IMAGE WAREHOUSE ARCHITECTURE IN THE ERA OF BIG DATA

Боцюра І.С.
Науковий керівник –ас. кафедри ПІ Терещенко Г.Ю.
Харківський національний університет радіоелектроніки
(61166, Харків, просп. Науки,14, каф. ПІ, тел. (057) 702-14-46)
e-mail:iryna.botsiura@nure.ua

More than 1.4 trillion photos are estimated to be taken in 2020. That's 3.8 billion photos per day or 44,4 thousand photos per second. In the foreseeable future, this figure will continue to grow. This is why it is so important to know more about different ways of storing and accessing images, when we are living in the era of big data. I would like to analyze this three methods:

- Storing images on disk as .png files
- Storing images in lightning memory-mapped databases (LMDB)
- Storing images in hierarchical data format (HDF5)

1.Storing on Disk. We save images to disk as .pngs, and name them using a unique image ID. In all realistic applications, it necessary to take care of saving meta data and there are several options to do that. One solution is to encode the labels into the image name. This has the advantage of not requiring any extra files. However, it also has the big disadvantage of forcing you to deal with all the files whenever you do anything with labels. Storing the labels in a separate file allows you to play around with the labels alone, without having to load the images.

2.Storing to LMDB. LMDB stands for Lightning Memory-Mapped Database. It is a key-value store, not a relational database. In terms of implementation, LMDB is a B+ tree, which basically means that it is a tree-like graph structure stored in memory where each key-value element is a node, and nodes can have many children. Nodes on the same level are linked to one another for fast traversal.

Critically, key components of the B+ tree are set to correspond to the page size of the host operating system, maximizing efficiency when accessing any key-value pair in the database. Since LMDB high-performance heavily relies on this particular point, LMDB efficiency has been shown to be dependent on the underlying file system and its implementation. Another key reason for the efficiency of LMDB is that it is memory-mapped. This means that it returns direct pointers to the memory addresses of both keys and values, without needing to copy anything in memory as most other databases do.

3.Storing With HDF5. HDF5 stands for Hierarchical Data Format. Interestingly, HDF has its origins in the National Center for Supercomputing Applications, as a portable, compact scientific data format. HDF files consist of two types of objects: Datasets and Groups. Datasets are multidimensional arrays, and groups consist of datasets or other groups. Multidimensional arrays of any size and type can be stored as a dataset, but the dimensions and type have to be

uniform within a dataset.

Comparison of these three methods

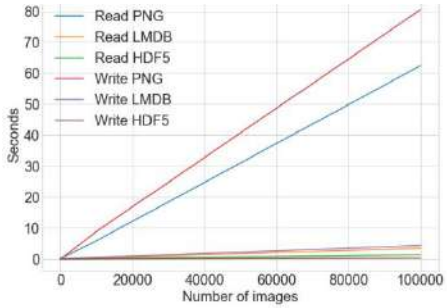

Figure 1 - read and write times of mentioned methods

The graph on Figure 1 shows the normal, unadjusted read and write times, which illustrate the drastic difference between storing to and reading from .png files and LMDB or HDF5. While exact results may vary depending on the machine, it is still seen why LMDB and HDF5 are worth thinking about. The difference between a 40-second and 4-second read time suddenly is the difference between waiting six hours for your model to train, or forty minutes.

Speed is not the only performance metric we are interested in. Dealing with very large datasets, disk space is also a very valid and relevant concern. On Figure 2 is the disk space used for each method for each quantity of images. Both HDF5 and LMDB take up more disk space than if we store using normal .png images. It is important to note that both LMDB and HDF5 disk usage and performance depend highly on various factors, including operating system and, more critically, the size of the data we store.

LMDB gains its efficiency from caching and taking advantage of OS page sizes. With larger images, we will end up with significantly more disk usage with LMDB, because images will not fit on LMDB's leaf pages, the regular storage location in the tree, and instead we will have many overflow pages. The LMDB bar in the chart above will shoot off the chart.

Conclusion: in the end, while storing images as .png files may be the easiest, there are large performance benefits to considering methods such as HDF5 or LMDB. Concerning all the advantages and disadvantages of using these three ways of storing and accessing images, we came to a conclusion, that among mentioned methods the HDF5 is the most optimal.

References

1. Internet Live Stats [Electronic resource] – Access mode: https://www.internetlivestats.com/.
2. B+-trees [Electronic resource] – Access mode: http://www.cburch.com/cs/340/reading/btree/index.html/.
3. Rebecca Stone Three Ways of Storing and Accessing Lots of Images in Python [Electronic resource] – Access mode: https://realpython.com/storing-images-in-python/.
4. S.-H. Lim, S. R. Young, R. M. Patton. An analysis of image storage systems for scalable training of deep neural networks, 2016. – 11 p.