Why Data Compression Is Important And Fun

Both lossy and lossless data compression are a fun, I will talk more about the area of my expertise, lossless data compression. As you probably know, typical lossy data compression systems make use of lossless compression at the end. Here's a trivial example of a lossless data transform, and on the next slide, an example lossy data transform. Swapping bits is reversible, discarding is not.

So why fun? First, you can win cash in data compression competitions, as I did a few times, see Hutter Prize for example. Next, I'd say there's a delightful zoo of lossless compression methods. And many of them were discovered just recently, for example methods based on Asymmetric Numeral Systems a dozen years ago, and even the Burrows-Wheeler Transform, just 20 something years ago.
How do you know whether you can invent yet another method until you try? You should try to invent data compression methods to see how fun this is, and whether you have enough luck, expertise and imagination. You will very likely learn a lot, and have lots of fun in the process.

More often the typical data for your algorithm are either provided to you or picked by you, and the case with test data is even more fun, because you can try to model how your test data were generated, and then find transforms that improve performance of the next compression steps, which are usually called context modeling and entropy encoding.
Here are my top three most interesting tips about transforms:
1. Typically they are simple, more or less, their implementations run fast, but
2. It is usually a set of transforms that gives you a competitive advantage against the existing algorithms, both general purpose and rather specialized.
3. Such transforms don't have to reduce data size, they can even increase size of data, as long as further compression steps run better on transformed data.

For example, switching end-of-line markers in a text or html file from Unix-style one byte to Windows-style two bytes increases uncompressed size, but in some cases may reduce compressed size.

Why are compression methods so different?
Context modeling methods assume elements of data depend on contexts somehow, and entropy encoding methods assume they are independent.
Also, some compression methods assume data are quantitative, like samples, while other methods assume data are qualitative, like symbols, and my experience says this is the most important distinction. Also, some methods may work given a few bytes at a time, that is, with seemingly infinite streams, while other methods need a block of data to start operating well. For example, Burrows-Wheeler Transform is barely competitive on blocks smaller than 10 kilobytes, while copy-paste methods, the so-called LZ77 family, can compress five bytes and then wait for further input. DCT as used in JPEG images uses blocks of 64 pixels, they become visible when compression ratio gets too big, 64:1 or more.

Why is data compression important?
First of all, the vast majority (by volume) of modern Internet traffic is compressed data. By the way, it's very likely that interstellar communication between technologically advanced civilizations, if they exist, is also usually compressed. And compressed data look like random noise, even if produced by a rather simple compression algorithm.
Second, almost all of the video you watch is decompressed video. The vast majority of images you see on tablets, phones, laptops and desktops, are decompressed images. You understand the world better once you learn the fundamentals of how they are compressed and then decompressed, and why, for example, you should not convert photographic images to PNG image format, and black and white scans of documents to JPEG format. In particular, PNG tries to compress output of a filter, which is pretty noisy in case of photographic images, with a copy-paste algorithm. That's why PNG performs so miserably on photographic images, see for example Lossless Photo Compression Benchmark. By the way, four image codecs written solely by me are on the Pareto Frontier in the Lossless Photo Compression Benchmark. And you can see what Pareto Frontier is and how it looks for English text compressors in the Large Text Compression

Benchmark.

Advanced data compression methods can be used to measure descriptive complexity of a file, and similarity between files. Descriptive complexity of a block of data, often called Kolmogorov complexity, is defined as the length of the shortest computer program (in a predetermined programming language) that produces this block of data as output. Interestingly you can easily compute an upper bound for the descriptive complexity of a file, but computing the lower bound is only possible for a fixed decompressor code, and only by brute force, that is, by an exhaustive search. In general it is not possible. You can’t say you have hit the limit, except for very small data blocks.
I am sure all basic data compression methods will prosper and find new important applications in future. For many reasons, for example, our resources in the material world are always limited. Also both compression and decompression(!) of seemingly random data can help you see how random it really is. By the way, for me personally it is important that highly complex compression methods cannot be used in deadly weapons, only the rather simple compression methods.

Here's a couple of practical advices for those who want to win cash in data compression competitions. First, try to find fast and simple transforms that improve significantly the work of the context modeling methods that you picked. Second, for deeper understanding of data and your compression algorithm, you should create and utilize various graphs and plots, visualizing not only the data you compress, but also the internal data produced by your algorithm, your intermediate data.

Let's have a look at a couple of next big achievements (for our civilization) that might be strongly connected with the concept of descriptive complexity. Disclaimer: what I say further in my talk is speculative, often very speculative

---

First, let's call it Standard Model Of Psychology, a rather compact model that will be fundamental for theories able to predict pretty well a lot, from choice of profession and political preferences to interpersonal relations.
Why are astrology and MBTI so popular, despite of all the scientific evidence against them? They try to occupy the empty spot where we must have the Standard Model Of Psychology. Questions to the audience.
-- How many people here are familiar with MBTI? Please raise a hand if you are.
-- And how many have heard of the Big Five personality inventory? Thank you!
-- Third and last question: how many of you know that there are correlations between four MBTI scales and four of the Big Five personality constructs?
Well, there are correlations, and they are rather strong, so maybe what MBTI and Big Five are trying to explore is somewhat similar, but what could it be, at the core? Psychology has no answer yet.

How could descriptive complexity or data compression methods be involved?
First, if you try to pack as much information as possible into four bits, as MBTI does, this is a lossy data compression process. How much data about personality of a normal, healthy human do you discard when packing all the data into four bits? Most likely much more than 50%. But how many bits would you need to achieve 90% accuracy predicting previously unseen behaviour in specified circumstances? I'd guess not more than 30 bits. Note the Standard Model Of Psychology can hardly be used in a deadly weapon, but it can be used as a strategic weapon.

Okay, what could the most valuable bits describe, except gender, age, things like that? If at least some of the data processing principles are applicable to data processing in humans, then more likely those are the most fundamental principles. One of the data analysis core concepts is that the variables are either categorical or numerical. Categorical are non-comparable, and numerical are comparable. The most important types of data, as far as I know, are quantitative, where elements are... let's call them ***samples***, and qualitative data, let's call the elements of qualitative data ***symbols***. Samples need only a couple scales and a source type to describe them, for example are they coming from an audio source, a visual source, a temperature indicator? Symbols need a reference to something bigger, like for example an ASCII table. This “something” increases descriptive complexity, but when you create symbolic systems, and when you create new symbols, you normally reduce descriptive complexity a lot more. Thus, this is kind of a data compression process. By the way, executable code clearly must contain symbols, it can't be made of samples only.

The other biggest difference between cognitive functions of humans could be that there are functions responsible for the inner reality, like your senses and feelings, and for outer reality, with things like human language, logic, imagination, perhaps intuition, and so on. Thus we can construct four cognitive functions, and maybe they somehow correspond to the functions explored by Carl Jung and named Sensation, Feeling, Thinking, Intuition? Or maybe there's something similar but different? Anyway, if you arrive at a model with, say, five fundamental cognitive functions, there must be an explanation why the set is complete, why there are five of them rather than four or six, okay?

Another big item on our civilization's road map is the ultimate nature of reality. How could reality be finite in time or space? Are elementary particles really indistinguishable and fully described by a handful of numbers? What could that mean? Is it true or not that all mathematically possible universes are equally real? If not, is only one of them real? Why?
What do laws of quantum mechanics actually tell us about reality?
A model of the ultimate nature of reality must explain why some events happen and become reality, while other events do not, even if equally probable a priori. Hypothesis that reality at the bottom level is a huge computational structure is discussed in details in, for example, Max Tegmark's book "Our Mathematical Universe". Better understanding of the ultimate nature of reality is highly important for sciences that study the core universe laws, like physics and cosmology. But also for other sciences including biology, genomics and even psychology, because it is possible that human genome "knows" more about the ultimate nature of reality than our civilization. It's very possible that our genome makes practical use of effects that we are not aware of yet. Also, don't forget that bacteria are ubiquitous in every human body, and the sum of their genomes is orders of magnitude more complex than the Homo sapiens genome alone.

How could descriptive complexity be involved?
Well, if it is true that reality, after all, is a huge computational structure, then time could be the measure of change in descriptive complexity, and then the Big Bang point could correspond to the "seed" of our reality, a structure so simple that it could be fully described with a few million bits, or maybe even few thousand bits. As we know, entropy of a closed system never decreases over time, and descriptive complexity is well related to the thermodynamic notion of entropy, so why not? The age of our universe would be the distance to the “seed” in a tree of computational structures such that every “child” structure is separated from the “parent” structure by a quantum of time, and it is usually bigger than the “parent” in terms of descriptive complexity, but all structures in the tree contain the same "core”. Can you imagine something a bit similar to Conway's Game Of Life, but a billion times more complex? First of all, not two-dimensional, not completely deterministic, and very likely not binary.

Humans are self-aware substructures in these huge computational structures, substructures of a relatively small size, and that's probably the main reason why we don't experience all “child” structures of a “parent” structure as equally real when a split happens. The splits look for us like totally random input into our reality. Because each structure has a **finite descriptive complexity**, it has a finite number of “child” structures. On one hand, just one path in the infinite tree is a reality for each of us, but on the other hand, from a human's point of view the vast majority of paths are observationally indistinguishable. Just like many nodes with the same “time stamp” are observationally indistinguishable from a human's viewpoint.
This hypothesis provides a lot of questions, I can try to answer some of them after my talk. Thank you!