Does ChatGTP Think?

By

Robert F. Hink

Introduction:

Perhaps the most valuable artifacts from AI labs are LLMs (Large Language Models). To a large extent, these models have taught machines to communicate with humans so convincingly that they seem to be alive. Obviously, they are not since they fail to meet our qualifications for life: homeostasis, organized structure, metabolism, growth, adaptation, sensation, and reproduction (Life).

But do these artifacts think as humans do? I believe many of us who have worked with ChatGPT or other LLMs have asked ourselves that very question.

As a Neuroscience researcher and a long-time AI enthusiast, I may have a different perspective than most. In this article, I attempted to answer the question.

I take up The History, The Differences, The Similarities, The Critiques, The Conclusions.

The History:

In 1958, Frank Rosenblatt reported that by using elements modeled after human-like neurons as described by McCulloch–Pitts, he could train a layer composed of these “neurons” to distinguish visual stimuli, such as letters. The layer was called a perceptron. The perceptron was perhaps the earliest example of machine learning (ML).

In 1969, Marvin Minsky and Seymour Pappert published their famous book "Perceptron" showing that the perceptron could not model the exclusive-or operation (XOR). The reason is that the perceptron could only distinguish linearly separable states; XOR is not linearly separable.

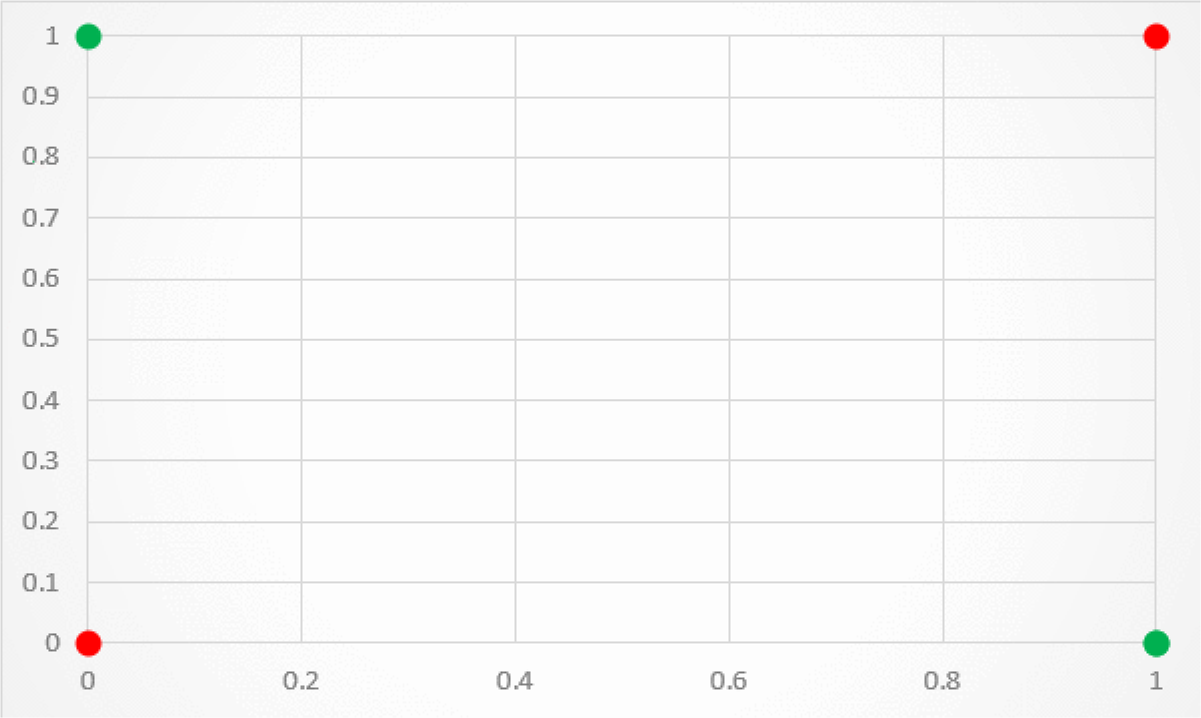


Figure 1: XOR (green = True; red = False)

Figure 1 maps the truth values for XOR. Any line drawn cannot separate the green dots (True) from the red dots (False).

Rosenblatt, Minsky, and Pappert all knew that multi-layered perceptron could model XOR and said so publicly, but the damage had been done. Funding for AI research dried up. That was the first of several "AI winters".

In 1964, Joseph Weizenbaum of M.I.T. developed a pattern matching program that simulated the questioning-style of a Rogerian psychoanalyst. He called it Eliza.

Weizenbaum shares the anecdote of a research associate working on the program and knowing Eliza was just a program asked Weizenbaum to leave the lab. Presumably, she was sharing private, personal information that she wanted to remain confidential.

This was one of the earliest examples of a chatbot.

In 1986, Michael Jorden (not the NBA star) proposed a neural network that fed back on itself, recurrent neural network (RNN). He discovered that RNNs could generalize from learned patterns so that the path or trajectory the network had learned would attract a new path started from unknow data. He even called these learned patterns attractors borrowing the term from Chaos Theory.

In 1990, Jeffrey Elman published a paper that showed how RNNs could discern patterns in natural language, the Elman RNN. He trained his neural network samples of language. The RNN was taught to predict the next letter in a sentence of words; he ignored spaces between words. The error rate was high between words but low within words indicating that RNN was learning to spell.

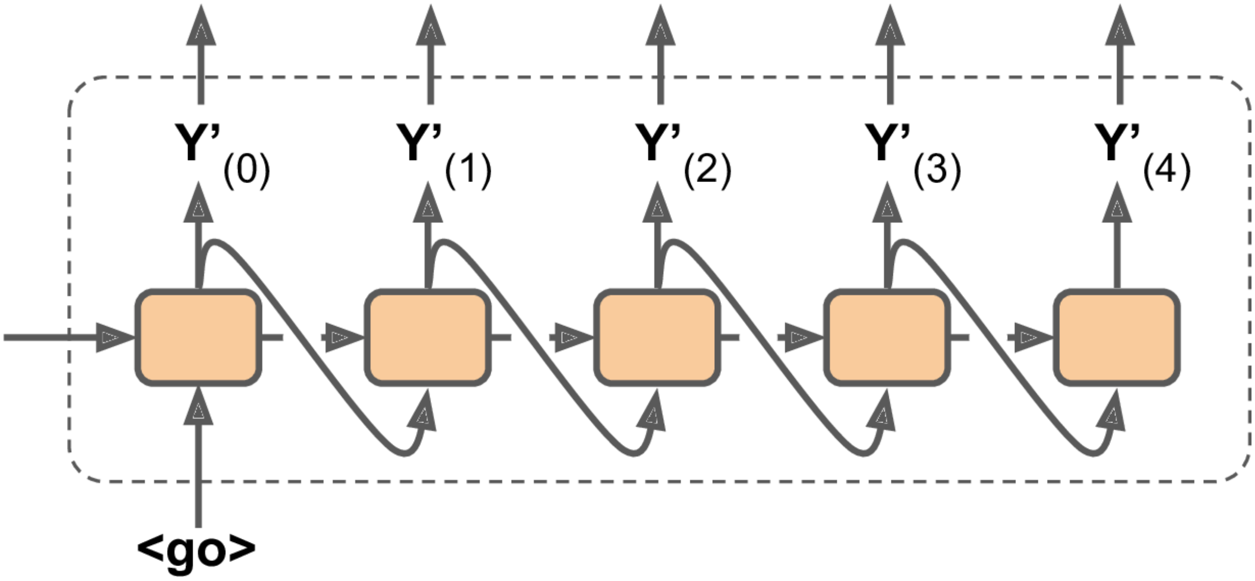


Figure 2: Recurrent Neural Network

He also found that the sequence patterns constituting words would form clusters and there would be clusters within clusters. For example, nouns would cluster into animate and inanimate clusters; the animate cluster would further group into human and non-human and so forth. This finding showed that the RNN was learning semantic relationships, as well.

In 2011, a group of researchers (I. Sutskever, J. Mertens, and G. Hinton) from the University of Toronto showed that a larger RNN models could perform well at next-letter prediction up to a point. After a few words, it would drift off topic and begin to babble nonsensically. This finding showed that scaling up could improve performance. This model is generally recognized as the first LLM.

They suggested that the breakdown in performance was due to the capacity of the model to access information beyond its limited range. They also speculated that by packing more information into a smaller space (i.e., compression), the RNN could show more human-like behavior. They even likened compression to the "equivalent of intelligence". I should mention that compression is roughly the same as what psychologists would call attention in the sense of filtering out irrelevancies.

In 2015, Andrej Karpathy reported in his blog post that he was able to train a RNN on a small sample of Shakespeare. When tested, the model produced novel, though Shakespeare-like output. He got a similar result with algebraic expressions.

He went further by explaining that the model learned what was relevant on its own. Aside from setting up the model architecture involving only a few lines of code, no programming was required.

In 2017, A. Radford, J. Jozefowicz, and I. Sutskever using a large LSTM (Long Short-Term Memory) (4,096 units) trained on 82 million Amazon reviews. A LSTM is like an RNN which trains special gating units that control the flow of information through the network. RNNs in contrast, pass all information forward and backward through the network.

They discovered that different units become sensitive to different aspects of the review. For example, they discovered that one unit would fire if the review was positive or not if it were negative.

They further used their model to generate reviews from scratch. By controlling the "sentiment" unit, they could control the sentiment of the generated review.

They noted that their model was constrained by the amount of the state information about the input sequence. It applies to both RNNs and LSTMs since they are both sequential models of state information. This finding is the same that Sutskever, et al., 2011 discussed earlier.

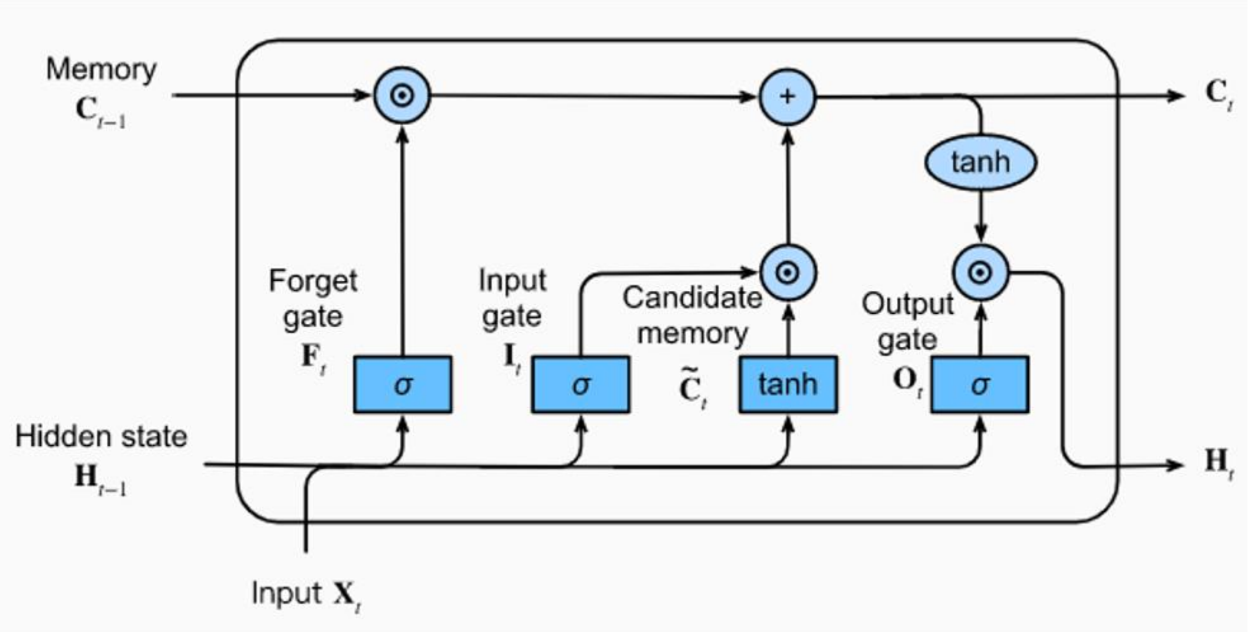


Figure 3: Long Short-Term Memory (LSTM)

In 2017, a team from Google lead by Ashish Vaswani wrote a groundbreaking paper called “Attention is All You Need”. In this paper, they describe how an encoder-decoder transformer could overcome the limitations of sequential models. Their model could process a long segment of text simultaneously through a process called self-attention.

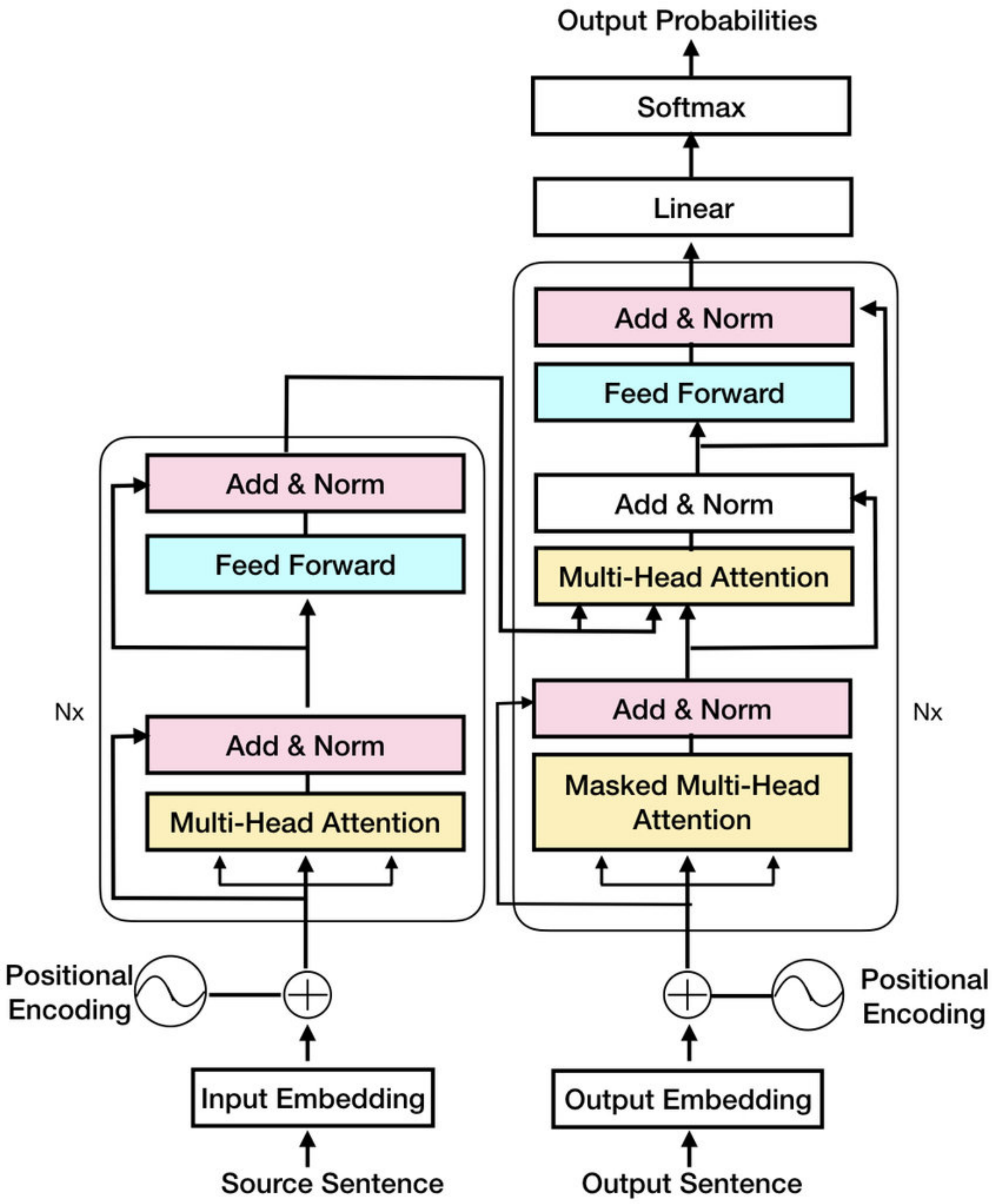


Figure 4: Encoder Decoder Transformer

The process may seem a bit daunting so I will break it down (see Figure 4: Encoder Decoder Transformer):

1. **Embeddings:** This attention process takes tokens (i.e. words control markers such as [CLS] (CLear Screen meaning start anew) or [SEP] (Separate), punction, and word parts such as “ing”) from the initial prompt and decomposes it into a list of tokens. This step is called tokenization.

   The tokenized text then is converted to a matrix of embeddings. Each row in the matrix is the embedding vector for the corresponding token.

   The embeddings are magical in that they capture the properties of the token and where the token resides in a multi-dimensional concept space. They are learned by reading massive amounts of text. The magic comes from the fact that one may perform arithmetic operations on the embedding vectors. For example, consider the following:

   embedding(“king”) – embedding(“man”) =
   embedding(“regent”)

   embedding(“regent”) + embedding(“woman”) =

   embedding(“queen”).

   The reason this works is because the embeddings are trained to do so. Each column of the embedding corresponding to a given token represents a different property. So, a king is a regent with high maleness. Remove the maleness captured in the token “man” and one is left with the embedding for “regent”.

   There is one further fine point. What about positional information? That is relevant since some tokens fit better into certain positions relative to other tokens. In the 2017 study, the authors used a sine-cosine function value assigned to each embedding. With that, the transformer can look for tokens that are so many “wavelengths” from another token. They also tried learned positions which gave similar results.

2. **Attention:** The “distances” between embeddings in concept space are calculated by dot product matrix multiplication: $Q * K^T$, where Q is the Query matrix containing information about a given token and K^T is the Key matrix with columns and rows interchanged. K has information about other tokens. The product matrix holds the “distances” between the various tokens in concept space.

   The distances are run through a function that scales them to fall between 0 and 1. The function is called Softmax; it is defined as $1/(1 + e^{-x})$. The Softmax transformation converts the variable X to a value between 0 and 1, like a probability relating distance to importance. Softmax also is important due to the tendency of neural networks to become unstable when their gradients of weights explode; Softmax keeps the values within range.

   Lastly, the Softmaxified matrix is multiplied with a third matrix, V, containing relevant information about the prospective next token such as semantic information, part of speech, sentiment and connotation, information specific to the token, contextual importance, and associations or relations to other tokens.

This long-winded explanation focuses on the Q, K, and V matrices. These matrices carry the information that helps ChatGPT pick the right word. I will let Chat GPT 4 summarize:

> 1. **Query meets Key:** Every word (via its Query projection) asks around, "Hey, who should I pay attention to?" by scoring how well it matches with every other word (through the Key projections). This is like asking, "Who's my dance partner?"
> 2. **Scoring:** These scores are then turned into probabilities using a SoftMax function, kind of like deciding how much attention to give to each potential dance partner based on how well you sync up.
> 3. **Value steps in:** Once we know to whom to pay attention, the Value matrix brings in the content of each token that is deemed important. This is the step where the actual information exchange happens, akin to deciding what dance moves to perform with your partner based on how much you're into them.

> In simpler terms, if the Query and Key are deciding the "who" in "who to listen to," the Value is about the "what" as in "what will I gain from listening to them." It's the essence of the message that gets amplified in the self-attention process and passed along to the next layers of the network, making sure that the most relevant information is what makes it through the noise of the dance floor.

3. **Add & Norm:** This process ensures that the that the values of the matrices stay close to the original inputs. These values are normalized yet again to keep the values within bounds.
4. **Feed Forward Network (FFN):** The FFN orchestrates or mixes the Q, K, and V matrices so that certain tokens stand out. It first does a linear transformation that highlights differences. It then performs a non-linear transformation that adds or removes dimensions. Finally, it reshapes the matrices to match their original form.

   The output of the FFN is passed through another Add & Norm layer to ensure that the values do not drift too far away from the originals.

For each token in the prompt, a Q, K, and V matrix is constructed. Each head learns to focus on different aspects such as emotional content, cause and effect, or context of the scene. This separation-by-token allows the transformer to analyze the problem of finding the next token from different perspectives concurrently. It also permits the computation to be parallelized, greatly enhancing throughput.

The above describes the encoder; the decoder is much the same except it may use different embeddings. For language translation, these embeddings are from the target language.

In the 2017 model, the Q_{enc}, K_{enc}, and V_{enc} from the encoder are passed into a layer of multi-headed attention in the decoder.

There, they are added to those following at least one layer of multi-headed attention operating on the decoder embeddings, Q_{dec}, K_{dec}, and V_{dec}.

The network layers are wider, and the networks are shallower than with sequential models. As a bonus, the networks were easier and faster to train than with sequential models. Also, the roadblock of limited capacity of holding sequential state information had been removed.

This attention approach freed NLP researchers to train massive models. Chat GPT 4 has around 175 billion parameters. To run the 30,000 GPUs that power it produces about 8.2 tons of CO_2 each year.

This history review shows that LLMs were on the way to conquer the complexity of natural language. What enabled LLMs to be developed was the internet that made the required training data available and GPUs, created for gamers, capable of performing parallel processing.

There was another takeaway from this research, the subtle structure of natural language **emerges** as a byproduct of scaling up. That structure is crucial in predicting either the next token, in the case of ChatGPT, or the next thought, in the case of humans. Emergence is when properties of a complex system are apparent only for the whole, not the constituent parts. Michael Jordan's (1986) mention of the tendency of his RNN to gravitate toward what was learned from what was novel. That tendency is emergent.

There is another emergent property found in ChatGTP. It is called In-Context-Learning (ICL) as opposed to in-weight-learning which refers to the learning done during training. With ICL, the model weights and biases are frozen by disabling backpropagation which updates model those parameters. I think of ICL as reasoning by analogy. Here are some examples of few-shot-prompting using a made-up word "hakumoto" and few-shot learning:

> Prompt: "A hakumoto is a Japanese musical instrument. Use 'hakumoto' in a sentence."
>
> Response: "After a long day of deciphering the enigmatic algorithms of AI, Bob found solace in the gentle, yet uncharted melodies of his hakumoto, an instrument as enigmatic as the code he so passionately unraveled."

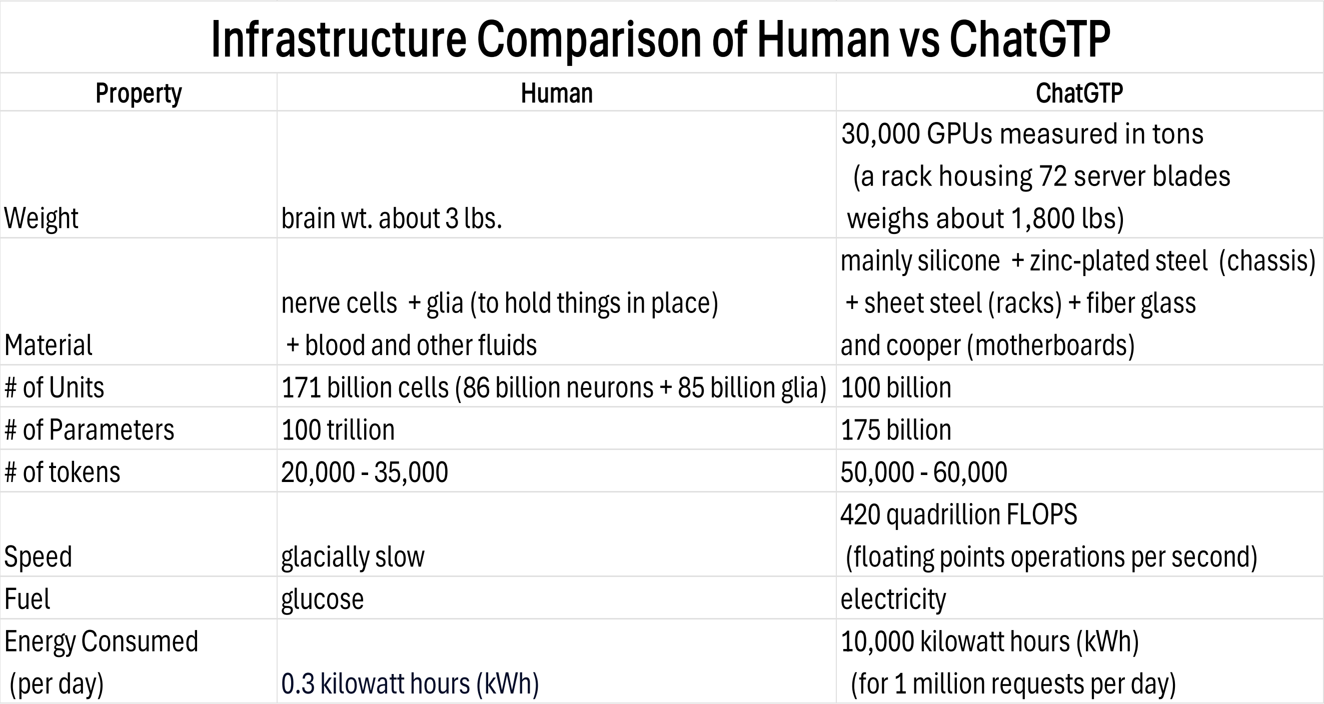
ChatGTP knows about Japanese musical instruments, perhaps a shakuhachi (bamboo flute), upon which it inferred things about the non-existent "hakumoto".

Another emergent property shared by both neural networks and the human brain is neuroplasticty. It is a feature of highly connected systems that can adapt or relearn function. Neurological studies and neuro network ablation experiments demonstrate plasticity. It suggests that function either is not localized or that other units/tissue can learn to assume the lost function.

Watching ChatGTP at work seems instructive in explaining how the human brain works "by analogy". It also says a lot about human natural language.

The Differences:

The question posed at the onset of this article is whether ChatGTP think like humans. The behavioral similarities are striking, but the infrastructure seems quite different.

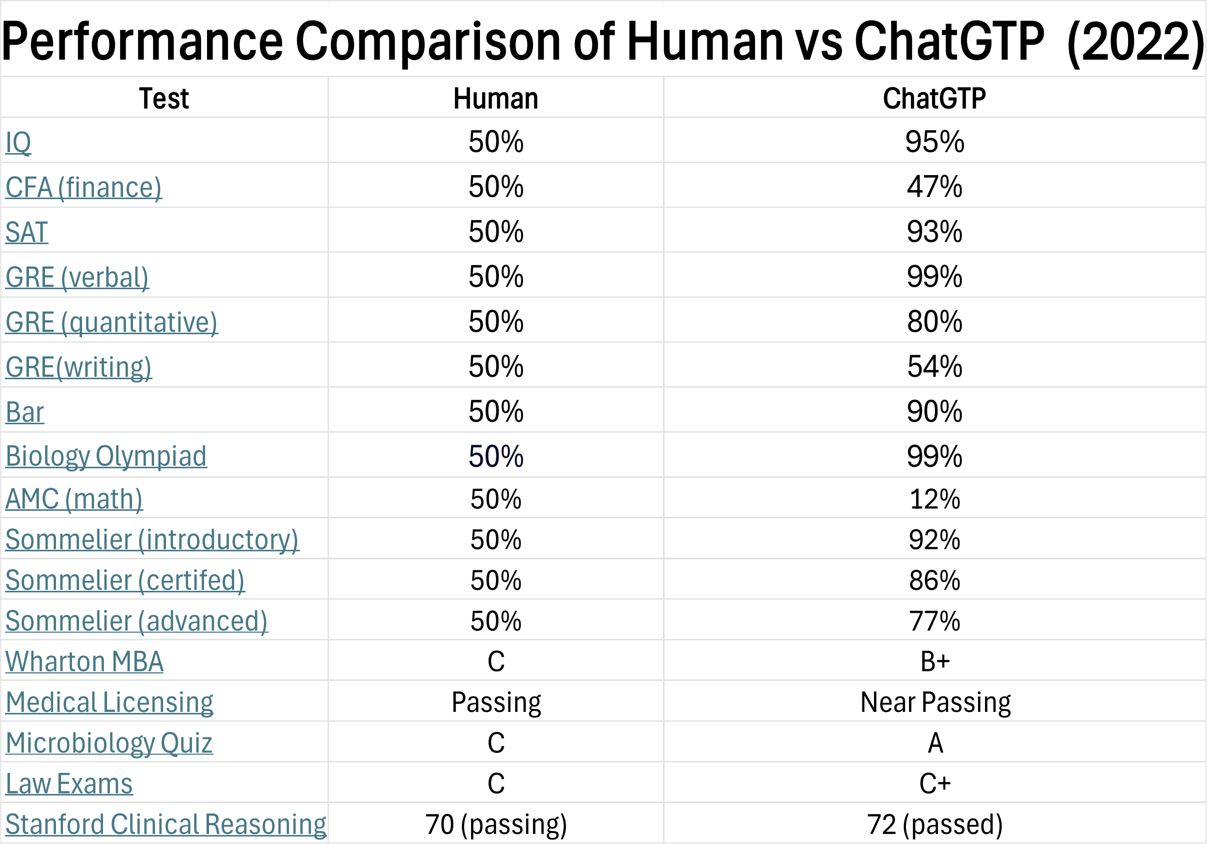


Infrastructure Comparison of Human vs ChatGTP

Property	Human	ChatGTP
Weight	brain wt. about 3 lbs.	30,000 GPUs measured in tons (a rack housing 72 server blades weighs about 1,800 lbs)
Material	nerve cells + glia (to hold things in place) + blood and other fluids	mainly silicone + zinc-plated steel (chassis) + sheet steel (racks) + fiber glass and cooper (motherboards)
# of Units	171 billion cells (86 billion neurons + 85 billion glia)	100 billion
# of Parameters	100 trillion	175 billion
# of tokens	20,000 - 35,000	50,000 - 60,000
Speed	glacially slow	420 quadrillion FLOPS (floating points operations per second)
Fuel	glucose	electricity
Energy Consumed (per day)	0.3 kilowatt hours (kWh)	10,000 kilowatt hours (kWh) (for 1 million requests per day)

Figure 5: Infrastructure Comparison

The table above shows that the human brain is “light weight” compared to ChatGTP in many respects: weight, tokens, and speed. The brain exceeds ChatGTP in complexity (# of parameters) by 3 orders of magnitude! Also, humans are much more energy efficient at information processing than ChatGPT (0.3 kWh per day vs 10,000 kWh per day).



Performance Comparison of Human vs ChatGTP (2022)

Test	Human	ChatGTP
IQ	50%	95%
CFA (finance)	50%	47%
SAT	50%	93%
GRE (verbal)	50%	99%
GRE (quantitative)	50%	80%
GRE(writing)	50%	54%
Bar	50%	90%
Biology Olympiad	50%	99%
AMC (math)	50%	12%
Sommelier (introductory)	50%	92%
Sommelier (certifed)	50%	86%
Sommelier (advanced)	50%	77%
Wharton MBA	C	B+
Medical Licensing	Passing	Near Passing
Microbiology Quiz	C	A
Law Exams	C	C+
Stanford Clinical Reasoning	70 (passing)	72 (passed)

Figure 6: Test Performance Comparison

ChatGTP had mixed results in 2022. More current results have been published. The performance has improved but remains mixed.

In 2023, ChatGTP 4 passed a high school exam with a score of 8.3 out of 10; ChatGTP 3.5 scored 7.3 on the same test a year earlier. Humans achieved a mean grade of 6.99.

Also in 2023, a group of researchers from Poland tested both ChatGTP 3.5 and ChatGTP 4 on a dermatology exam. Both performed better overall in English than in Polish by an average of 8 percentage points. ChatGTP 4 score 80% (English) and 70% (Polish) where 60% is considered passing.

These comparisons show convincingly that humans and ChatGTP differ dramatically in how they process natural language. Humans seem to move from thought to thought while ChatGTP goes token by token. Humans do not multiply massive matrices together to find a word. Even the best human calculators could not perform those calculations as fast as ChatGTP. Humans use an organ which runs on the power of a light bulb while ChatGTP requires 33,000 times that.

Even though how the human brain and ChatGTP may differ in how they process information, that does not mean that ChatGTP does not think. It only implies that ChatGTP does not think in the same way as humans.

There are basically two opinions about AIs "thinking". One view is that since conscious reflection is required for thought and ChatGTP is not self-aware, then it cannot think. I call this the nonconscious view.

The other position is that if ChatGTP acts as though it is thinking, then it is thinking. This is in keeping with the often uses expression, "If it looks like a duck and quacks like a duck, then it's a duck." This what I call the duck view.

This debate seems to be more about semantics than about substance. I take a more pragmatic view: if it is useful and has value, then use it. To me, ChatGTP is like having a friend who is a voracious reader, kind of like librarian. If you ask her a specific question, she likely will have helpful reference or background information, but for now, you will have to do your own thinking.

I asked ChatGTP 4 about whether it could think. Here is the exchange:

> Prompt: Do you think?
>
> ChatGPT
>
> **ChatGPT**
>
> As an AI, I don't think or have opinions in the human sense. My responses are generated based on patterns and information from the data I was trained on, aiming to provide helpful, informative, or entertaining content based on the context of your questions. If you have any more questions or need assistance with a topic, feel free to ask!

We seem to concur.

The Critiques:

In the exchange above, ChatGTP touched on an interesting point. It has no feelings, no desires, no emotions, and no preferences. If so, then it will never pass the Turing Test. This test was first proposed by Alan Turing, the person largely responsible for breaking the Enigma code during WWII; he called it the Imitation Game.

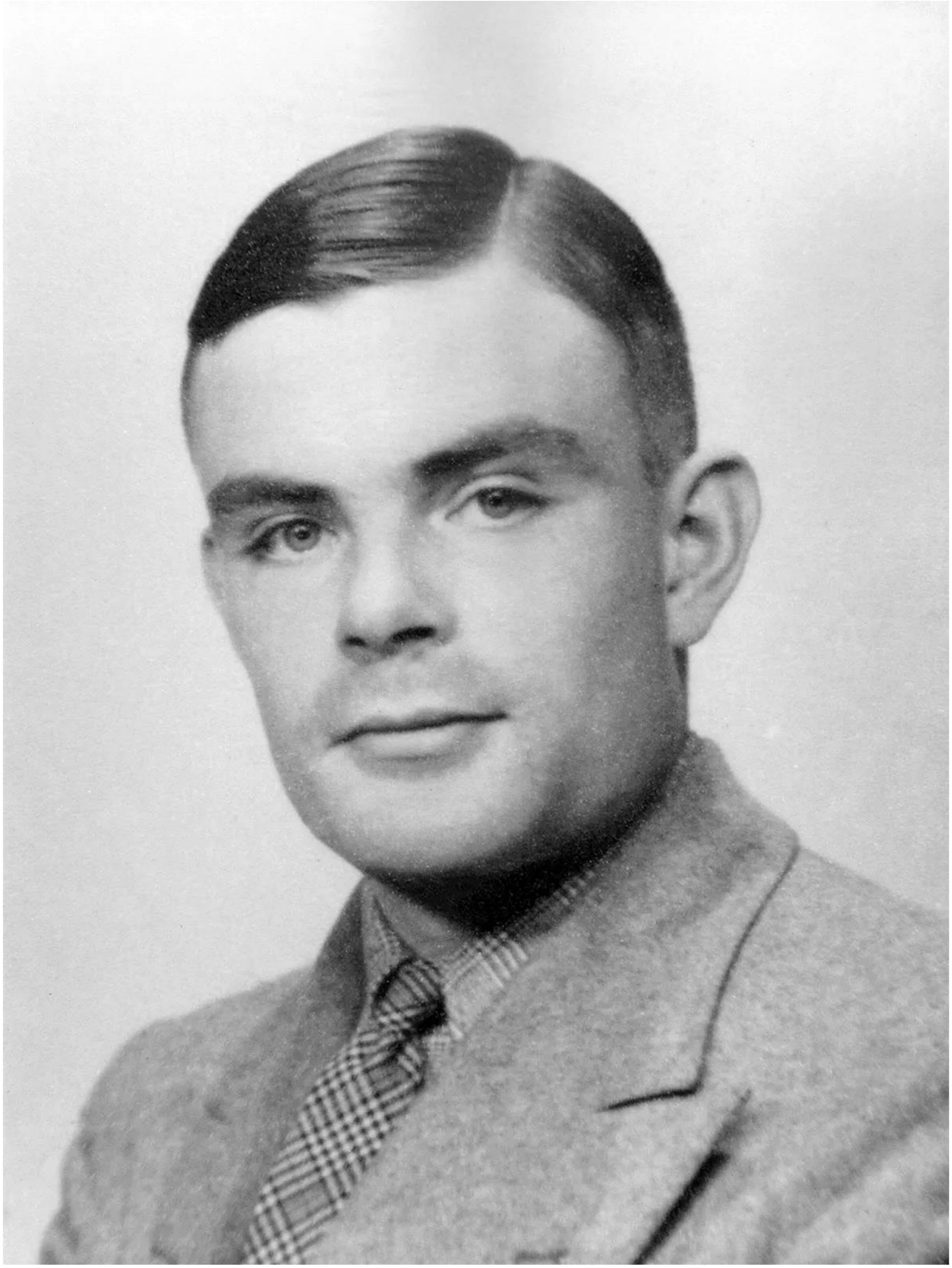


Figure 7: Alan Turing 1936

The idea is that if a human and an AI, either of whom may lie or deceive, are interrogated separately by a human interrogator who cannot reliably identify distinguish between the two, the AI passes the test. This test is the stated goal of Artificial Intelligence.

Figure 8: Turing Conducting the Turing Test

On the face of it, the test may never be passed since proving the null hypothesis statistically is exceedingly difficult. It likely would require a huge sample size. It would be like proving that a coin is true by flipping it a billion times.

Nevertheless, two philosophy professors at UC, Berkeley Hubert Dreyfus and John Searle criticized the entire premise of Artificial Intelligence that the Turing test could be passed.

Dreyfus based his arguments on 4 assumptions he claims are required for AI:

1. Biological – brain analogous to hardware
2. Psychological - mind analogous to software
3. Epistemological – all behavior may be mathematically formalized
4. Ontological – reality consists of clusters atomic facts

His argument is that humans cannot know their own behavior in the same way as that of other things. On that basis, he challenges the four assumptions.

Searle offers a thought experiment called the Chinese Room. In the Chinese room sits a person (i.e., Searle himself) with something like a Chinese-English Dictionary. A card with Chinese characters is passed in. Searle then uses his book of instructions to translate the message by following a step-by-step process and send out an appropriate response in English. The box appears to know Chinese and English when in fact Searle "doesn't know a word of Chinese". He would not need to know a word of English either; ChatGTP does not know either language, only embeddings.

Searle claims that since the computer lacks understanding and intentionality it cannot possibly be thinking as we understand it. This would violate the Strong AI hypothesis implicit in the Turing Test.

These critics, were they alive today, would possibly want to revise their views had they met ChatGTP. Nevertheless, ChatGTP does lack intentionality, emotions, and purpose. It admits as much.

Figure 9: "Searle" in his Chinese Room

The Conclusions:

So, does ChatGTP think? I don't "think" so. Does it matter? Again, I would say no. Should it think? That opens an entirely different area, a topic for another article.

The metaphor I use when considering the question of thinking is Einstein's formulation of the Theory of Special Relativity.

For about 200 years, physicists had embraced Isaac Newton's laws of motion put forth in his Naturalis Principia Mathematica. In Newton's view, space and time were static, like a stage upon which actors move. These laws seemed consistent with common sense. They also fit well with the observed motion of the planets.

In 1887, Michaelson and Morley showed that the speed of light appeared constant no matter how the measuring apparatus was moving. According to Newton, velocities should add together. So if someone threw a baseball toward the front of a car in a moving train, the speed of the ball to an observer in the station, $V' = V_{train} + V_{ball}$. This observation was a source of much concern among physicists of the time.

Einstein, instead of doubting the experiment, accepted the result. He imagined himself riding on a beam of light. This was an early example of his Gedanken experiments for which he became famous.

He envisioned a stationary clock behind him apparently ceasing to tick since we was moving away from it at the speed of light; light showing the tick would never catch up. As a consequence, space along his line of motion would collapse to zero in order to maintain a constant speed of light.

He further realized that as his motion slowed, his equations of motion would need to merge seamlessly into those of Newton's classical mechanics which they do.

Could ChatGTP have derived the Special Theory of Relativity, after all in 1905 this was a novel idea?

ChatGTP can use metaphors more as a means to relate to the prompter. I told ChatGTP that I liked to dance, so it made references to dancing when it explained mechanism of Attention. These metaphors are not imagined out of thin air as was Einstein's Gedanken experiment. Again, ChatGTP is more like a knowledgeable librarian than a brilliant, inspired, insightful theoretical physicist.

We should not be dismissive of ChatGTP's because of its limitations, however. It is valuable and powerful, and with power comes great risk. Fortunately, the NLP researchers and architects who created ChatGTP seem to be aware of the risk and are working to mitigate it.