generated by Scienteer.

Research Plan/Project Summary

Scienteer Project ID: 204010
Category: SYSTEMS SOFTWARE
Student(s): Rohit Maheshwari and Abhiram Gadde
Project Title: Limiting Lossy Compression and Improving Efficiency for Image Processing using Convolutional Neural Networks
Division: Senior
School Name: TEXAS ACADEMY OF MATHEMATICS AND SCIENCES H S
School City: Denton, Texas, US

Rationale

Convolutional neural networks (CNNs) are a type of deep learning algorithm that is highly utilized in the background of image classification and computer vision. CNNs take input images and assign importance to various aspects or orientations of the image content to differentiate between the objects in the image itself. CNNs become more efficient by not requiring human supervision, but instead, introducing a new method to perform feature learning. The goal of CNNs is to reduce the form in which images are processed without losing the unique attributes utilized for the final image classification. Rather than primitive ideas of human classification, CNNs learn by quantifying images through different layers through filters and kernels which produce outputs that gradually become more detailed than the initial inputs.

One of the most common applications of CNNs is image classification, where the goal is to classify an image into one of several predefined categories, such as "cat," "dog," or "car." CNNs are able to achieve high levels of accuracy on this task by learning to identify key features of the input images, such as the shape of an animal's ears or the pattern of a car's headlights. Along with image classification, CNNs can be extended to a wide range of fields including, but not limited to, image and video recognition, medical image analysis, computer vision, and human-computer interfaces. By making CNNs more effective, professionals may be able to better perform tasks at a much higher speed. From image and video recognition to medical imaging and beyond, CNNs are helping to push the boundaries of what is possible with machine learning and artificial intelligence.

This project shines light on a methodology that reduces the number of computations while an image goes through a new filter or kernel in a CNN. The CNN model’s layers consist of 4-dimensional matrices which hold weights used to classify the input and provide an accurate output. These weights can be adjusted with a foundation of past CNNs and logical improvement. Through each layer, millions of matrix multiplications take place to classify the image contents in a more precise manner. However, because many of the matrices in the layers are similar, a majority of these computations can be simplified after one computation was made. For instance, if two matrices are the exact same, the matrix multiplication product would be the exact same. Therefore, the second matrix computation would yield the same result as the first computation, so it does not have to be computed. This would save computation time knowing that many of the matrices are similar. By adjusting the threshold in determining the requirements to reduce computations, these images will compress without losing too much data–hence, increasing the efficiency of the CNN model.

Creating a more efficient CNN model grants the opportunity to make machine learning more feasible for edge devices. Utilizing human strategies to reduce computations, the fast inferences reduce the amount of energy required for sustainability–lower energy yields longevity. These models can then be deployed in fields of image and speech recognition. By increasing the efficiency and time complexity, the model will not require much energy in a larger scope. With edge devices controlling data sequences and control flow between networks, a faster CNN model allows traffic flow to transmit faster, saving an abundant amount of energy in a matter of a few seconds.

Hypothesis

This project hopes to determine the characteristics regarding the threshold in defining matches between matrices. There are two main thresholds in defining a match: 1) the percentage threshold of how similar two matrices are and 2) the absolute value range of proximity between two integers in a matrix. These thresholds will be tested with the expected outputs to determine which percentage and absolute value range threshold optimally yield the expected outcome while reducing as many computations as possible. Once these matches are defined, the number of computations will be reduced to increase the efficiency of the CNN model.

Materials

The following materials were used in this research project:

- Laptop
  - Microsoft Visual Studio Code Integrated Development Environment
- Python
  - Numpy Library
  - CSV Import
- Lutz Roeder's *Netron* application
- Pruned Tensor Files from MLPerf™ Tiny Deep Learning Benchmarks (GitHub)
- Microsoft Excel

Procedure

4-Dimensional Input Matrix

The four dimensions of the input matrix organize the weights of a tensor. The first dimension represents the number of cubes that exist in the tensor. The second dimension represents the number of filters that exist in a cube. The third and fourth dimensions represent rows and columns, respectively, in each filter of the cube. The input will be iterated through the third dimension to test whether the filter matches with at least one other cube.

Matches/Threshold Identification

The definition of determining a match relies on two adjustable thresholds. As mentioned in the purpose, the thresholds are the percent similarity between two matrices and the differences of the corresponding (same indices) integers in the two matrices. With these two sliders for thresholds, the definition of a match will alter based on the chosen similarity percentages and the chosen differences between integers. Depending on the number of matrices that are matched, the amount of computations for all the matching matrices will become one, which increases the efficiency and reduces time complexity while traversing the CNN layers.

High-Level Method of Approach

To check the validity of the project's objective, a program can be used to first collect statistics for comparisons. By inputting pruned tensor files directly into Roeder's Netron application, NumPy arrays can be received and placed back as input into the proposed program. With Python's NumPy library, the filters can be individually checked across each pair of cubes in the 4-dimensional array of the input. The goal, in this case, is to identify how many matches/duplicates can be found across a series of cubes to ensure that computations can indeed be reduced. If duplicates are found, the computation for the initial filter can be reused for the matched filters without any need to redo further computations. Similarly, this process can be applied to each cube to reduce computation on a larger scale. This benchmark data can be compared to brute computations through Microsoft Excel data analysis to test the efficiency of this method.

Data Analysis

The CSV (Comma Separated Values) file will use exported values from the algorithm to observe four pieces of data. The four-dimensional matrices from the pruned Resnet files are eighty percent sparsed with zeros to replicate the weights of a true input image. But this sparsity will be adjusted with different percentages to capture all potential images. The observed values include 1) the number of filters that

match and are not all zeros, 2) the number of filters that are all zeros, 3) the number of cubes that match and are not all zero, and 4) the number of cubes that are all zeros. The definition of “matching matrices” will change based on the method used to create and test different thresholds (the percentage similarity of the matrix and the absolute value offset between the integers in the matrix). These values for each threshold will be plotted to determine the percent error between the CNN model’s prediction and the true/expected result. Determining the most optimal algorithm may reduce computations or may not reduce computations, but this data will be determined to explore the results of this methodology to potentially reduce the number of computations while traversing the layers in a CNN.

generated by Scienteer.

Project Summary

Changes from Original Research Plan

Project Conclusions

Bibliography

Reference Source

Awati, R. (2022, September 29). convolutional neural networks. Enterprise AI. Retrieved December 10, 2022, from www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network#:~:text=A%20CNN%20can%20have%20multiple,can%20start%20as%20simple%20features.

Han, S., Mao, H., & Dally, W. J. (2015). Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149.

Mishra, M. (2020, August 26). Convolutional Neural Networks, explained. Medium. Retrieved December 10, 2022, from www.towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

Prabhakar, R. B., Kuhar, S., Agrawal, R., Hughes, C. J., & Fletcher, C. W. (2021, June). SumMerge: an efficient algorithm and implementation for weight repetition-aware DNN inference. In Proceedings of the ACM International Conference on Supercomputing (pp. 279-290).

Saha, S. (2018, December 15). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. Medium. Retrieved December 10, 2022, from www.towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53