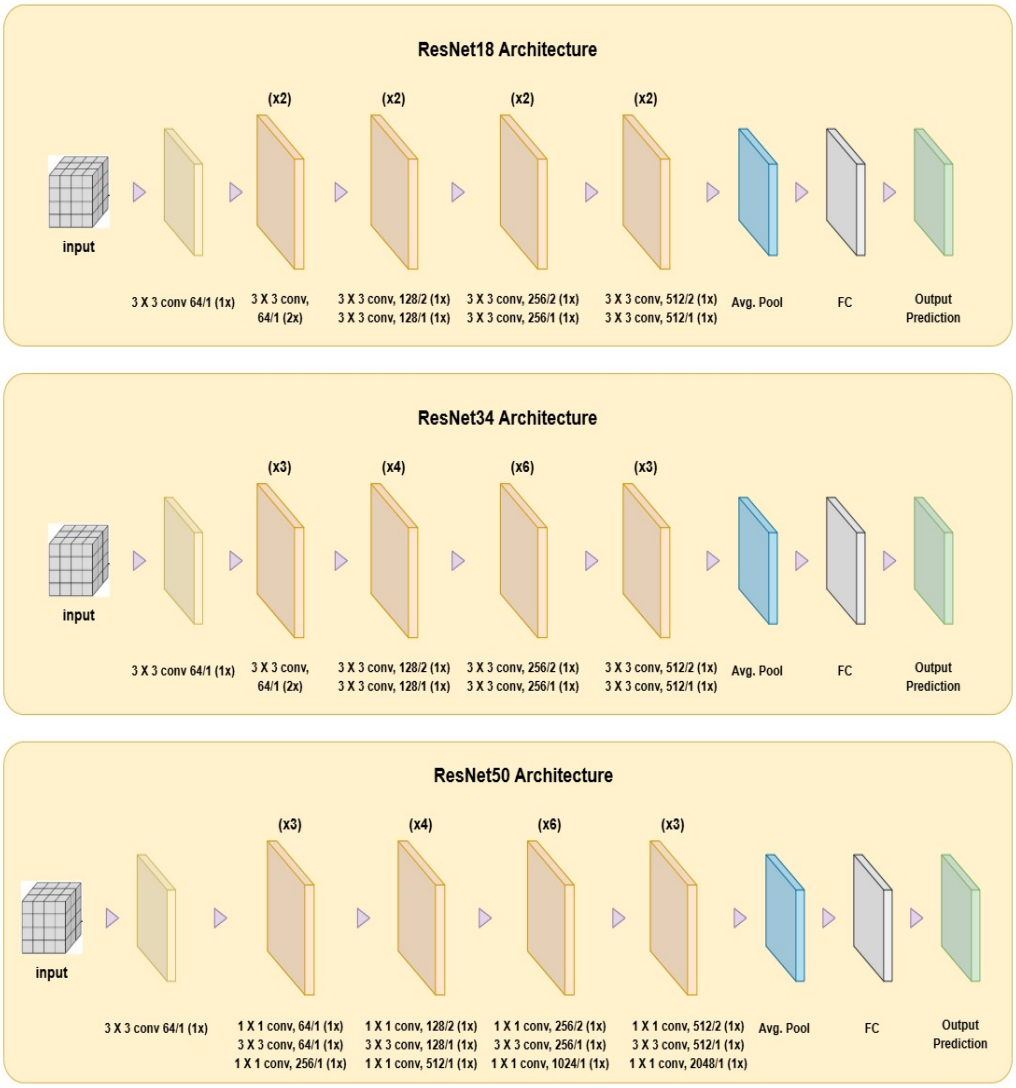


Figure 1: Overview of the ResNet architectures (ResNet18, ResNet34, and ResNet50) utilized in this study, highlighting the difference in depth and convolutional blocks.

- **Stochastic Gradient Descent (SGD) [5]:** We utilize SGD with momentum, a classical optimization algorithm that updates parameters based on the gradient of the loss function with respect to a subset of the data. While often slower to converge than adaptive methods, SGD is renowned for finding flatter minima in the loss landscape, which frequently correlates with better generalization on unseen test data; a critical factor in active learning where labeled data is scarce.
- **RMSProp [6]:** The Root Mean Square Propagation (RMSProp) optimizer addresses the diminishing learning rates observed in earlier adaptive methods (like Adagrad). By maintaining a moving average of the squared gradients, RMSProp adapts the learning rate for each parameter individually. This allows for faster convergence in non-convex settings and effective handling of non-stationary objectives, making it robust against the noise introduced by the iterative data selection process.
- **Adam [7]:** Adaptive Moment Estimation (Adam) combines the benefits of RMSProp and momentum. It computes individual adaptive learning rates for different parameters from estimates of the first and second moments of the gradients. Adam is widely adopted for its

rapid convergence speed and low memory requirements. However, in the context of active learning, its aggressive optimization can sometimes lead to overfitting on smaller, initial labeled sets.

- **AdamW [8]:** AdamW is a modification of the Adam optimizer that decouples weight decay from the gradient update. In standard Adam, L_2 regularization is not equivalent to weight decay, which can lead to suboptimal generalization. AdamW corrects this by applying weight decay directly to the parameters, yielding better training stability and improved generalization performance, particularly for deep residual networks trained on image classification tasks.

4 EXPERIMENT SETUP

4.1 Implementation and Computing Environment

Our experiments were conducted on Concordia University's High-Performance Computing (HPC) facility, *Speed*. We utilized NVIDIA GPUs (V100, 32GB RAM) available within the cluster nodes for computational efficiency. The active learning pipelines were implemented using the PyTorch framework.

4.2 Dataset and Preprocessing

We utilize the CIFAR-10 dataset, consisting of 60,000 32×32 color images across 10 classes. The dataset is split into 50,000 training images and 10,000 test images.

- **Data Augmentation:** To prevent overfitting, especially in the early active learning cycles where labeled data is minimal, we apply standard augmentations during training.
- **Normalization:** Input images are normalized using the channel-wise means and standard deviations calculated from the training set.

4.3 Model Architectures

We evaluate three variations of the ResNet family to analyze the trade-off between model capacity and active learning efficiency.

- **ResNet18:** The shallowest network with **11M parameters**, serving as a baseline for lower computational cost.
- **ResNet34:** A deeper variant with **21M parameters** providing a middle ground in terms of complexity.
- **ResNet50:** A deep architecture with **25M parameters** utilizing "bottleneck" blocks, testing whether higher capacity models benefit more from uncertainty sampling despite the limited data budget.

All models are initialized with random weights at the start of the experiment to simulate a scenario where no pre-trained weights are available for the specific domain.

4.4 Optimization Strategies

We compare four separate optimization algorithms to determine their stability and convergence speed within the AL loop. The hyperparameters for each were selected based on standard literature benchmarks for CIFAR-10:

1. **Stochastic Gradient Descent (SGD):** Used with a Nesterov momentum of 0.9 and a weight decay of $5e^{-4}$. This serves as the standard baseline for computer vision tasks [5].
2. **Adam:** An adaptive moment estimation method. We utilize default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) [7].
3. **AdamW:** A modification of Adam that decouples weight decay from the gradient update. This is crucial for regularization in adaptive methods [8].

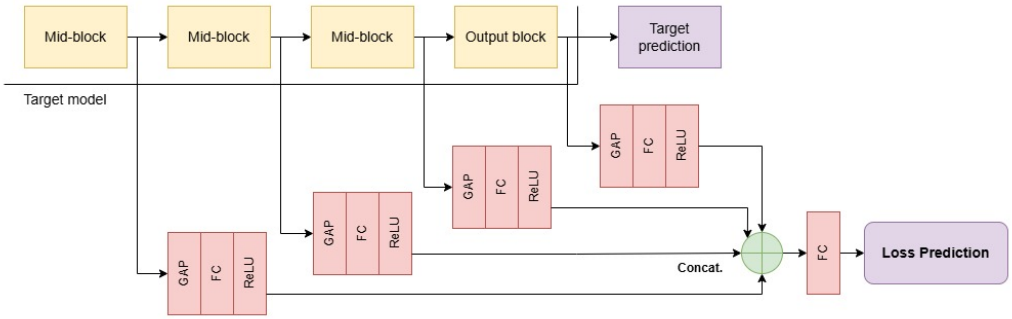


Figure 2: The architectural flow of the Active Learning module. The module extracts features from mid level blocks of the target ResNet, aggregates them, and predicts the loss value to guide sample selection [4].

4. **RMSProp:** Utilizes a moving average of squared gradients to normalize the gradient. We set the smoothing constant $\alpha = 0.99$ [6].

4.5 Active Learning Configuration

The experiment simulates a pool-based active learning scenario. The interaction between the target model and the loss prediction mechanism is detailed in Fig. 2.

- **Query Strategy (Loss Prediction Module):** Instead of relying on fixed heuristic metrics like entropy, we employ a learnable *Loss Prediction Module* [4]. As illustrated in Fig. 2, this module is attached to the intermediate layers of the target ResNet model. It aggregates feature maps from multiple depths (mid blocks) using Global Average Pooling (GAP) and Fully Connected (FC) layers. These features are concatenated and passed through a final FC layer to predict a scalar loss value for a given input. During the active learning acquisition step, the model queries samples from the unlabeled pool that yield the highest predicted loss, under the assumption that high loss samples are the most informative for the current model state.
- **Budget Protocol:** The total budget is 10,000 samples.
  - *Cold Start:* As the model has no initial knowledge, the first cycle ($n = 1$) selects 1,000 samples randomly to establish an initial decision boundary.
  - *Active Cycles:* For cycles $n = 2$ to 10, the model queries the top-1,000 most informative samples (highest predicted loss) from the remaining unlabeled pool.
- **Training Routine:** After each acquisition step, the model is retrained from scratch (re-initialized) on the accumulated labeled set to prevent catastrophic forgetting and to isolate the effect of the added data. After 120 epochs, we stop the gradient from the loss prediction module propagated to the target model.

5 PERFORMANCE EVALUATION

The goal of this study is to analyze different model-optimizer pairings. To achieve this we compare the performance of each model–optimizer combination on the CIFAR-10 dataset. The evaluation in our project naturally extends standard classification evaluation with budget aware metrics that are common in active learning. The following metrics were evaluated after every cycle:

1. **Accuracy (per active learning cycle)**: Tracks model's ability to classify correctly across cycles. After each cycle, we evaluate accuracy on the CIFAR-10 test set:

$$\text{Accuracy}^{(t)} = \frac{1}{n_{\text{test}}} \sum_{j=1}^{n_{\text{test}}} \mathbf{1}\left[f_\theta^{(t)}\left(x_j^{(\text{test})}\right) = y_j^{(\text{test})} \right].$$

2. **Sample Efficiency**: Measures how quickly a model reaches a given accuracy relative to the labeled data. Conceptually, we visualize:

$$\text{Accuracy}^{(t)} \text{ vs. } N_{\text{labels}}^{(t)}.$$

3. **Accuracy vs. Budget**: Treating the labeling budget B as the x-axis, we report:

$$\text{Accuracy}(B), \qquad B \in \{B_1, \ldots, B_{10}\}.$$

4. **Label Efficiency**: Measures accuracy gain per unit of labeled data:

$$\text{Label Efficiency}^{(t)} = \frac{\text{Accuracy}^{(t)} - \text{Accuracy}^{(0)}}{B^{(t)} - B^{(0)}}.$$

6 RESULTS

This section presents the comparative analysis of ResNet architectures and optimizers across the ten active learning cycles. We examine the impact of optimization strategies on model convergence as the labeled dataset size grows from 1,000 to 10,000.

6.1 ResNet-18 Performance

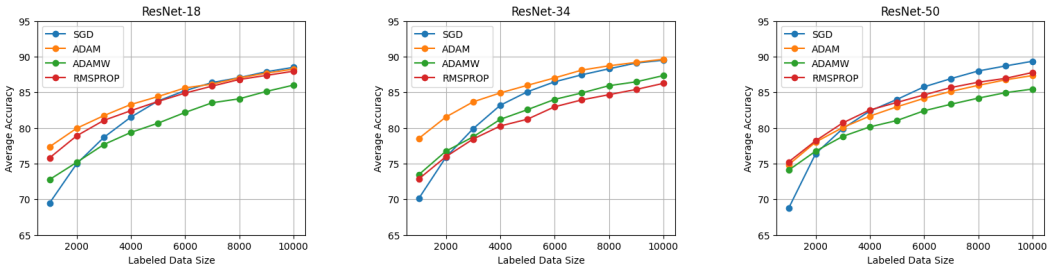
For the ResNet-18 architecture, the results (Fig. 3a) indicate a clear distinction in the early stages of learning. Adaptive optimizers, particularly Adam, demonstrate superior performance during the initial "cold start" phases (1,000 to 3,000 samples), achieving approximately 77% accuracy compared to SGD's 69%.

However, as the labeled data budget increases, SGD closes this gap significantly. By the final cycle (10,000 samples), all optimizers converge towards a similar range (86% - 88%), with SGD notably outperforming the adaptive methods in the final cycles. This reinforces the hypothesis that while adaptive methods learn faster with sparse data, SGD with momentum generalizes better once sufficient data is available.

6.2 ResNet-34 Performance

The trends observed in ResNet-34 (Fig. 3b) mirror those of ResNet-18 but with higher overall accuracy ceilings due to the increased model depth. Adam and AdamW maintain a lead in sample efficiency during the first half of the experiment. Interestingly, RMSProp shows a slower learning curve compared to Adam but consistently improves.

By the 10,000 sample mark, SGD and Adam achieve near parity at roughly 89.5% accuracy. This suggests that for mid-depth residual networks, the choice of optimizer is critical when the labeling budget is extremely tight (< 4000 samples), but becomes less significant as the budget expands.



(a) Performance Comparison of Optimizers on ResNet-18

(b) Performance Comparison of Optimizers on ResNet-34

(c) Performance Comparison of Optimizers on ResNet-50

Figure 3: Performance comaprison of 3 architectures (ResNet-18, ResNet-34, ResNet-50) on 4 different optimizers (SGD, ADAM, ADAMW, RMSProp).

6.3 ResNet-50 Performance

The deepest architecture, ResNet-50 (Fig. 3c), reveals the most pronounced difference in optimizer behavior. At the initial 1,000 sample mark, SGD performs poorly ($< 70\%$ accuracy), likely due to the difficulty of optimizing a high-capacity model with very little data (overfitting risk). In contrast, Adam starts strong at $\approx 75\%$.

However, ResNet-50 with SGD exhibits the steepest learning curve, eventually overtaking all other optimizers by the 6,000-sample mark and finishing with the highest accuracy of all experiments ($\approx 90\%$). This highlights a crucial trade-off: deeper models combined with SGD require a larger "warm-up" period of labeled data but ultimately yield the best decision boundaries.

7 CONCLUSION

In this paper, we conducted a systematic evaluation of ResNet architectures and optimization algorithms within a Loss Prediction-based active learning framework. Our experiments on the CIFAR-10 dataset demonstrate that there is no single "best" optimizer for all stages of active learning. Instead, we observed a distinct cross-over effect: adaptive optimizers (Adam, AdamW) are significantly more sample-efficient in the early, low-data regimes, while SGD with momentum provides superior generalization as the labeled dataset grows.

Specifically, for the deepest architecture (ResNet-50), SGD started with the lowest accuracy but concluded with the highest, suggesting that deeper models benefit most from the regularization effects of SGD once sufficient data is available. These findings imply that future active learning systems could benefit from a hybrid optimization strategy—utilizing Adam for the initial "cold start" cycles to rapidly acquire informative samples, and switching to SGD in later cycles to maximize final model accuracy.

Limitations and Future Work. A limitation of this study is ;the CIFAR-10 dataset; as it is the only dataset for performance benchmarking. In more complex tasks, results may vary; for e.g., on higher resolution datasets like ImageNet. Additionally, retraining from scratch at every cycle is computationally expensive. Future work can explore hybrid optimization strategies—starting with Adam and switching to SGD—and investigate fine-tuning techniques to reduce the computational overhead of the active learning pipeline.

References

[1] B. Settles, *Active Learning*. in Synthesis Lectures on Artificial Intelligence and Machine Learning. Cham: Springer International Publishing, 2012. doi: 10.1007/978-3-031-01560-1.

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

[3] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[4] D. Yoo and I. S. Kweon, "Learning Loss for Active Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 93-102.

[5] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International Conference on Machine Learning (ICML)*, 2013, pp. 1139-1147.

[6] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26-31, 2012.

[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[8] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations (ICLR)*, 2019.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York: Springer, 2009.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The comparative nature of the study is stated in the abstract and introduction The findings regarding the trade-off between sample efficiency and final accuracy is discussed, and are supported by the Results section.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: There is a dedicated paragraph "Limitations and Future Work," in the Conclusion, which discusses the restriction to the CIFAR-10 dataset and the computational cost of retraining.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: This paper presents an empirical comparative study and does not propose new theoretical theorems or proofs.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The Experiment Setup section details the architectures, specific hyperparameters (learning rates, momentum, weight decay), the query strategy (Loss Prediction), and the budget protocol used.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The CIFAR-10 dataset is publicly available. A link to the code repository is provided with the author information.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The Experiment Setup section explicitly lists the batch size, epochs, learning rate schedulers, optimizer parameters, and data augmentation techniques.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We conducted multiple trials, for which we report the general trends in accuracy and sample efficiency in the main text and figures. This is done without explicitly plotting error bars or calculating statistical significance values in this report.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The Implementation and Computing Environment section specifies the use of Concordia University's HPC facility and NVIDIA V100 (32GB) GPUs.

9. **Broader impacts**

   Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

   Answer: [NA]

   Justification: Our project explores optimization strategies for active learning on a standard benchmark (CIFAR-10) and does not have immediate or specific societal impacts requiring discussion beyond general machine learning efficiency.

10. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The study utilizes standard image classification models and public datasets which do not pose any risk for misuse.

11. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new datasets or assets were created; this study relies on preexisting, publicly available benchmarks and architectures.

12. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [NA]

    Justification: Large Language Models were not used as a component of the core research methodology.