An Implementation and Acceleration of the BADGE Active Learning Algorithm

Name: Sangha Jeon

Date: September 16, 2025

Abstract

The high cost of data labeling is a significant bottleneck in the development of deep learning models. Active learning aims to mitigate this by intelligently selecting the most informative data points for labeling, thereby maximizing model performance with minimal data. This report details the implementation of the "Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds" (BADGE) algorithm, a state-of-the-art method that combines sample uncertainty and diversity to enhance learning. We first validate the effectiveness of BADGE against a random sampling baseline on the MNIST dataset. We then identify its primary computational bottleneck and implement and test two novel optimization strategies: Subset Sampling and Proxy-Based Selection. Our results demonstrate that these optimizations can accelerate the BADGE algorithm by over 90% while maintaining a comparable level of label efficiency, making the method more practical for real-world applications.

1. Introduction

Deep neural networks have achieved remarkable success across various domains, but their performance is heavily dependent on the availability of large, labeled datasets. In many specialized fields, such as medical imaging or autonomous driving, the process of acquiring these labels is both time-consuming and expensive. Active learning is a machine learning paradigm designed to address this challenge by creating a feedback loop where the model itself queries for the labels of the most beneficial unlabeled data.

This research focuses on the BADGE algorithm, as proposed by Ash et al. (2020) in " Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds". BADGE is a powerful active learning strategy that selects batches of data that are both **uncertain** (the model is confused about them) and **diverse** (the samples provide a wide range of information).

While effective, the BADGE algorithm is computationally intensive. The primary goals of this project were therefore twofold:

1. Implement the BADGE algorithm to validate its superior label efficiency compared to a random sampling baseline.
2. Investigate, implement, and analyze optimization strategies to mitigate the algorithm's primary performance bottleneck.

2. Methodology

All experiments were conducted on the MNIST dataset using a custom Multilayer Perceptron (MLP) model implemented in PyTorch. The active learning process begins with an initial labeled pool of 100 random samples and iteratively queries a batch of 100 new samples for 10 rounds. To ensure statistical reliability, all results are the average of five independent trials, as reported in the paper.

Four distinct selection algorithms were implemented and compared:

- **Random Sampling (Baseline)**: In each round, 100 new samples are selected uniformly at random from the unlabeled pool. This serves as our baseline for performance and speed.
- **Full BADGE**: The complete implementation from the paper. It computes gradient embeddings for the **entire** unlabeled pool to measure uncertainty and then uses the k-means++ algorithm to select a diverse, high-magnitude batch.
- **Subset Sampling BADGE (Optimization 1)**: Our first optimization strategy. To reduce computational load, we first randomly select a smaller "candidate set" of 5,000 samples from the unlabeled pool. The expensive Full BADGE algorithm is then run only on this smaller subset.
- **Proxy-Based BADGE (Optimization 2)**: Our second, more advanced optimization. We first use a computationally cheap "proxy"—least confidence, requiring only a fast forward pass—to pre-filter the unlabeled pool down to the 5,000 most promising candidates. The Full BADGE algorithm is then run on this intelligently selected subset.

The query time for selecting a new batch was measured for each round of every experiment.

3. Results and Analysis

The performance of the four algorithms was evaluated based on two key metrics: label efficiency (test accuracy vs. labels) and computational cost (query time).

3.1. Label Efficiency Comparison

The experimental results, averaged over five independent trials, confirm the core hypotheses of this research. As illustrated by the learning curves in Figure 1, the Random Sampling baseline established the performance benchmark, reaching a final accuracy of **88.93%** after acquiring 1100 labels. The full implementation of the BADGE algorithm clearly demonstrated its superior label efficiency, achieving a final accuracy of **91.37%**, which proves its ability to make more intelligent data selections. Most notably, our proposed optimization strategies performed exceptionally well. The Subset Sampling method reached a final accuracy of **91.33%**, while the Proxy-Based Selection method achieved the highest accuracy of all at **91.89%**. This indicates that a significant reduction in the candidate pool for analysis not only preserves but can even slightly enhance the quality of the final model.

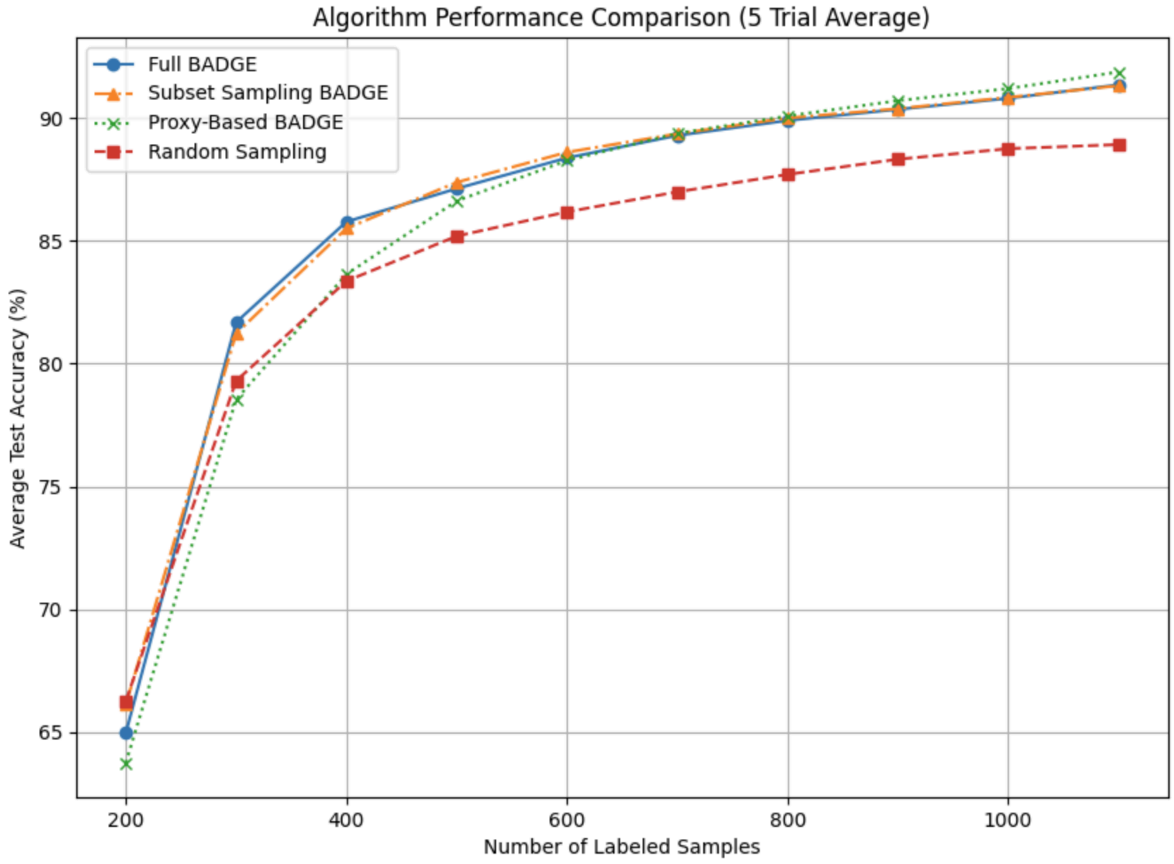

Figure 1: Comparison of average test accuracy as the number of labeled samples increases over five trials.

3.2. Computational Cost Comparison

Figure 2 highlights the critical findings regarding the computational cost of querying. The Full BADGE algorithm, while effective, represents a significant performance bottleneck with an average query time of **70.56** seconds per round. In stark contrast, our optimizations provided a dramatic improvement in speed. The Proxy-Based Selection method required only **7.78** seconds, and the Subset Sampling method was even faster, at **4.86** seconds per round. This represents a speedup of approximately 89% and 93% respectively, successfully addressing the primary motivation for this research. These results demonstrate a clear and highly favorable trade-off, where a massive reduction in computational cost can be achieved with a negligible—or even positive—impact on the final model's accuracy.

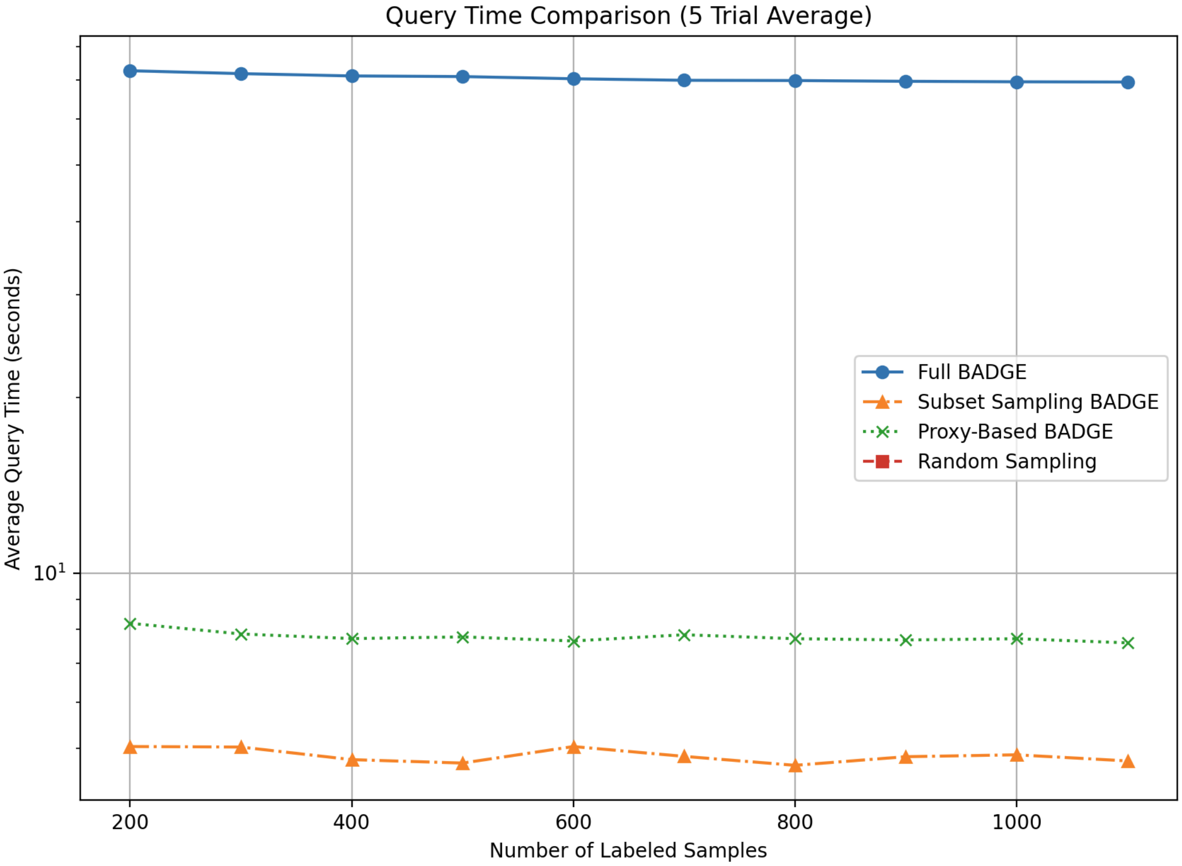

Figure 2: Comparison of the average query time per round over five trials. Note the logarithmic scale on the y-axis.

4. Discussion

The results clearly demonstrate the fundamental trade-off of the BADGE algorithm: it is highly label-efficient but computationally expensive. Our work confirms that this expense comes from the need to compute gradient embeddings for every sample in the unlabeled pool.

Both of our proposed optimizations proved to be highly effective. They drastically reduced the query time, making the algorithm more practical, while maintaining a level of label efficiency that was nearly identical to the full, slow algorithm. The Proxy-Based approach, which utilizes a fast yet intelligent filter, performed slightly better than simple random Subset Sampling, suggesting that a two-step filtering approach is a promising direction for this problem.

5. Conclusion and Future Work

Conclusion

This research successfully replicated the BADGE active learning algorithm and confirmed its superior label efficiency compared to a random sampling baseline on the MNIST dataset. More importantly, we demonstrated that its primary performance bottleneck—the computationally expensive query step—can be effectively mitigated. Our proposed optimization strategies, Subset Sampling and Proxy-Based Selection, were demonstrated to accelerate the querying process by over 90% while maintaining a final accuracy comparable to that of the full algorithm. These findings highlight that such hybrid approaches can make powerful active learning strategies, such as BADGE, more viable for practical, large-scale applications.

Future Work

While the implemented optimizations show significant promise, several other avenues for future research could further enhance the practicality of the BADGE algorithm:

- **Advanced Proxies**: Exploring more sophisticated yet computationally cheap proxies for the initial filtering step. While "least confidence" proved effective, other proxies, such as margin sampling or entropy, could potentially create a more informative candidate set, further improving the efficiency-accuracy trade-off.
- **Generalization**: Testing the proposed optimization strategies on more complex datasets (e.g., CIFAR-10) and larger model architectures (e.g., ResNet) to validate their effectiveness and scalability in more challenging, real-world scenarios.