CMSC858N: (Course Project Report) Scalability of Parallel Similarity Search to couple with Deep Learning

Gihan Jayatilaka
Department of Computer Science, University of Maryland in College Park
`gihan AT umd DOT edu. May 15, 2023`

1 Introduction

Image classification is the most basic task of computer vision. This has been studied for a longer period of time with classical, modern learning, and hybrid algorithms. This work attempts to evaluate the performance of similarity search as a tool for image classification. Similarity search is used after encoding the images by a neural network. We try to evaluate a parallel implementation of similarity search for its runtime as well as accuracy for the task. This report presents an overview of the individual components used in the related work section. Our experimental setup is mentioned in the methodology section. The time and accuracy results are presented and analyzed next. Finally, several conclusions are drawn from the observation.

2 Related work

Neural Networks have been the state-of-the-art (SOTA) for image classification since 2012[12][1]. Preliminary information about the neural networks is omitted in this report since it is not the focus area of the project. Almost every Neural Network architecture has two components; namely, the embedding backbone and the classification head. The general idea is that the backbone (mostly convolutional) will transform the image into a meaningful representation that could be classified by the head (mostly fully connected). Generally, the backbones are deep and the heads are shallow. As of today, there are backbones that go hundreds of layers deep [17]. The decision heads have been 1 or 2 layers in the past. Modern architectures have heads that are only one layer deep [8]. Recently, vision transformers [7] based on self attention has replaced (or complemented) convolution backbones.

Datasets are an integral part of image classification problems in both developing algorithms/systems (ie: training data) and evaluating the systems (testing and validation). The datasets are of varied image size, count and diversity. In addition they are of varied difficulty (determined by the number of classes and the fine/coarseness of labels). The smallest/easiest of the spectrum are MNIST [15], CIFAR 10 [10], CIFAR 100 [11] which deal with small images of handwritten digits and images. Imagenet [6]is a larger dataset. Tiny Imagenet is a smaller version of the same dataset [14].

Similarity search is a classical problem in algorithms. This line of work attempts to find the most similar elements among a set of elements to a given element. Approaches differ in this domain according to what is measured as similarity and what number of similar elements are searched for. Nearest Neighbour Search is a kind of similarity search when the set of elements are on a metric space (a space with a distance function defined on every pair of elements). A generalized version of this is the k-Nearest Neighbour search (KNN) where the k-closest elements are picked instead of one.

Parallel implementation of algorithms refers to implementing an algorithm such that it can utilize more than one processing unit at a time. Processing units can be either CPU, GPU, TPU or application-specific processor cores. In general, parallel implementations speed up the algorithms. As far as

[1]paper published in 2017

neural networks for large-scale computer vision tasks are concerned, they have had parallel GPU implementations for as long as they were performing well [9, 1, 16]. Neural networks (on both their parameter optimization and prediction steps) are inherently parallel. Because of this reason, most neural network based parallelization is not studied rigorously. It is mostly considered to be an implementation detail.

Parallel implementation of other algorithms (such as similarity search) is both a rigorous theoretical research area as well as a practical engineering field. FAISS [5] is a library that supports parallel similarity search on both CPU and GPU. The library goes for approximate k NN search for time efficiency. The CPU acceleration is achieved by multi threading using the BLAS library (originally [13]). In addition, the SIMD instructions are used under the hood for faster computation. The GPU acceleration is done with the CUDA architecture in mind. The implementation exploits the memory and register architecture of CUDA to gain maximum performance.

3 Methodology

This work formulates the image classification problem as a two stage problem. Given $(x_i \in R^{H\times W\times 3}, y_i \in [C])$ being an image and it's true label, we find the predicted label $\hat{y}_i$. Let the training set be (X_i^t, Y_i^t) and test set be (X_i^s, Y_i^s). The neural network (f) embeds the x_i's into $f(x_i) \in R^D$. The similarity search algorithm (S) finds k nearest neighbours to $f(x_i^s)$ given embeddings of all the training data points $f(X_j^t)$. We infer the predicted label of x_i based on the mode of the true labels of the k nearest neighbors.

We run this experiment using ViT as the embedding neural network on Imagenet[2] and Tiny Imagenet datasets. We measure the time it takes to run the experiments. We report the results as the percentile accuracy of the predicted labels and the time taken for embedding and similarity search.

All embeddings are done on 2 Nvidia A5000 GPUs. All CPU tests for similarity search are done on server nodes with 32 processor cores and 64 GB RAM. All GPU tests for similarity search is done on similar server nodes with an additional Nvidia A5000 GPU (of 24GB VRAM). Experiments are implemented in python (including time measurement).

4 Results and Analysis

First, we measure the time taken for the neural network embedding. This result is shown in Table 1. As per the results, modern GPUs can do the embedding step for the complete Imagenet dataset in around an hour.

Table 1: Time Taken for Embedding

Dataset	**Time Taken (mm:ss)**
Tiny Imagenet	4:58
Imagenet (10% of the dataset)	8:56
Imagenet (full dataset)	64:22

Then, we measure the time taken for similarity search using the FAISS library. The results are reported in Table 2. We measure the time taken to build the index, query the index for the training examples and query the index for the validation set examples.

The time taken to build the index is almost negligible for both CPU and GPU. This implies that no heavy lifting is done on the index-building step. Querying the training and validation sets take time comparable to the size of the dataset. It should noted that CPU time for querying passes the time for embedding when the dataset is larger (imagenet).

In addition to the time consumption of the algorithms, we show the accuracy[3] of the approach tested in Table 3. Both training and validation set results are shown for our method. The column DINO has the result from the paper proposing our embedding technique[2]. This results is from the embedding

[2]We refer to Imagenet 1k dataset as Imagenet in this report.

[3]All accuracies mentioned in this report are top 1 accuracies.

coupled with a trainable decision head. The last column (SOTA) has the state of the art result for the particular dataset as of the time of writing this report [3, 4].

Table 2: Time taken for similarity search

Experiment		**Index building time (sec)**	**Similarity Search Time (sec)**	
Dataset	**Device**		**Train Set**	**Validation Set**
Tiny Imagenet	CPU	0.03	11.40	0.75
	GPU	0.02	7.94	0.76
Imagenet (10%)	CPU	0.03	15.95	6.01
	GPU	0.12	1.32	0.46
Imagenet	CPU	0.30	4689.15 *(= 78:09)*	364.18 *(=6:04)*
	GPU	0.29	108.72	4.24

Table 3: Top 1 Accuracy Comparision (percentage)

Dataset	**Our's (Train)**	**Our's (Validation)**	**DINO**	**SOTA**
Tiny Imagenet	81.66	77.73	-	92.98
Imagenet (10%)	79.70	73.07	-	-
Imagenet	83.47	77.77	80.01	91.10

The results show that kNN algorithm is a good choice of similarity search for the image classification problem. Trying to match the image embeddings to the images with known labels in the training set is a feasible approach for image classification. While this gives a direct window to the full training dataset in the query step (in contrast to total deep learning models having to learn this in their decision heads), the actual accuracy values are lagging behind. However, the gap is small enough for this approach to be valid.

Similarity search has been seeded up by parallel implementation in the FAISS library. However, it is clearly visible that GPU acceleration outperforms CPU acceleration by several folds for larger datasets. However, the difference is not clearly visible for smaller datasets. This can be attributed to multiple factors. GPUs have more cores than CPUs and they can execute more instructions at a given time period. However, copying data from CPU memory to GPU memory takes time. This overhead shadows the speedup gained by the higher core count for smaller datasets.

5 Conclusion

This work has shown the viability of using similarity search techniques for image classification problems. The experimental setup which combines vision transformer based embedding (trained as per DINO) with FAISS library similarity search (k=10 k nearest neighbor search with Euclidean distance as the dis-similarity metric) has shown competitive performance for Imagenet and Imagenet tiny datasets in terms of accuracy. The GPU-accelerated version of the algorithm reduces the runtime so this is a viable solution for modern image classification systems. In future work, this idea could be extended to multiple encoding techniques. Other ViT variants as well as Conv architectures can be explored. In addition, multiple GPU benchmarks for similarity search could be explored.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *Osdi*, volume 16, pages 265–283. Savannah, GA, USA, 2016.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[3] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, et al. Symbolic discovery of optimization algorithms. *arXiv preprint arXiv:2302.06675*, 2023.

[4] Rishit Dagli. Astroformer: More data might not be all you need for classification. *arXiv preprint arXiv:2304.05350*, 2023.

[5] Dimitrios Danopoulos, Christoforos Kachris, and Dimitrios Soudris. Approximate similarity search with faiss framework using fpgas on the cloud. In *Embedded Computer Systems: Architectures, Modeling, and Simulation: 19th International Conference, SAMOS 2019, Samos, Greece, July 7–11, 2019, Proceedings 19*, pages 373–386. Springer, 2019.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014.

[10] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). . URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[11] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research). . URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[13] Chuck L Lawson, Richard J. Hanson, David R Kincaid, and Fred T. Krogh. Basic linear algebra subprograms for fortran usage. *ACM Transactions on Mathematical Software (TOMS)*, 5(3): 308–323, 1979.

[14] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

[15] Yann LeCun and Corinna Cortes.

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.