From Alan Turing to GPT-3: The Evolution of Computer Speech | Otherwords https://www.youtube.com/watch?v=d2UccTPnl4w&t=25s Transcript: https://dontveter.com/ec/computerenglish.pdf

In the mid-1960s a computer scientist named Joseph Weisenbaum created one of the first natural language processing programs, called ELIZA.

The user would type conversational sentences and get responses in return--a system that today we might call a "chatbot," like the ones you find on customer support websites.

ELIZA used simple pattern matching and substitution to give the illusion of comprehension.

She was designed with the personality of a psychotherapist, so that she could reply with canned responses, like "Why do you say that?" or "Is that important to you?"

Weisenbaum actually created ELIZA to demonstrate how shallow computer language skills were, so imagine his surprise that many users actually believed she understood what they were saying.

I guess it's in our nature to anthropomorphize things. After all, I've been calling a computer program "she".

ELIZA was one of the first programs that could participate in the Turing Test, a theoretical experiment proposed by computer scientist Alan Turing about 15 years earlier as an assessment of artificial intelligence.

An evaluator would eavesdrop on a typed conversation between two participants: a human and a computer.

If the evaluator was unable to determine which was which, the computer was said to have passed the test.

Half a century later, computer technology has evolved exponentially. We all carry around a natural language processor in our pockets vastly more advanced than ELIZA.

We ask it to give us directions, keep track of our schedules, even turn our lights on and off.

But is any of this technology close to passing the Turing Test?

Even if it is, are computers really talking to us? Or are they still just faking it?

Not long after ELIZA, another computer scientist named Terry Winograd developed his own natural language processor called SHRDLU.

Unlike ELIZA, SHRDLU was supposed to actually understand what it was talking about--so to speak.

Its comprehension was limited to a handful of simple objects and actions, but users could instruct it to rearrange the objects around a virtual room, and then ask questions to verify that it knew what it had accomplished.

SHRDLU was one of the first successful attempts to teach a computer grammar, to put words into categories like nouns, verbs and prepositions.

To have rules about how they could combine and assign meanings to various combinations.

The technology was popularized in the text adventure video games of the 1980s, which allowed players to interact with the game world by typing simple verb-noun commands like "GET SWORD," "GO EAST," or "OPEN DOOR."

Half the fun of these games was trying different combinations to see just how... extensive the vocabulary was.

But this technology did not advance as quickly as some hoped, because programmers soon realized that they had to manually code all the rules of English grammar, which are a lot more complicated than "open door."

Even a simple rule like "add -ed to make a verb past tense" has enough exceptions to give a human ESL student a headache.

Even a simple sentence like "It's raining," would confound a computer because it couldn't proceed without knowing what "it" is.

Grammar is also dependent on the listener having at least some understanding of how the world works.

Consider these two sentences: Take the cap off the milk and pour it. Take the sheet off the bed and fold it.

These seem like pretty simple instructions, but that's only because we already know that milk pours and sheets fold.

A computer without that prior knowledge would have no way of knowing what the "it's" refer to.

And how about these two sentences: Sarah drew the dog with a pencil. Sarah drew the dog with a bone.

Again, neither of these sentences would give a human much trouble.

But a computer couldn't be sure of the meanings unless it already knew that dogs don't carry pencils, and that you can't draw with bones.

So just to get a computer to understand basic grammar, you'd have to manually encode it with vast complex rules of syntax and a broad understanding of how thousands of different objects interact with each other.

It's no wonder that in the 1990s computer scientists kind of gave up on so-called symbolic language processing in favor of a new strategy: statistical language processing.

Instead of trying to teach a computer the rules of language, they developed algorithms that could analyze large bodies of text, look for patterns, and then make guesses based on statistical probabilities.

For instance, when you ask Siri "What's the weather looking like today?" she doesn't bother parsing your grammar.

She simply homes in on keywords and guesses what you're looking for based on how common the request is.

She'd probably give you the same answer if you said "whether today" or even just "weather."

Another application of this technology is predictive text, which is what your phone does when it tries to guess which word you're going to type next.

For example, if you type the word "good" the algorithm knows from looking at thousands of pages of text that the most likely word to follow is "morning."

It doesn't know why or what those words mean--just that it's a statistical probability.

Believe it or not, there was a time when some thought this was how human speech worked; that our brains picked words one by one in order, deciding what word was most likely to follow the one before.

20th century linguists like Noam Chomsky have shown that human grammar is far too complex to be constructed this way.

Take this sentence, for instance: The fact that you went for a walk this late at night without an umbrella even after hearing the weather report on the radio this afternoon.

You can probably tell that there's something missing. That's because the opening few words obligate the speaker to finish the sentence with a verb phrase, like "is unbelievable."

Your brain subconsciously remembers this commitment, even though there are 23 words in between.

Language is full of such grammatical promises, like either-ors or if-thens.

A computer program that only considers one word at a time would never be able to fulfill them.

However, recent advancements in digital neural networks are raising expectations of what predictive text can achieve.

In 2020, the artificial intelligence company OpenAI released a beta version of one of the most sophisticated natural language processors ever created, called GPT-3.

To find out how GPT-3's extraordinary language capabilities are achieved, I spoke with OpenAI technical director Dr. Ashley Pilipiszyn.

How is GPT-3 different from previous NLPs?

Yeah, so unlike most AI systems which are designed for one use case, GPT-3 and our API provides a general purpose text in text out interface, and it allows users to try it on virtually any English language task.

Say I have a piece of a legal document maybe an NDA and I would ask GPT-3 to summarize this legal document like a second grader.

And GPT-3 would then be able to provide a couple sentences actually compressing and making that legal document into a much more understandable piece of text.

So our model actually doesn't have a goal or objective other than predicting the next word.

Like most predictive text programs, GPT-3 is trained by feeding it a large body of text for analysis, known as a corpus.

And GPT-3's corpus is enormous. Somewhere around 2 billion pages of text, taken from wikipedia, digital books, and a vast swath of the web.

It analyzes this text, using hundreds of billions of parameters looking for probabilities.

And because it does much of it unsupervised, even its programmers don't know exactly what patterns it's finding in our human speech.

When GPT-3 is asked to complete a prompt, it uses what it's learned to guess what should come next.

But where your phone guesses words, GPT-3 guesses "tokens": four character blocks of text including spaces and symbols.

And can you tell us a little bit more about these particular tokens? So you and I as humans, when we see a sentence we see a specific set of words with spaces in between, etc.

When GPT-3 quote-unquote "sees," they actually are seeing tokens, which you can think of actually like a jigsaw puzzle--that allows GPT-3 to process more text.

It's really trying to predict what the next token is going to be in a sentence, based on all of the previous text it's seen before in that prompt.

Because of its longer memory it's able to complete grammatical commitments like "the fact that..."

And thanks to its huge corpus, it actually does seem to know that dogs are more interested in bones than pencils.

It can even apply grammatical rules to unfamiliar words.

A famous experiment from the 1950s asked children to complete the following prompt.

Even though they had never seen the word "wug" before, the vast majority of children were able to correctly apply an "-s" to make it plural.

Similarly, even though GPT-3 has probably never seen the word "ferfumfer" in all its billions of pages of corpus, it still knows to add an "-s" if you have two of them.

So can GPT-3 pass the Turing Test?

Uh, most people familiar with the traditional Turing Test would probably say it comes very close.

It can actually start to feel like you really are interacting, you know, with that person.

But the longer you talk with it and really begin to push it, you do come across some mistakes. And so that does kind of indicate that okay yeah this isn't human.

So ultimately, it's it's coming close but it's not quite there.

Despite GPT-3's often impressive performance, there is a fundamental difference between human speech and what GPT-3 does.

Humans don't learn language by memorizing likely orders of words, but instead word categories.

Chomsky demonstrated this with a famously nonsensical sentence: "Colorless green ideas sleep furiously."

Even though all your life you've probably never heard any of these words follow the one before it, you still know that the sentence is grammatically correct, because the order of the word categories is correct.

But GPT-3 does not make a distinction between form and content.

It doesn't care that "colorless green ideas" is a grammatically correct noun phrase.

Only that the likelihood of those tokens going together is very, very small.

That's why a slightly more advanced question from the wug test can get... interesting results.

Our brains seem to be hardwired for grammar, which ironically is closer to how the old SHRDLU program worked.

We have thoughts about the world that exist prior to and independent of language, and we use grammar as the container to deliver those thoughts to others.

For all its complexity, GPT-3 is still guessing one token at a time, based on statistical probabilities.

It has no plan or goal, unless given one by a human. That's why the developers at OpenAI prefer to think of GPT-3 as a writer's tool rather than a writer itself.

It's pretty astounding at faking what human speech sounds like, but humans are still the only computers that can create the thoughts behind the words.

A big thanks to Dr. Ashley Pilipiszyn from OpenAI, and many of the responses shown in this episode were obtained through AI Dungeon, a procedurally generated text adventure powered by GPT-3.

You can check it out yourself through the link in the description.