CIFAR-100 Classification: A Comprehensive Analysis of Methodology, Results, and Comparative Study with Current Approaches

Gabriel Ardeleanu, Andreea Arsene

January 2024

Abstract

This paper addresses the CIFAR-100 classification problem, aiming to assign 60,000 images to one of its 100 classes accurately. Our research employs a neural network trained to determine the highest match percentage as the image class. Our methodology leverages a Convolutional Neural Network with normalized batches, incorporating architectural techniques to enhance machine learning implementation results. Results show robust performance in image categorization, with our algorithm demonstrating comparable outcomes to recent studies. This paper contributes a detailed account of the implemented methodology and insights into achieved outcomes, providing a benchmark for future research in CIFAR-100 classification.

1 Introduction

The task of CIFAR-100 classification involves the meticulous assignment of 60,000 images into one of its 100 classes, presenting a formidable challenge at the intersection of computer vision and machine learning. The complexity of this problem lies not only in the vast dataset but also in the intricacies of accurately categorizing diverse images. Image classification, a fundamental aspect of computer vision, finds practical applications in various domains, including autonomous vehicles, medical image analysis, and content recommendation systems.

In addressing the challenge posed by the CIFAR-100 problem, we strive for high accuracy and aim to actively contribute to advancing image classification methodologies. Our approach involves thorough consultation with state-of-the-art sources, seeking inspiration from recent breakthroughs and employing sophisticated techniques. Specifically, we focus on a source [1] that grants access to a wealth of information and provides detailed rankings of prior solutions in the

[1] Source: https://paperswithcode.com/sota/image-classification-on-cifar-100

CIFAR-100 classification. By utilizing this source, we not only gain valuable insights but it sets the base for a comprehensive comparative analysis, enabling us to benchmark our results against top-performing solutions and potentially enhance our methodologies based on lessons learned from the most successful implementations.

Our methodology relies on a Convolutional Neural Network (CNN) for effective data analysis, utilizing normalized batches to ensure stable and efficient training. The inclusion of renowned architectural techniques enhances the model's performance. Employing an optimizer scheduler for learning rate, our approach dynamically adjusts the learning rate for improved convergence and generalization, fostering simpler and more generalizable representations. Additionally, a dropout technique effectively removes a set percentage of the used images, boosting the chances of avoiding overfitting in our model.

Our paper details the process of refining a base implementation, outlining the strategic enhancements employed to address challenges. The comparative analysis of our results against state-of-the-art entries for the same problem showcases our methodology's standing. Despite not reaching the top of the list, our approach places us in a comfortable position within the ranking, underscoring the viability and competitiveness of our contributions to the field.

2 Related work

In exploring the results of related works in CIFAR-100 classification, an initially promising implementation particularly piqued our interest. The source under consideration presents a similar objective of tackling the CIFAR-100 problem, employing the EfficientNet architecture and leveraging a pretrained model for enhanced performance. A notable mention is the fact that in the found source, the algorithm proceeds to classify the results in 1000 classes instead of 100. The selected source, however, presented noteworthy challenges in terms of achieving high accuracy. Despite the utilization of the EfficientNet architecture and the advantage of a pretrained model, the reported results indicated an accuracy level of approximately 60 percent using nothing but the simple model, and up to 82 percent after applying various enhancing tehniques. This outcome prompted us to examine the runtime efficiency and overall viability of the methodology. It is clear from the results that the process experiences a slow convergence, not only evidenced by the total runtime—extending almost two hours—but also by the small gradual improvement in partial results observed throughout several hundred epochs. The slower pace of increase in validation accuracy implies potential inefficiencies in the training process or architectural choices, allowing further improvements to achieve better results.

One noteworthy entry [2] stands out as the recent best implementation, leveraging a method known as SparseSwin. This implementation intricately combines

[2]Source: https://paperswithcode.com/paper/sparseswin-swin-transformer-with-sparse

the Swin Transformer architecture with sparse transformer blocks. The Swin Transformer, initially introduced as a vision transformer variant, has demonstrated remarkable capabilities in capturing long-range dependencies within images. The integration of sparse transformer blocks refines this architecture, introducing sparsity patterns to enhance computational efficiency. The resulting implementation achieved an impressive accuracy of 85.35 percent, positioning it as a benchmark for comparison with other recent entries.

Standing as the overall best entry [3] in recent studies, an implementation with an exceptional final accuracy of 96.02 percent, was realized through the application of Sharpness-Aware Minimization (SAM), a sophisticated method designed for efficiently improving generalization in machine learning models. SAM operates by minimizing a sharpness-aware objective, effectively enhancing the model's ability to generalize well to unseen data. The utilization of this technique has set a new State-of-the-art goal, pushing the boundaries of classification accuracy. With its noteworthy performance, this overall best entry sets a high standard for image classification.

3 Methodology

3.1 Architectural improvements

As an initial step, we decided to re-implement the algorithm inspired by the source discussed in the Related Work section, specifically a Convolutional Neural Network (CNN) model employing batch normalization. The outcome of this re-implementation, while presenting slightly lower to comparable results, emerged as an overall superior solution. The key advancement is a substantial reduction in the total runtime, which was by a significant 20-30 minutes. The introduction of a caching function for the dataset involves moving all data to the GPU, and post pretraining, we eliminate the cached part. Additionally, to optimize the batch size, we utilize higher dimension sizes for the images during pretraining. This efficiency enhancement signifies an important step in achieving a more time-effective and resource-efficient solution for the CIFAR-100 classification problem.

This CNN is configured with four convolutional layers, each strategically designed with input and output channels ranging between 128 and 256, culminating in a final layer with 512 channels. The activation function for these layers is linear, complemented by the integration of dropout with a 0.5 rate to mitigate overfitting. Notably, our experimentation led us to observe that the addition of two extra convolutional layers did not yield substantial improvements in addressing overfitting. Moreover, alterations in the number of channels, whether an increase or decrease, failed to significantly impact the model's susceptibility to overfitting.

[3] Source: https://paperswithcode.com/paper/sharpness-aware-minimization-for-efficiently-1

We introduce a Transform function that plays a key role in augmenting our dataset. Applied to our input images, the Transform function strategically modifies information by incorporating various filters and noise patterns on a small scale. This approach ensures that the essence and content of the original images are retained, preventing a complete loss of valuable information. The Transform function acts as a way to multiply the effective number and variety of training data, providing a diverse range of inputs for the model. By introducing subtle variations through transformations, we aim to boost the model's generalization capabilities, facilitating a desired performance when confronted with diverse images during evaluation.

To further enhance the training process, we incorporate two optimization techniques: optimizer scheduler for learning rate and dropout. The optimizer scheduler for learning rate introduces dynamism into the algorithm by varying its values during training, allowing the model to explore different learning rates, promoting faster convergence and improved generalization. Concurrently, dropout, a regularization technique, effectively reduces the number of images used for training, improving the chances of avoiding overfitting. This strategic combination contributes to the overall robustness and efficiency of our training regimen for the CIFAR-100 classification task.

While our independent implementation of this model exhibited promise, we chose to leverage a similar pretrained model, augmenting it with additional regularization techniques like dropout. Furthermore, adaptations were made to the model's function to ensure proper functionality for handling 100 classes instead of the original 1000.

3.2 Application Flow

The project begins with elementary adjustments to the input dataset, strategically caching it directly on the GPU. This proactive measure ensures optimal efficiency by mitigating the communication overall time associated with transferring data back and forth between the GPU and CPU. The application of One Hot Encoding transforms the categorical labels into a format conducive to neural network training. The dataset, comprising 60,000 images, is judiciously partitioned, with 50,000 images marked for training and 10,000 for validation, adhering to the established practice of maintaining the validation set at approximately 20

Moving into the training phase, the neural network follows a conventional flow. It utilizes the information from the input to calculate values for each layer, progressively moving towards the computation of the temporary result for each image. Immediately after, the loss is calculated with respect to the actual output, triggering the application of backpropagation to iteratively adjust the weights and biases. During this stage, optimization steps, such as scheduling learning rate and dropout, may be employed to fine-tune the model's performance.

Upon the conclusion of the training session, the project transitions into the vali-

dation phase. Here, the system records correct predictions, incrementally boosting the overall accuracy of our implementation. This iterative and comprehensive flow encapsulates the essence of our approach to addressing the CIFAR-100 classification problem.

4 Results

In showcasing the outcomes of our implementation, we provide compelling evidence that the methods and enhancements discussed have notably elevated our system's performance. Employing strategic architectural improvements, dataset transformations, and optimization techniques, our model has undergone a transformative journey. In addition, we will present results obtained from experiments involving varied values for main parameters and the selective exclusion of certain method improvements. While our enhancements successfully improved the initial implementation, the achieved results place our system within the range of the overall majority of recent studies, boasting an accuracy of approximately 84 percent. Unfortunately, we are not able to fully compare every implementation due to lack of information, as certain parameters were not found in the related work's documentations.

Implementation	Ours	Initial Refference	Recent Best	Best Overall
Best Accuracy	83	82	85,35	96,08
Epochs	28	15	100	200, up to 400
Batch Size	32	8	12	512
Run Time(∼mins)	90	120	Not Reported	Not Reported

Table 1: Result comparison

Despite our results being limited to comparisons with recent best entries and the absence of a key parameter necessary for a comprehensive comparison, the evident outcome is that our project comfortably resides within the sphere of comparable results. While acknowledging the inherent limitations, the achieved performance showcases a competitive standing among contemporary implementations. The absence of certain parameters for comparison does not diminish the notable strides made by our project, proving its capacity to be placed within the current landscape of image classification methodologies.

5 Conclusion

In approaching the CIFAR-100 problem, our initial research aimed at navigating guiding papers and benchmarking our progress against comparable results. This groundwork set the stage for our subsequent endeavors.

Our implementation was designed to enhance an initially suboptimal entry, employing renowned architectures for image classification problems. We addressed

key issues and optimized performance through strategic enhancements.

Upon achieving satisfactory results, we conducted a thorough comparison with state-of-the-art papers. Our project consistently achieved around 83 percent accuracy, placing it within the majority of comparable results. However, a notable gap was observed compared to the best results, which surpassed the 95 percent accuracy threshold.

Despite this distinction, our implementation holds the potential to yield commendable results for the CIFAR-100 problem. Our modest placement in the ranking of presented entries reflects a respectable standing within the contemporary landscape of solutions, highlighting the value and contributions of our approach to image classification methodologies.

6 Bibliography

Advanced Topics in Neural Networks subject:
https://sites.google.com/view/atnn/home?authuser=0

State of the art Benchmarks:
https://paperswithcode.com/sota/image-classification-on-cifar-100

Initial Refference:
https://towardsdatascience.com/cifar-100-transfer-learning-using-efficientnet-ed3ed7b89af2

Recent Best Implementation - SparseSwin:
https://paperswithcode.com/paper/sparseswin-swin-transformer-with-sparse

Best Implementation Overall - SAM:
https://paperswithcode.com/paper/sharpness-aware-minimization-for-efficiently-
1