(/)Jay Alammar (/)

Visualizing machine learning one concept at a time.
@JayAlammar (https://twitter.com/JayAlammar) on Twitter. YouTube Channel (https://www.youtube.com/channel/UCmOwsoHty5PrmE-3QhUBfPQ)

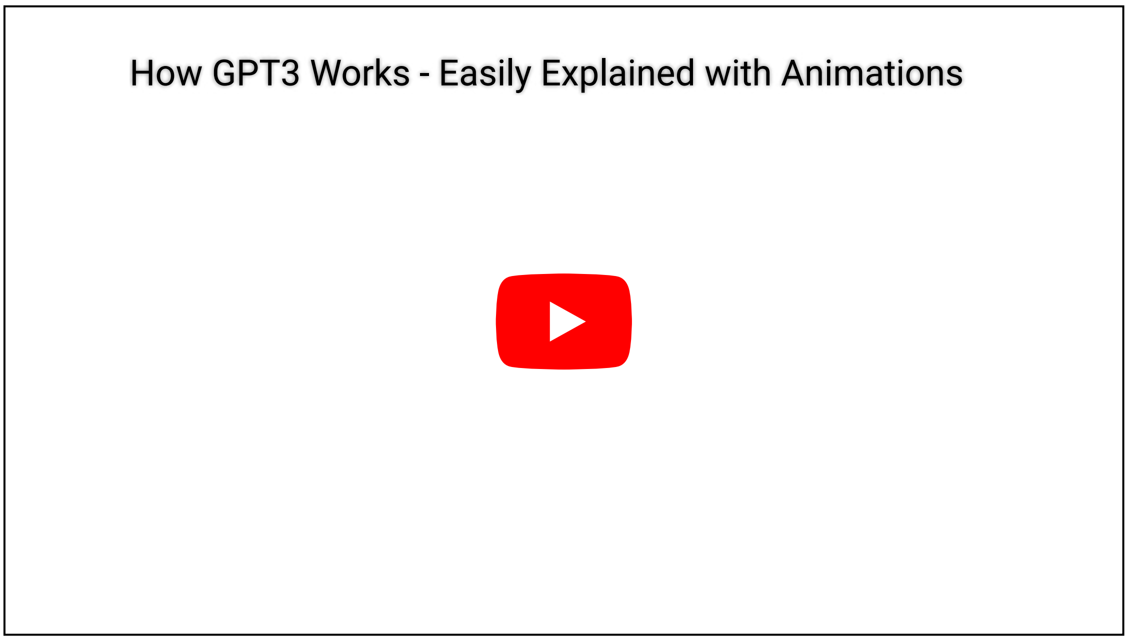
Blog (/) About (/about)

How GPT3 Works - Visualizations and Animations

Discussions: Hacker News (397 points, 97 comments) (https://news.ycombinator.com/item?id=23967887), Reddit r/MachineLearning (247 points, 27 comments) (https://www.reddit.com/r/MachineLearning/comments/hwxn26/p_how_gpt3_works_visuals_and_animations/)

Translations: German (https://www.arnevogel.com/wie-gpt3-funktioniert/), Korean (https://chloamme.github.io/2021/12/18/how-gpt3-works-visualizations-animations-korean.html), Chinese (Simplified) (https://blogcn.acacess.com/how-gpt3-works-visualizations-and-animations-zhong-yi), Russian (https://habr.com/ru/post/514698/)

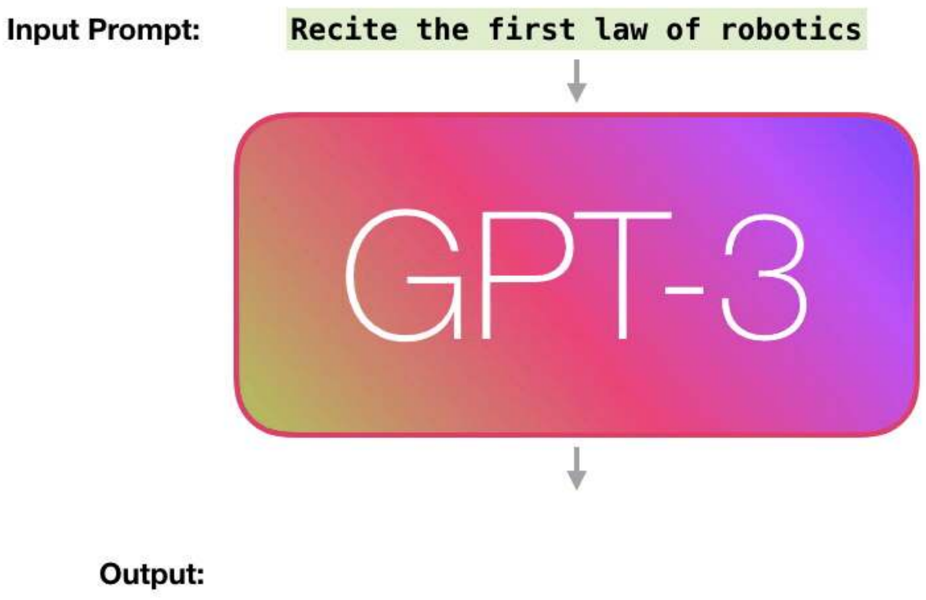
The tech world is abuzz (https://www.theverge.com/21346343/gpt-3-explainer-openai-examples-errors-agi-potential) with GPT3 hype. Massive language models (like GPT3) are starting to surprise us with their abilities. While not yet completely reliable for most businesses to put in front of their customers, these models are showing sparks of cleverness that are sure to accelerate the march of automation and the possibilities of intelligent computer systems. Let’s remove the aura of mystery around GPT3 and learn how it’s trained and how it works.



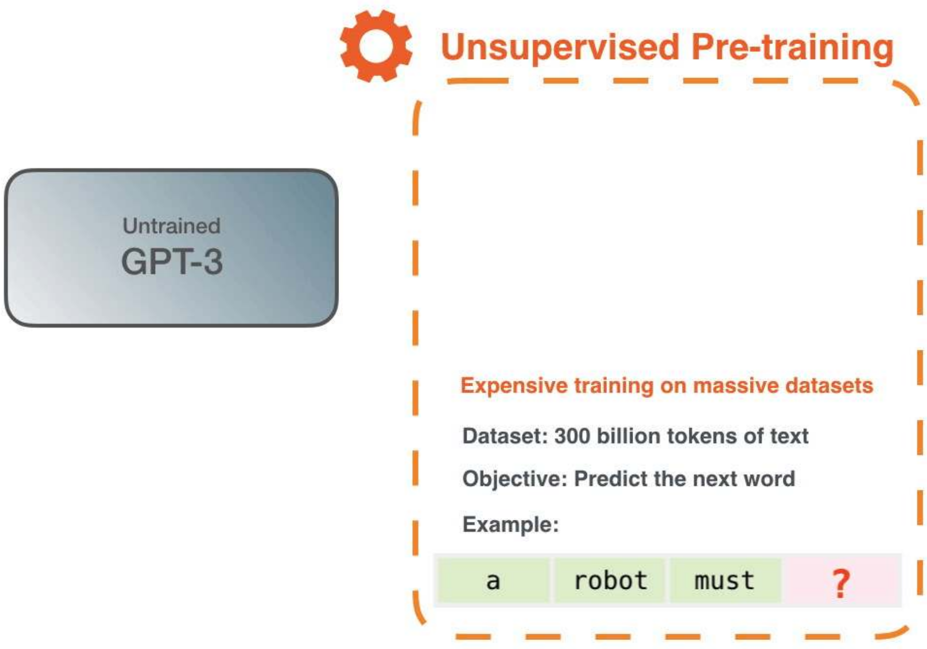
A trained language model generates text.

We can optionally pass it some text as input, which influences its output.

The output is generated from what the model “learned” during its training period where it scanned vast amounts of text.

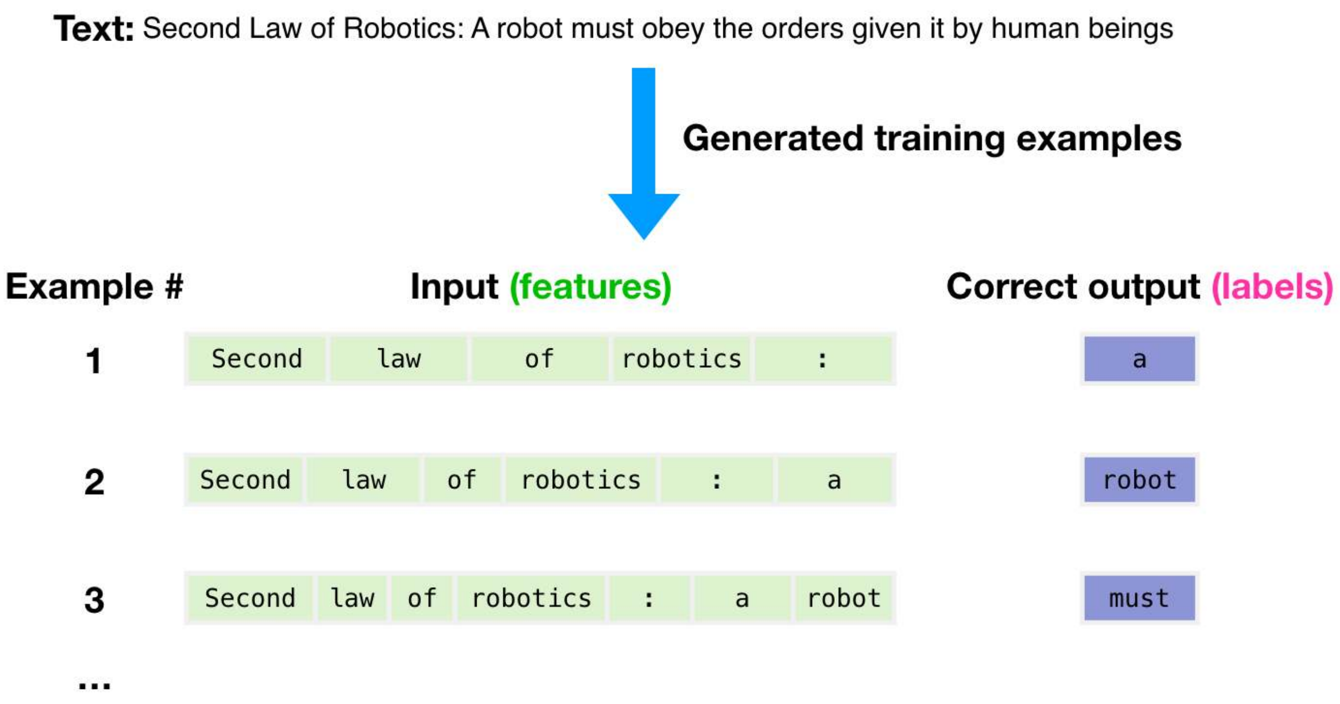


Training is the process of exposing the model to lots of text. That process has been completed. All the experiments you see now are from that one trained model. It was estimated to cost 355 GPU years and cost $4.6m.


Unsupervised Pre-training
Untrained
GPT-3
Expensive training on massive datasets
Dataset: 300 billion tokens of text
Objective: Predict the next word
Example:
a
robot
must
?

The dataset of 300 billion tokens of text is used to generate training examples for the model. For example, these are three training examples generated from the one sentence at the top.

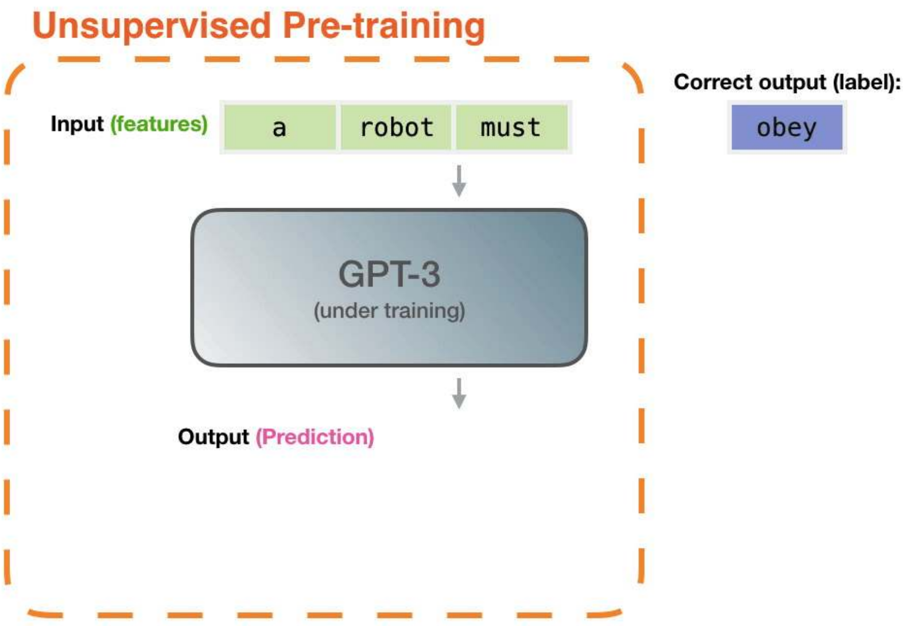
You can see how you can slide a window across all the text and make lots of examples.


Text: Second Law of Robotics: A robot must obey the orders given it by human beings
Generated training examples
Example #
Input (features)
Correct output (labels)
1
Second law of robotics :
a
2
Second law of robotics : a
robot
3
Second law of robotics : a robot
must
...

The model is presented with an example. We only show it the features and ask it to predict the next word.

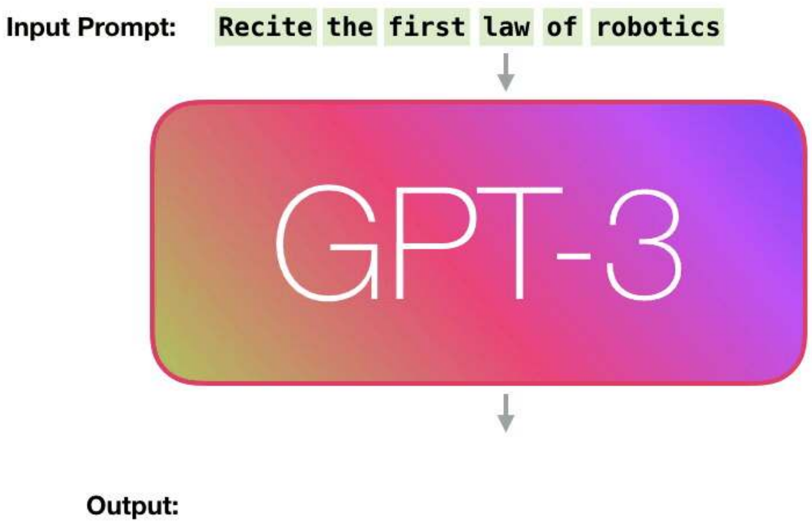
The model’s prediction will be wrong. We calculate the error in its prediction and update the model so next time it makes a better prediction.

Repeat millions of times



Now let’s look at these same steps with a bit more detail.

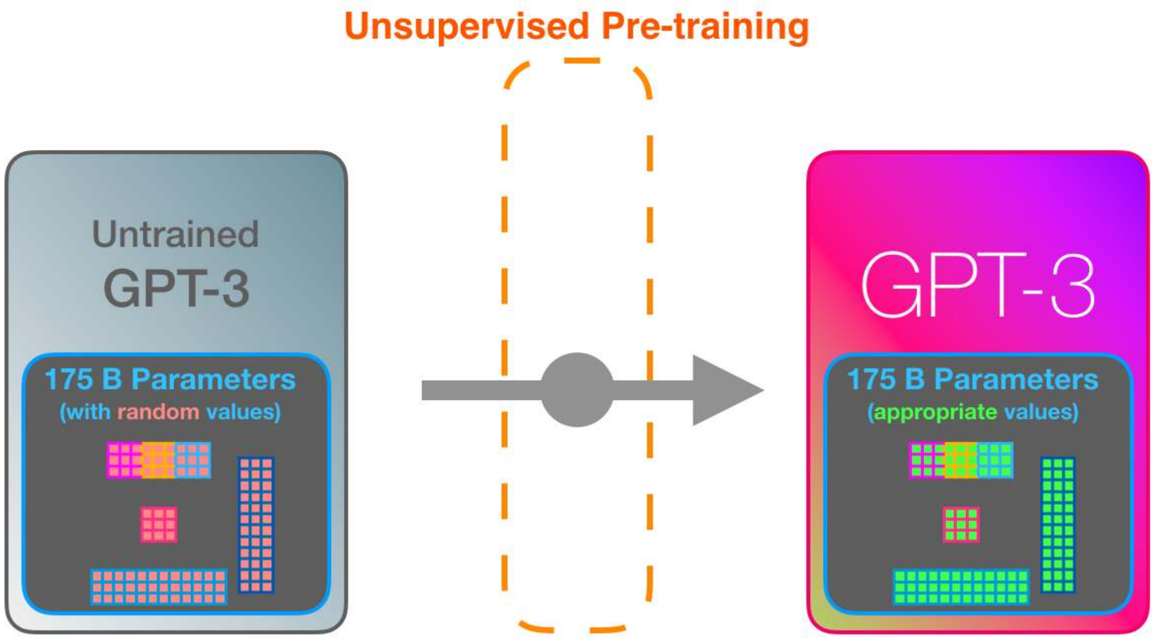
GPT3 actually generates output one token at a time (let’s assume a token is a word for now).



Please note: This is a description of how GPT-3 works and not a discussion of what is novel about it (which is mainly the ridiculously large scale). The architecture is a transformer decoder model based on this paper https://arxiv.org/pdf/1801.10198.pdf

GPT3 is MASSIVE. It encodes what it learns from training in 175 billion numbers (called parameters). These numbers are used to calculate which token to generate at each run.

The untrained model starts with random parameters. Training finds values that lead to better predictions.

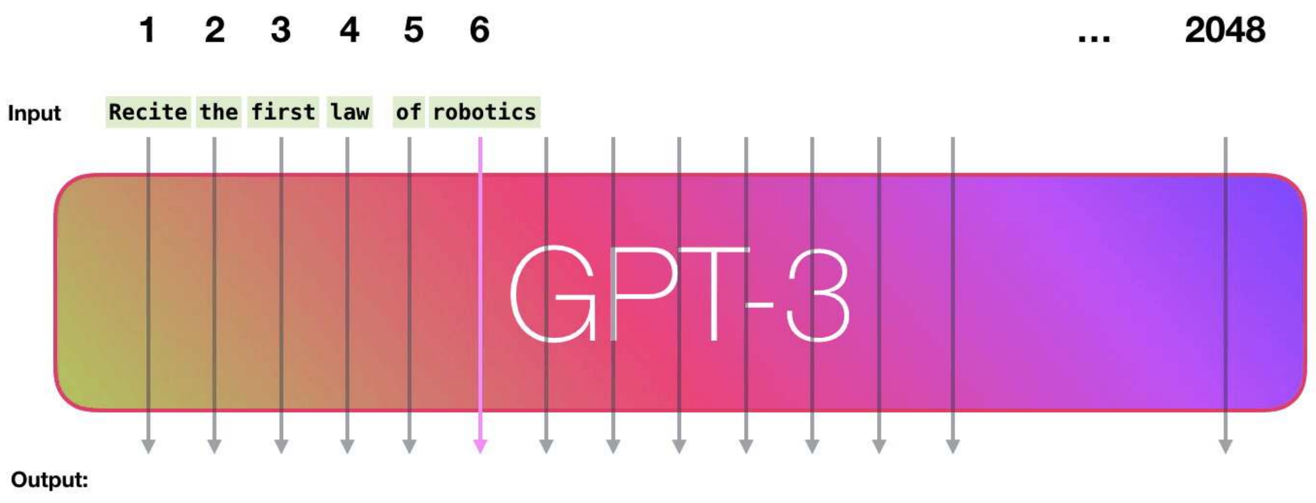


These numbers are part of hundreds of matrices inside the model. Prediction is mostly a lot of matrix multiplication.

In my Intro to AI on YouTube (https://youtube.com/watch?v=mSTCzNgDJy4), I showed a simple ML model with one parameter. A good start to unpack this 175B monstrosity.

To shed light on how these parameters are distributed and used, we'll need to open the model and look inside.

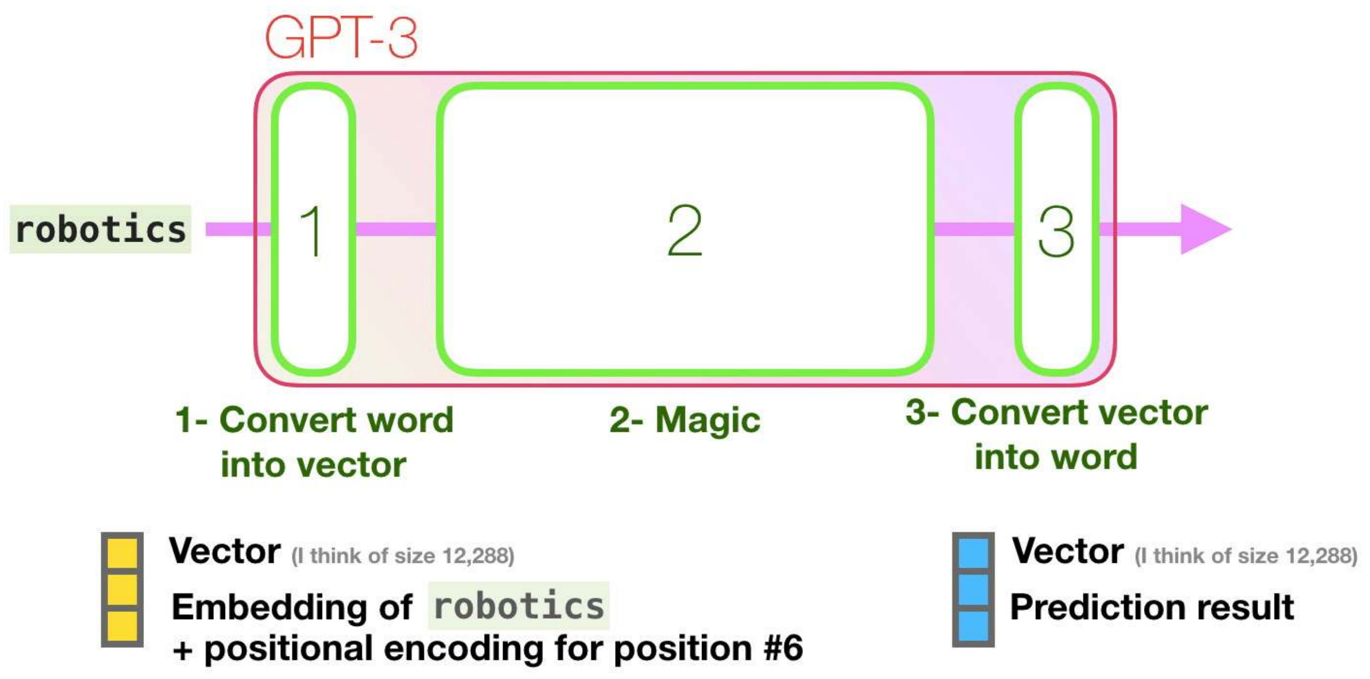
GPT3 is 2048 tokens wide. That is its “context window”. That means it has 2048 tracks along which tokens are processed.



Let's follow the purple track. How does a system process the word "robotics" and produce "A"?

High-level steps:

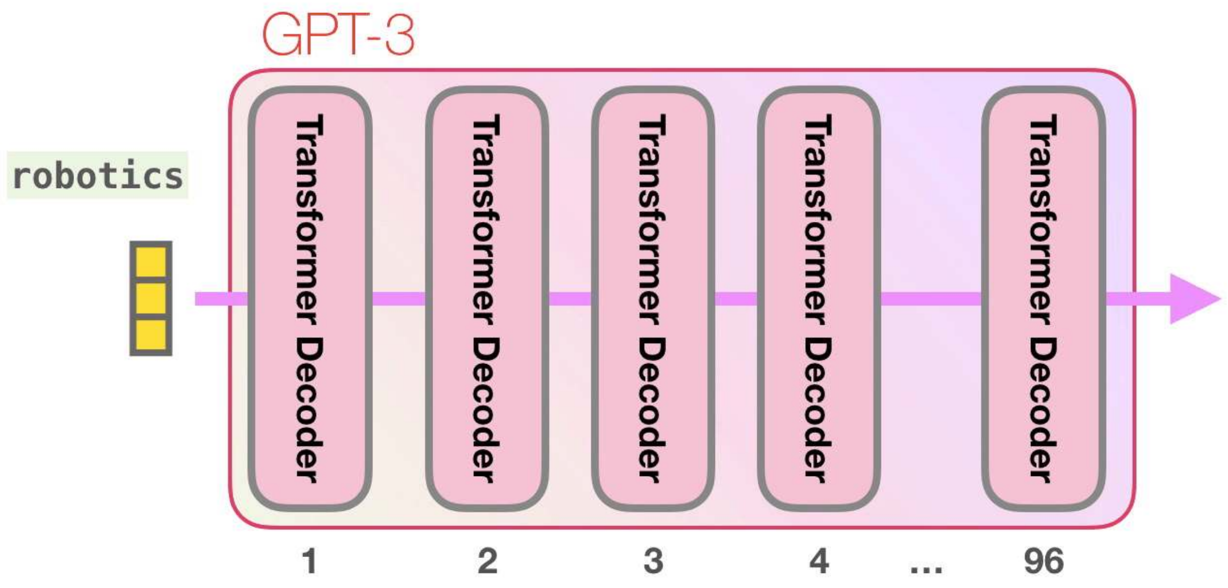
1. Convert the word to a vector (list of numbers) representing the word (https://jalammar.github.io/illustrated-word2vec/)
2. Compute prediction
3. Convert resulting vector to word


GPT-3
robotics
1
2
3
1- Convert word into vector
2- Magic
3- Convert vector into word
Vector (I think of size 12,288)
Embedding of robotics + positional encoding for position #6
Vector (I think of size 12,288)
Prediction result

The important calculations of the GPT3 occur inside its stack of 96 transformer decoder layers.

See all these layers? This is the “depth” in “deep learning”.

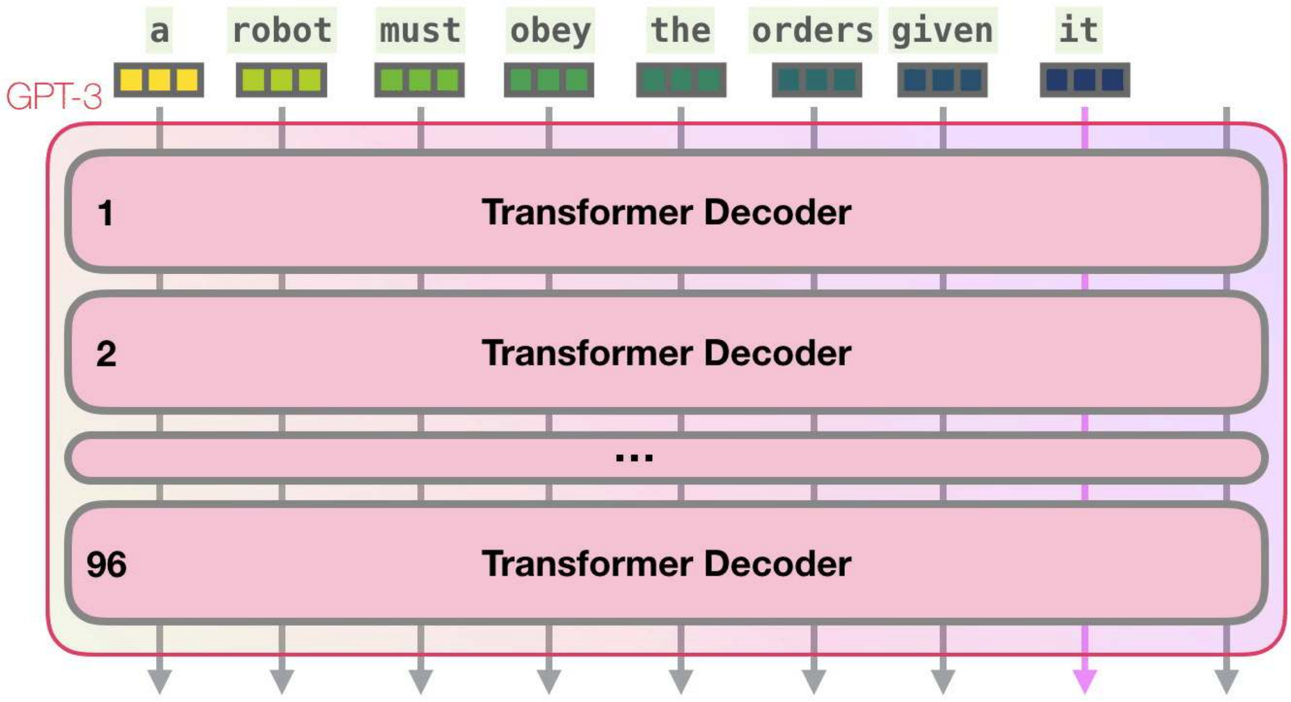
Each of these layers has its own 1.8B parameter to make its calculations. That is where the “magic” happens. This is a high-level view of that process:



You can see a detailed explanation of everything inside the decoder in my blog post The Illustrated GPT2 (https://jalammar.github.io/illustrated-gpt2/).

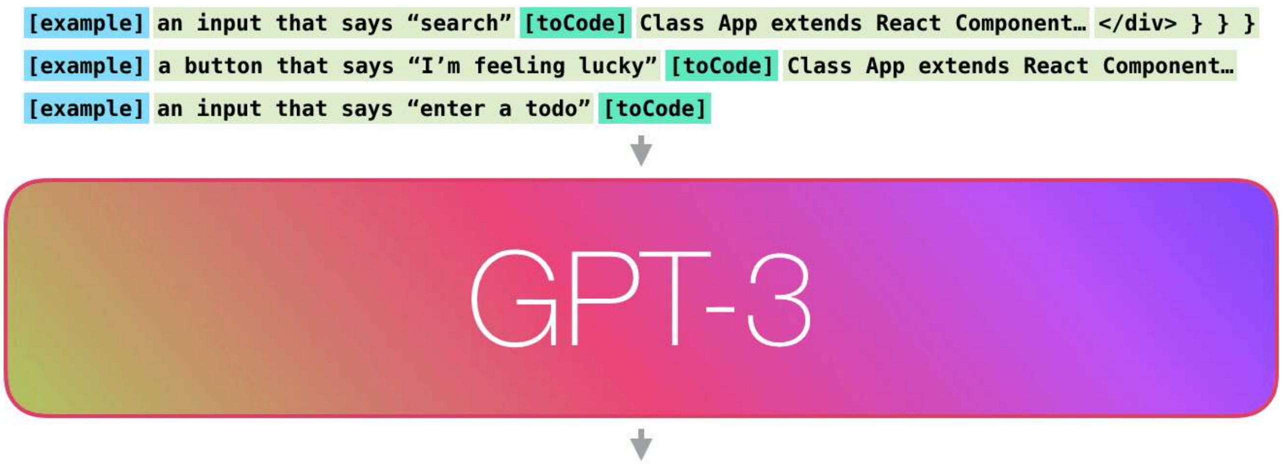
The difference with GPT3 is the alternating dense and sparse self-attention layers (https://arxiv.org/pdf/1904.10509.pdf).

This is an X-ray of an input and response (“Okay human”) within GPT3. Notice how every token flows through the entire layer stack. We don’t care about the output of the first words. When the input is done, we start caring about the output. We feed every word back into the model.



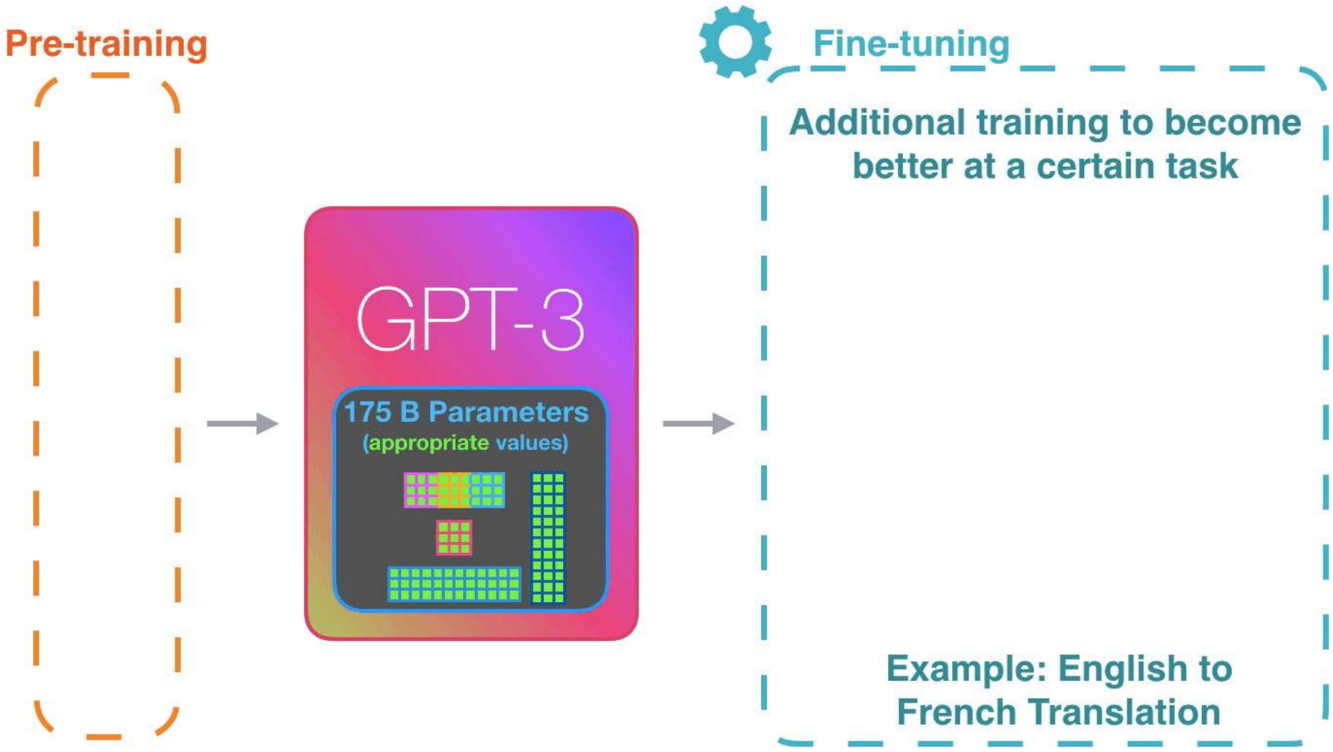
In the React code generation example (https://twitter.com/sharifshameem/status/1284421499915403264), the description would be the input prompt (in green), in addition to a couple of examples of description=>code, I believe. And the react code would be generated like the pink tokens here token after token.

My assumption is that the priming examples and the description are appended as input, with specific tokens separating examples and the results. Then fed into the model.


```
[example] an input that says "search" [toCode] Class App extends React Component… </div> } } }
[example] a button that says "I'm feeling lucky" [toCode] Class App extends React Component…
[example] an input that says "enter a todo" [toCode]
```

It’s impressive that this works like this. Because you just wait until fine-tuning is rolled out for the GPT3. The possibilities will be even more amazing.

Fine-tuning actually updates the model’s weights to make the model better at a certain task.


Pre-training
GPT-3
175 B Parameters
(appropriate values)
Fine-tuning
Additional training to become better at a certain task
Example: English to French Translation

Written on July 27, 2020

Subscribe to get notified about upcoming posts by email

Email Address

Subscribe

(https://github.com/jalammar) (https://www.linkedin.com/in/jalammar)
(https://www.twitter.com/jayalammar)