Storing and updating large amounts of data

Tom Rochette <tom.rochette@coreteks.org>

January 20, 2020 — b834f70

1 Question

How can an agent efficiently store terabytes of data, with hundreds of gigabytes updated daily?

2 Answer

This question comes from the idea that if we want to implement an artificial intelligence, it will have to be able to process a large amount of data, similar to how we need to process a stream of sensorial (sight, hearing, taste, touch, smell) inputs actively more than 12 hours per day.

In human beings, even though we perceive a large amount of incoming data, a lot of it is compressed through differencing, that is, comparing the previous input with the new input and only storing the difference. This is similar to how video is currently encoded and compressed. To be able to accomplish this feat we however need two things: a temporary buffer to store the previous input, and a mechanism to differentiate between the previous and the current input.

That differentiation mechanism can be highly complex depending on the degree of compression desired. For example, if you shift all the pixels in an image by 1 on the x-axis, your differentiation mechanism may simply tell you that all the pixels have changed, return a delta between their previous value and new value and be done. In some cases you may be lucky and a large number of pixels have remained the same value. However, a much better differentiation mechanism would realize that everything has moved by one pixel on the x-axis and instead return to you that it detected a x+=1 transform, which compresses the transformation a lot more than by the simple pixel by pixel difference.

In the brain we make use of the fact that the sensory inputs are different modes. Each is compressed somewhat independently from the others. As such, we would expect information that is similar in format to be compressed together (text with text, video with video, audio with audio, etc.) as it is likely to lead to the highest compression. Furthermore, being able to make use of the structure within the data will lead to better compression than simply compressing blindly a collection of inputs as an opaque blob.

I would expect such a compression system to make use of two types of compression: offline and online. Offline compression would occur during low periods of activity and would be able to offer higher levels of compression. Online compression would occur when the system is actively being used.

Online compression would rely on a lookup dictionary with a most used recently retention policy to compress blocks of data that have already been seen numerous times. The quality of the online compression highly depends on the assumption that what will be observed in the future is highly likely to be like what has been observed in the past. During the day, we spend most of our time in the same environment. As such, we experience and observe the same things for an extended amount of time. Being able to determine what is similar and what is different is what will lead to the highest amount of compression.

Offline compression would rely on the ability to make the most efficient use of the compute and memory available as this process would be time-constrained. It might be possible that the online and offline systems

share information such that the online compressor can let the offline compressor know regions of data that might be ripe for recompression. In the case that both systems do not communicate, the offline system would likely benefit from knowing which regions have already been compressed to the fullest so that it spends most of its time processing data that was recently added. When it is done with this step, it can then attempt to increase the compression efficiency of all the data stored. Here again it should be able to make use of the differencing approach given that days will likely be highly similar. As such, we would expect the amount of space necessary to store a day to decrease drastically as more and more days of data are observed, possibly to the point where new days of data can be expressed as segments of previous days entirely.

3 References

- Compression
- Rationale for a large text compression benchmark