Knowledge as Information Compression, or Big Thing in the Chinese Room

What is knowledge? What does it mean for something to understand?

Searle once posed the now-famous “Chinese room” thought experiment. In it, a non-Chinese speaker is equipped with a huge dictionary; one of every possible Chinese phrase and the appropriate corresponding response. With it, the non-speaker can communicate can communicate as fluidly as a native with someone outside the room (using slips of paper), despite not actually knowing Chinese.

The idea here is that although the “black box” (as viewed from the outside) of the Chinese room seems to know Chinese, passing ever possible native-speaker test that you could think to administer. Yet it does not have any knowledge proper, at least in any definition of the word knowledge that we are used to. In fact, the intelligence of the non-Chinese speaker is not even used in the room; they only look up pages and copy down characters. So let's dispense with the middleman and replace them with a robot that looks up pages in the book and writes down characters. From this lens, the Chinese room clearly does not *actually* understand anything; it’s just a big book, a mechanical lookup machine, and a copying machine! If you wrote the Chinese room as code, assuming you have digitized the book, the entire behavior of it (a simple lookup table) could be fully encapsulated within a few lines.

We now have a problem. Between the lookup table, and the native speaker, one understands Chinese and one does not. Yet there is no test we could administer that could differentiate them. When it comes to linguistic behavior, they are identical. This is Searle's response to the Turing test; our conclusion being that despite two things looking and behaving the same, one could be said to understand, and one could not.

Clearly, something is wrong. Consider first (irrelevant though it may seem) the logistical challenges of building what we will call the "lookup table" model of the Chinese room. Let's assume first that the most the outside communicator will write to the lookup table is about 50 characters of coherent Chinese. Chinese has a natural language entropy of ~3.3 bits per character, meaning the average character in Chinese has around $2^{3.3} \approx 10$ "valid" following characters. Therefore, a rough approximate of the number of coherent Chinese phrases under 30 characters is:

$$\sum_{i=1}^{50} 3.3^i \approx 1.2 \times 10^{26}$$

Assuming you can write around 4 characters in a square centimeter, and that in a meter-high stack of thin paper, you can put (fun fact!) around 20,000 sheets, a "universal sub-50 length Chinese dictionary" would have to be around a 60 quintillion meters tall! That's around 30 times the distance from us to Alpha Centauri, the nearest star system to us - a suspiciously big room for a reasonably-sized Chinese speaker. And yet, a native human speaker manages to fit all of it in a few squishy handfuls of grey matter. Conservative (and very, *very* rough [1]) estimates for the amount of memory needed to simulate the human brain give a number of around 10 terabytes ($\sim 10^{11}$ bytes), and we can sustain conversations way past the context length of 50 chars. Yet the dictionary takes around a factor of 10^{15} more memory.

Searle goes on to posit that no matter how complex an algorithm gets, at no point would it gain what we call "intelligence." Since all algorithms are just configurations of the same "lookup-style" logical building blocks that made up the Chinese room that clearly did not *actually* understand Chinese, we can extrapolate to say that at no point in its development will *any* program learn to understand anything, despite appearing to do so at first glance! Furthermore, it is difficult to say that the machine possesses knowledge, not having any sort of "belief system" that they can use to interpret the Chinese. It does not *mean* anything in its reply. Therefore, something must be deeply different between us humans and any computer whatsoever, in order so that understanding may be placed into the gap. Unfortunately, science has not given us any indication that this is the case, at least in terms of computation; as far as we know, we can simulate any physical system (including the brain) to an arbitrarily large degree of precision as long as we have the time and compute to run the process. [2]

So what is it that makes us different? Well, have a conversation with a regular human being, then just gaze at the stack of paper stretching across the galaxy! It *seems* so qualitatively different that it's hard to imagine there not being any "true" difference - that is, a difference of epistemological significance, such that we can call one "knowing" and one not so. But this is the point of the Chinese room; to elucidate whether such a difference exists, and if so, where it is.

The answer to the first question, I think, is a clear and resounding yes. First, it is clear that the Chinese-speaking human and the astronomically-large stack of paper accomplish the same goal. This goal, stated more quantitatively, is to approximate a function, mapping from the set of valid input strings to the set of valid outputs. The difference between the human and the lookup table lies in to what degree they are able to compress this information. One packs the same amount of "valid information" in around 1 quadrillionth of the space! This is possible because human brains have learned underlying patterns in language that allow us to store a large amount of information in what are essentially "parallel" or "blanket" rules - that is, rules that allow us to unify different aspects of language with relatively little *total* information stored. For instance, since we have useful tools like syntax and semantics, (which are essentially loose "rules" we use to parse language), we can understand that "waffles and orange juice" is a valid answer to all of "What did you have for breakfast today?", and "What did you chow down on this morning?" and "What are your cooked carbohydrate and beverage of choice?", whereas an answer like "Volkswagen Beetle" is not. These rules are the underlying patterns that allow us to *compress* our knowledge into something that can fit inside our skulls rather than span our stellar neighborhood.

What we need is a better metric with which to understand the idea of knowledge. I claim that these rules - the degree to which you are able to compress information about a dataset - is the clear and indeed quite possibly the *only* difference between understanding and non-understanding. There is a fundamental difference between true knowledge and rote memorization. This idea is very intimately related to some fundamental facts about the process of induction and information theory. When we, as humans, "see patterns", what we are really doing is generating a compressed rule that fits past data. The next time we need to predict what will happen given a set of inputs, we can often make an accurate prediction by relying on our rule. Thus, information compression is also the fundamental tool that allows us to *generalize* to unseen data. This is what most obviously differentiates us from a raw-data-storage-based lookup table. When faced with a phrase, perhaps 51 characters long, or something not in the lookup table, the Chinese room machine fails immediately. It has thus far relied on the equivalent of memorization, meaning it cannot generalize to the unseen. Yet when we can compress information, we approximate the likely "true function" that generates our data. This brings with it the power of generalization, and it is why we, as knowledgeable humans, can create linguistically coherent answers to questions that we have never heard before.

It turns out that this is also the underlying problem in machine learning; finding the simplest function that still fits the data. In the context of machine learning, over-complicating a function is called "overfitting." It essentially amounts to a memorization of the training data, meaning you lose any relevant generalization capability. This is bad; it both takes up more space (the function becomes more complicated [3]) and is extremely inaccurate (high variance, since the function depends exclusively on the particular dataset provided). To combat rote memorization, ML engineers use a particular tool called a "regularization term" that essentially measures how simple a function is, and optimizes for it as to avoid overfitting. The proof for why such a thing is good comes from information theory. [4] In this sense, machine learning is not just a catchphrase for an applied branch of statistics that deals with fitting models to data. It is real, genuine, machine *learning*.

So, what is knowledge? The thing that differentiates something that understands and something that does not is simply a matter of information compression. Or, stated differently, a system that is more knowledgeable is one that can encapsulate the behavior of data using a simpler function. Yet it might be wise to be more explicit. We say that a function has more "knowledge" of the data if a) it fits existing data and b) it can be encoded in less space.

This model of knowledge also has some very real-world analogies, even outside of machine learning. Suppose, for example, that you want to choose how to design an education system. One teaches kids physics through rote formulaic memorization, i.e. if you see an problem in such-and-such a form you should put this number here and this number here and compute, if you see a problem in another form you should put this number here... etc, etc. The other mode of education decides to teach kids the laws of physics from the ground-up, allowing them to derive the needed computations by seeing patterns and observing rules in the systems themselves. Most people would say that the second education method is better, and now they have justification; kids that learn patterns, can *synthesize knowledge* (i.e. are able to compress information), have truly *better knowledge* than those that merely memorize. This is a fact not captured by JTB. This means: a) They will be able to generalize to unseen patterns, meaning that their knowledge has more *scope*. b) As a result of the compression of information, their knowledge has a drastically reduced mental load (think about how difficult it is to remember a list of formulas outright! you need to encode their patterns to learn them). This means that they will retain their knowledge for longer. This model of computation is what we colloquially mean by "understanding" - the second child understands more than the first child, despite them performing indistinguishably on the test.

This is what the question "*why?*", a favorite among children, is really asking. It's asking whether you can unify some aspect of the phenomenon in question in order to fit in with what we already observe about the world - that is, to be able to derive the new data (to compress more information) using existing rules. For instance, an knowledgeable answer to the question "why do apples fall" might be something like "Because gravity tends to pull objects (not just apples!) towards each other, so the earth and the apple are pulled towards one another." This is a satisfying answer since it not only applies to apples; rather, it is an underlying principle that the knower can use to answer plenty of other questions; "why do rocks fall?", for instance. Not only that, it allows us to make useful predictions; that, for example, this pineapple will fall too when dropped. The opposite would be an answer like "because it does." This is the equivalent of memorization, or the "lookup-table" approach; it tells you only about apples. That is, in order to learn whether or not pineapples fall, you would have to ask again, and for rocks yet again. The compression factor is zero in this case.

True learning is not equal, as many people would think, to knowing more things. In fact, it is quite the opposite - the person that knows one thing (that gravity pulls things towards each other) knows more than the person that knows many (apples fall, pineapples fall, rocks fall). [5] [6]

Some alarming corollaries emerge that I'd like to address. The first is the main point of the Chinese room; that the same function can be encoded in radically different ways while displaying identical behavior. The only differentiation factor we have is the size of the function (how much space it takes to encode) as well as (if you are of a computational bent) how much time it takes to compute. [7] Further, this implies that there is no such thing as "knowledge proper." The "generating function", or "true function", of data can be described by an infinite number of generation methods. It is not clear that one is more "valid," or "true" than the other. In such an absence of direction - given that each contender has an equal claim to the data - we are forced to find a metric that allows us to likely good approximations from likely bad ones. The best we can do is to say that more knowledge is encoded in less complex functions, since with compression comes generalization ability. Another corollary is that the knowledge metric is defined on a set of existing data. This means that the "more knowledgeable" function can actually be proven wrong if new data is acquired. So the knowledgeability of a function is rather a measure of how well you can do given what you have; it does not have any claim to correctness. Yet because of Solomonoff's compression-generalization idea, it is still a better option (assuming that the data is all you can work with) than any other alternative generating function. Thus, knowledge does not *have* to be "true," since any good model has a possibility of being false. The best we can do is a high probability of truth.

Philosophers like to come up with fun acronyms like JTB - the Justified, True Belief model of knowledge - that play very prominent roles in philosophical discourse. This makes me frustrated. The definition of knowledge explored in this sketch is more fundamental than "Justified, True Belief." In fact, it does not even require that the fact (short for "observed nature of true generating function") is true; a word philosophers are remarkably difficult at defining. Lt's cross that bridge when we get to it. It is also not anthropocentric, in stark contrast to JTB. [8] To know something, according to JTB, requires an entity (largely assumed to be human) that has the capacity to "believe" and "justify." What those words mean is also not so clear.

Another problem with the JTB theory of knowledge - and existing conceptions of knowledge in general - is that it is possible to create knowledge out of thin air. Suppose I wanted to generate knowledge about written words. For each new word that I wrote, I could generate a new "piece" of knowledge by saying; hey, this word is "cat"! It is justified, true, and believed. Using this method, I could obtain infinite pieces of knowledge by writing some benign word, like "cat", over and over ad infinitum. Does this mean I have infinite knowledge? I would argue that the answer is no. All I've done is generate more *information*. This is a vital distinction to make. While writing cat again and again made new information, it did not add a substantial amount of new knowledge, since the encoding function for "cat" ten times is about the same length as the encoding function for "cat" 1000 times. You can just replace a number in your for-loop. This is a much better representation of what we usually mean by "knowledge" - I *know* that I wrote cat 1000 times, but I did not gain much for every single time that I wrote the word "cat". The difference between information and knowledge is a matter of compression. One merely exists, and the other *generates*.

Another implication of this model of knowledge is that knowledge is a product of functions, not humans. The reason we say that humans possess knowledge is because they have the tools they need to observe and approximate functions. In other words, the have the capacity to learn. But ChatGPT is able to pass the Chinese room test easily. It is trained on the entire internet, but it doesn't just memorize it; it learns to approximate the same function that we learn when we develop the capacity to speak. To be able to compress the unimaginably huge space between possible questions and possible responses - our minds have a difficult time with envisioning exponentiality: just think of the size of that stack of paper! - into a space that can fit inside a modern datacenter is already a miracle. It is even more of a miracle that the same amount of coherency is can be contained within the few quarts of grey matter right behind your eyes. You must know some rules about language (and, by proxy, the world in general) to be able to compress so much into so little space. It is not yet clear whether the rules that ChatGPT relies on are the same ones that we do, but the information-compression theory of knowledge says that there is nothing special about us. If the heuristic for knowing something is to what degree you can compress the information present in data, if a new GPT model came out that approximated English in less space than it takes the human mind (which is, theory aside, very hard to verify), then it would, in an odd way, "know" English better than humans do. We may be nearing this point already, where language models can essentially simulate "Chinese room" behavior in even less space than a human mind.

Alan Turing was one of the earliest people to think about simulated behavior. He created a new metric, the Turing Test, in order to determine whether a machine should be treated as intelligent. The idea was that if behavior was identical between two systems, a human and a machine, that they should have equal credence applied to their claims of understanding. In light of the Chinese room, the Turing Test (which is essentially what Searle applies) falls very radically short. The reason it still remains relevant is only because we cannot write lookup tables a quintillion quintillion characters long. Theoretically, it is trivial to pass the Turing Test without understanding.[9] But we are getting close to something much more impressive (and in fact, have already done in some weak sense[10]); passing the Turing Test using an amount of resources that is non-intractable. To this end, I propose a new Turing Test; one which not only takes into account whether or not behaviors are indistinguishable (which is easy, at least in theory), but also how much space it takes to run each program. This way, a machine that passes the Turing Test has a very genuine claim to understanding; it has the same generalization capacity, is as efficient, and would understand language maybe even better than we do. [11]

---

1. This value is very difficult to estimate, given how different from discrete binary computation our brains are. Take this number with a grain of salt - it could be a few orders of magnitude more or less. This is the best number I could get from the internet. ↩

2. There is a separate problem, the so-called "hard problem of consciousness", that Searle aims to elucidate here. For now, I will keep this bag of worms safely closed. ↩
3. An assumption is that you are using a "reasonable model of computation." For more on this, read up on Komolgorov Complexity. ↩
4. As it turns out, the simpler a function is (assuming some facts about the distribution of "generating functions") the more likely it is to be a good predictor for future data. For more on this, explore Solomonoff Induction. It's a joy that there is a robust mathematical underpinning to this statement (though it does not come without assumptions of its own). ↩
5. Some would be tempted to say that this is not the only difference between these two explanations; one clearly gives a *reason* to believe it, and one only states the facts outright. In my mind, *reasons* - that is, statements about cause and effect - are merely shorthands for the highly-compressed "generator" of the data. For instance, "gravity exists" is really a shorthand for "for all things with mass, they tend to fall towards one another in such-and-such a way." And that's why apples fall; since they have mass, they tend to fall towards other massive things. This is merely a "compressed rule" that we can use to encapsulate much information. ↩
6. This is not always strictly true. For example, knowing the laws of physics does not allow you to predict that the declaration of independence was signed in 1776. I plan to write more about this later, but this finding arises from the notion of unpredictability in *emergent behavior* - that merely knowing the rules of a system does not necessarily allow you to know the state of it. ↩
7. For now, we will ignore the time-complexity aspect. This is an ongoing area of research for me. There are perhaps connections to complexity theory and computational irreducibility. The quick answer is that you probably also have to take into account time if you don't want a horribly inefficient generating function for your data. This would be the opposite of the instant lookup table; a machine that is very small, yet takes a decade to answer any question. ↩
8. Defining knowledge as JTB is kind of like defining a computer as ICM; as something that we Interact with, can Calculate with, and is made of Metal. Yes, that's what computer usually means in day-to-day language, but there is a much more abstract notion of computation that Turing formulated back in the 1930s and 40s, one that is much more philosophically significant. This, in some ways, is the "true" notion of a computer, independent from the specific macintoshes and user interfaces that now populate our collective notions of one. This more abstract notion is what I hope to elucidate when it comes to knowledge. ↩
9. There are more problems with the Turing Test; one is that a human is used as the differentiator, to determine whether given behavior comes from a machine or a human. But humans are famously easy to deceive. [10-1] Perhaps a stronger version would be to ask if *any* algorithm existed (within certain size constraints) that could reliably distinguish their behaviors. In this case, I suspect optimal performance would resemble human neural architecture in closer ways. I will write more about this later, but the basic idea is that evolution is a pretty good (?) optimizer [12], and that it is also subject to space constraints (bigger brain, more energy, more starving chance). ↩
10. See ELIZA, and note that ChatGPT can already convincingly pass as human. ↩ ↩
11. A good question: don't you have to be able to reason or have some sort of epistemic contact with reality to really know something? That is, don't you need to be able to "know" what you are referencing in the real world? My answer: an "epistemic contact with reality" is part of our compression algorithm that allows us to reason with abstract ideas and equate them to one another. This greatly increases the compression factor of our language function. Usually, reasoning tends to have an anthropocentric connotation. Yet if this ability to reason, to grapple with abstract concepts and link them into coherent output is an efficient way of compressing language (and I believe it is), I have no reason to believe that a similarly-efficient algorithmic model would not adapt the same strategy. Under my definition of understanding, "human-style" reasoning is not a requisite to understanding, but if this style of reasoning is the most efficient strategy, to "understand" language as well as a human does would imply that they would adopt reasoning skills anyway. But to learn the same patterns that we do, we might need to have a model trained on multimodal data (the same way we are!), so it could learn to associate certain visual patterns with linguistic ones, etc etc. ↩
12. Is this a footnote to a footnote? Yes. Anyway, I disagree here; evolution is a pretty naive form of stochastic gradient descent. Perhaps the optimal machine would be radically different, although searching for minima is known to be a very hard problem computationally, so it remains to be seen whether or not it would actually be feasible to find a kind of architecture that was better than that of

the brain. Also, to even have hope of converging on the human brain through mere Turing testing, you would probably have to design a more sophisticated Turing Test; one that not only measured language capability, but also very human capabilities like navigation, spatial awareness, body language, etc. But this is not as feasible. ↩