

June 26, 2023

Text-to-text AI Self-attention

GPT is Open AI's large language model.

It generates human-like text in response to a prompt.



**ChatGPT is a front end for the model,
where users can interact with it as a chatbot.**

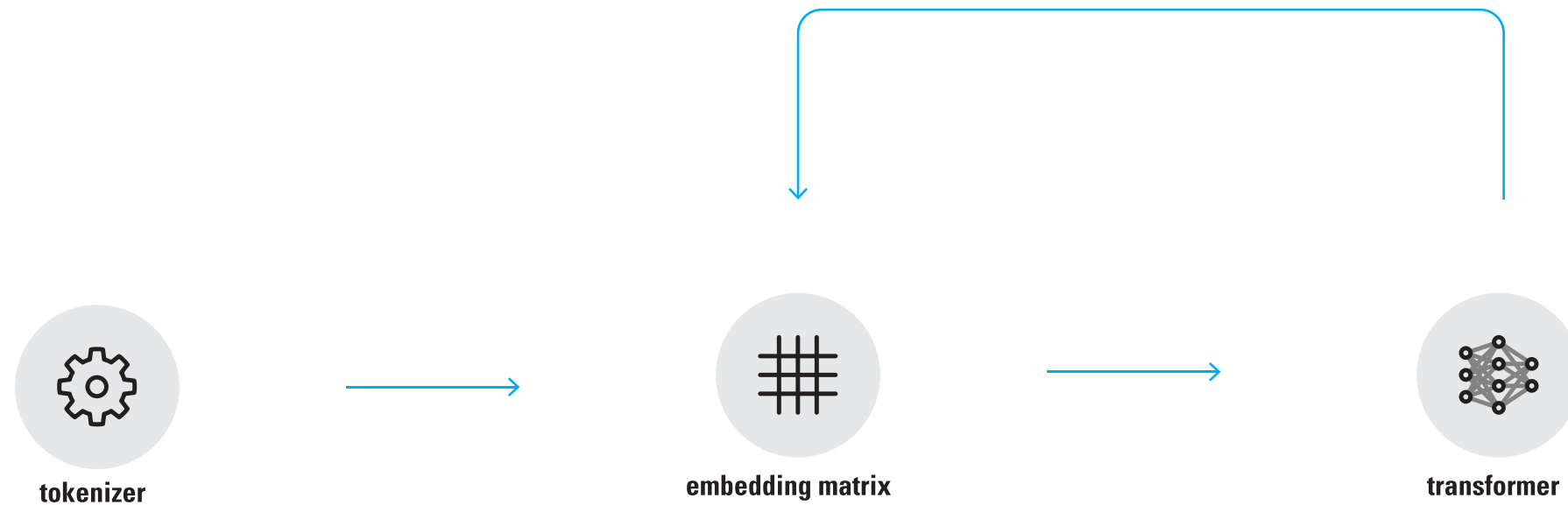
users:
input text



GPT generates:
output text

GPT has 3 main parts:

The tokenizer, the embedding matrix, and the transformer.



Let's start with a simple overview.
First the user inputs text as a prompt.



input text

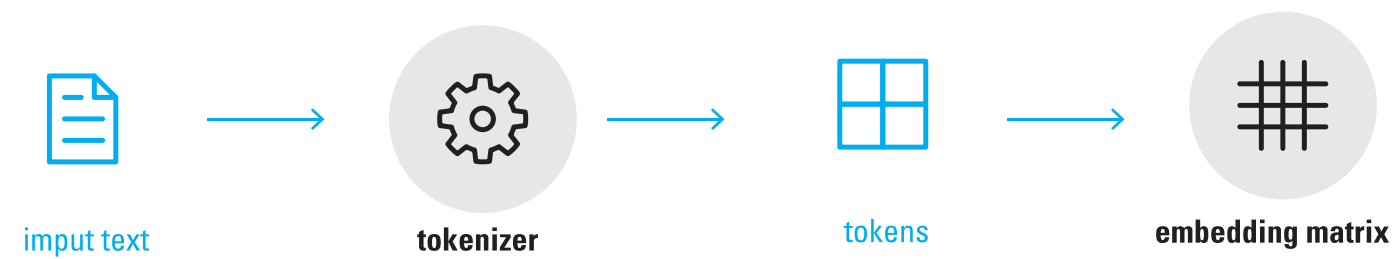
The text is passed to the tokenizer.



The tokenizer breaks up the input into a set of tokens.



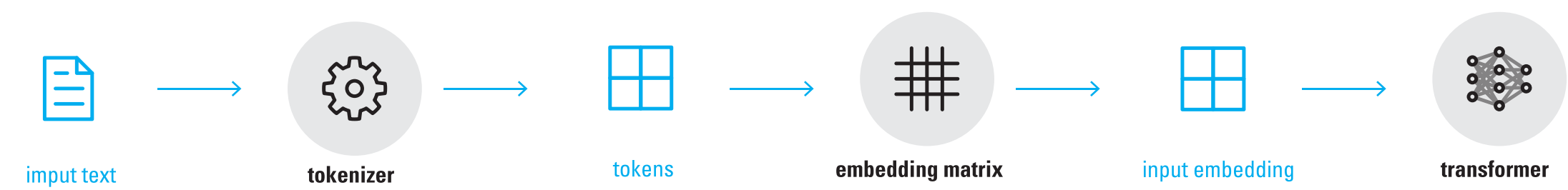
The tokens are sent to the embedding matrix.



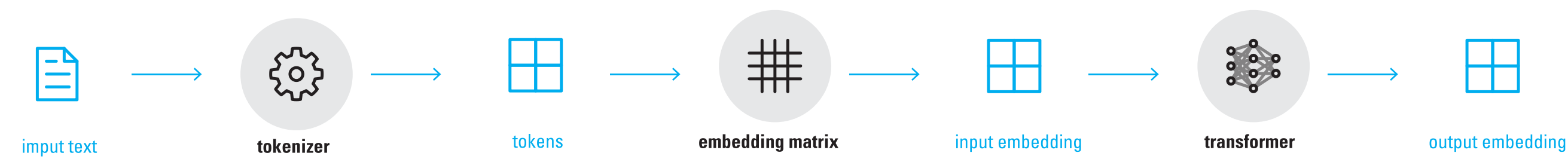
The tokens are assigned their respective vector embeddings.



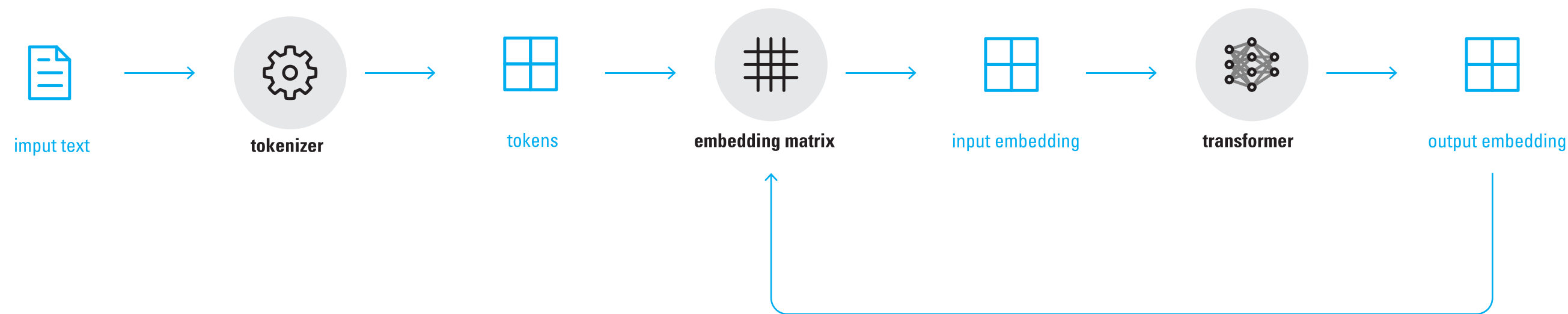
The vector embeddings are sent to the transformer.



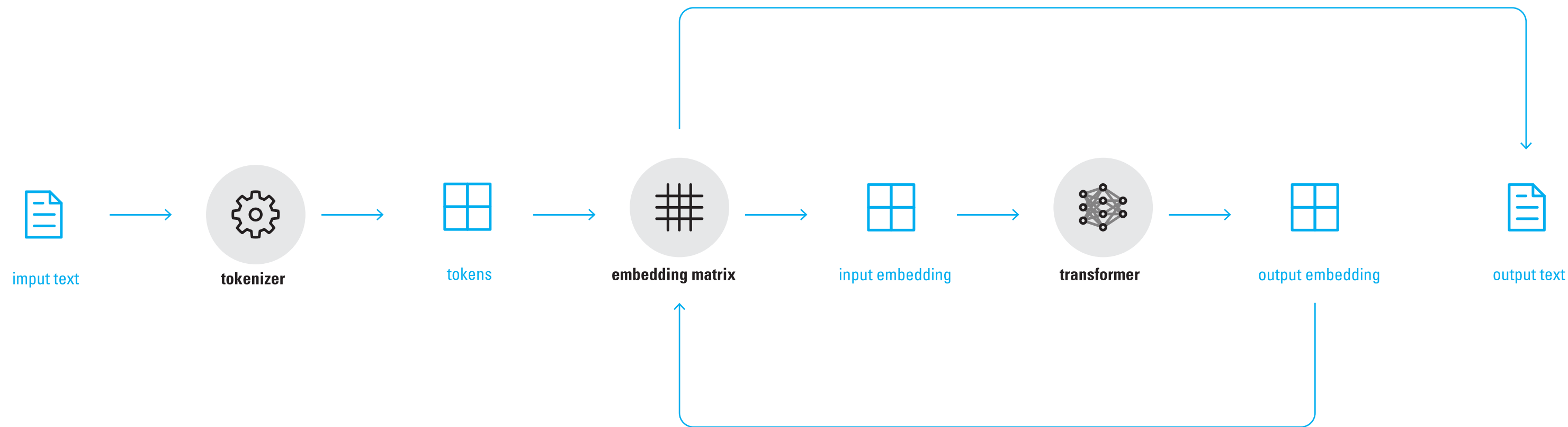
The transformer transforms the embeddings into the embedding for the next token.



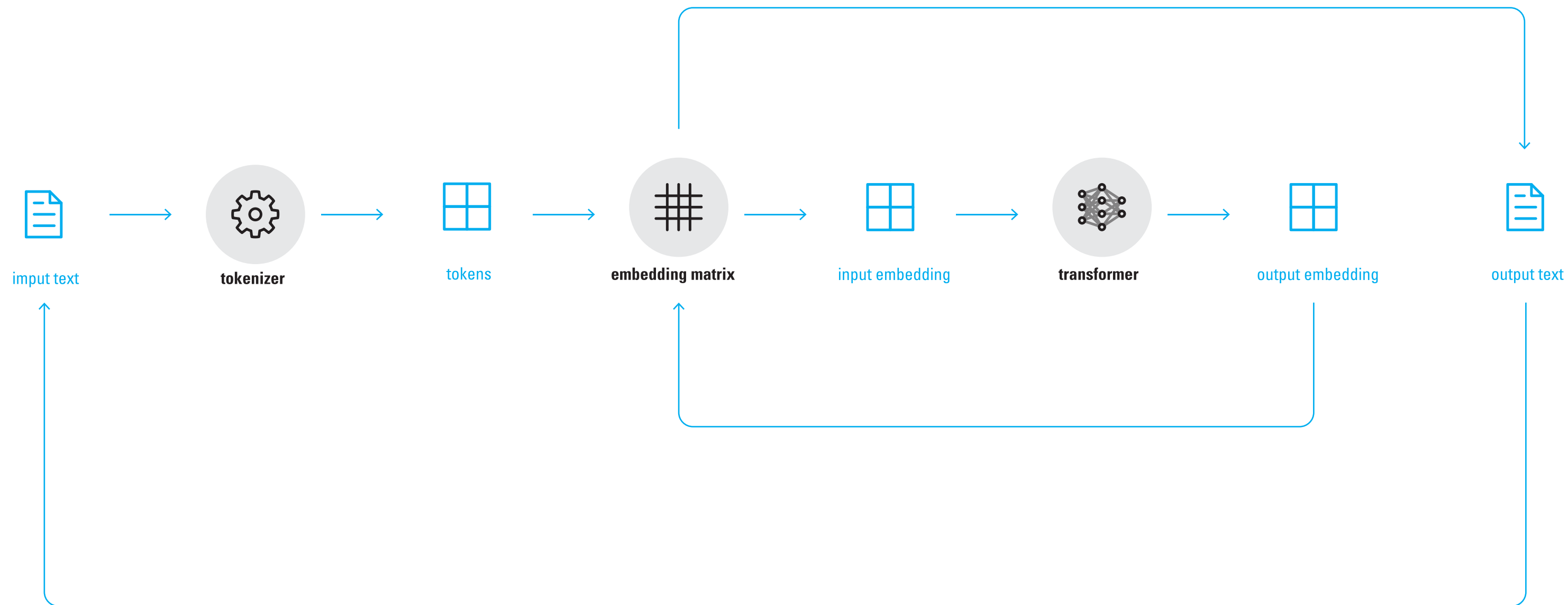
The output embeddings are sent back to the embedding matrix.



**The output embedding is multiplied by the matrix,
resulting in a list of probabilities for all the tokens in the vocabulary.**

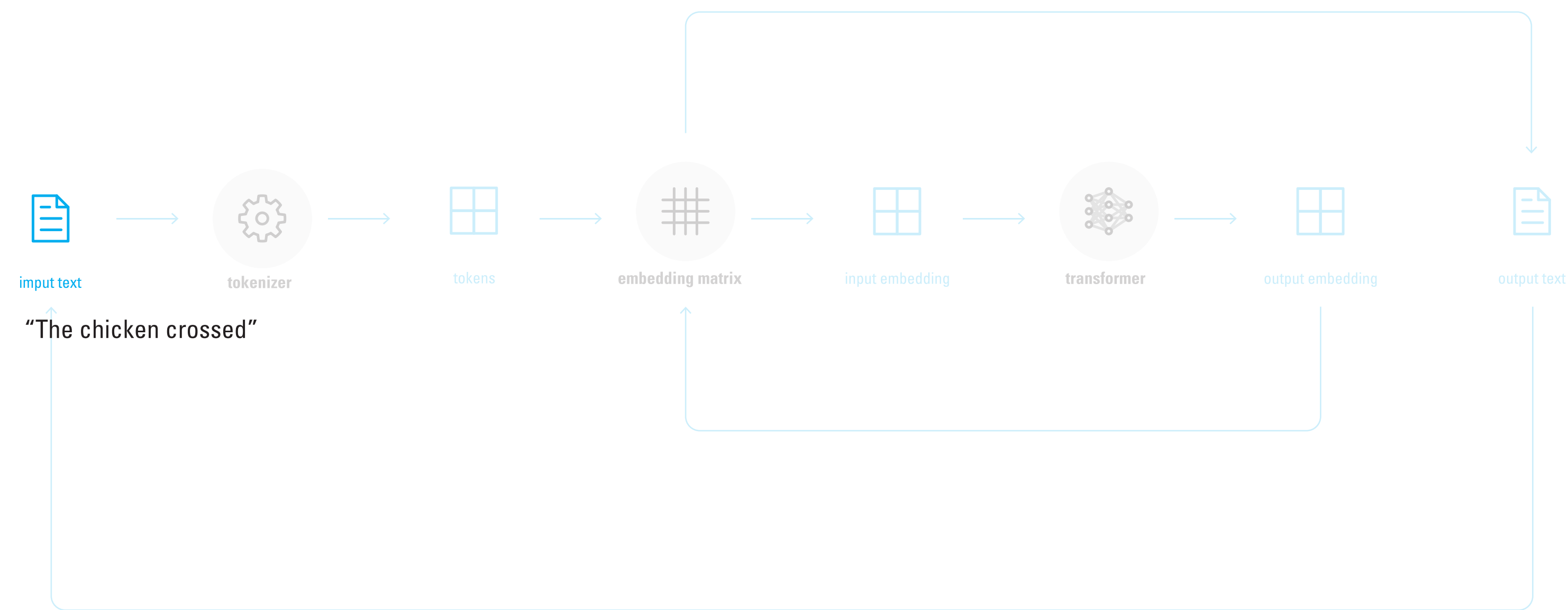


**The output token is selected from the top of this list,
appended to the input, and passed back through the system to predict the next token.**

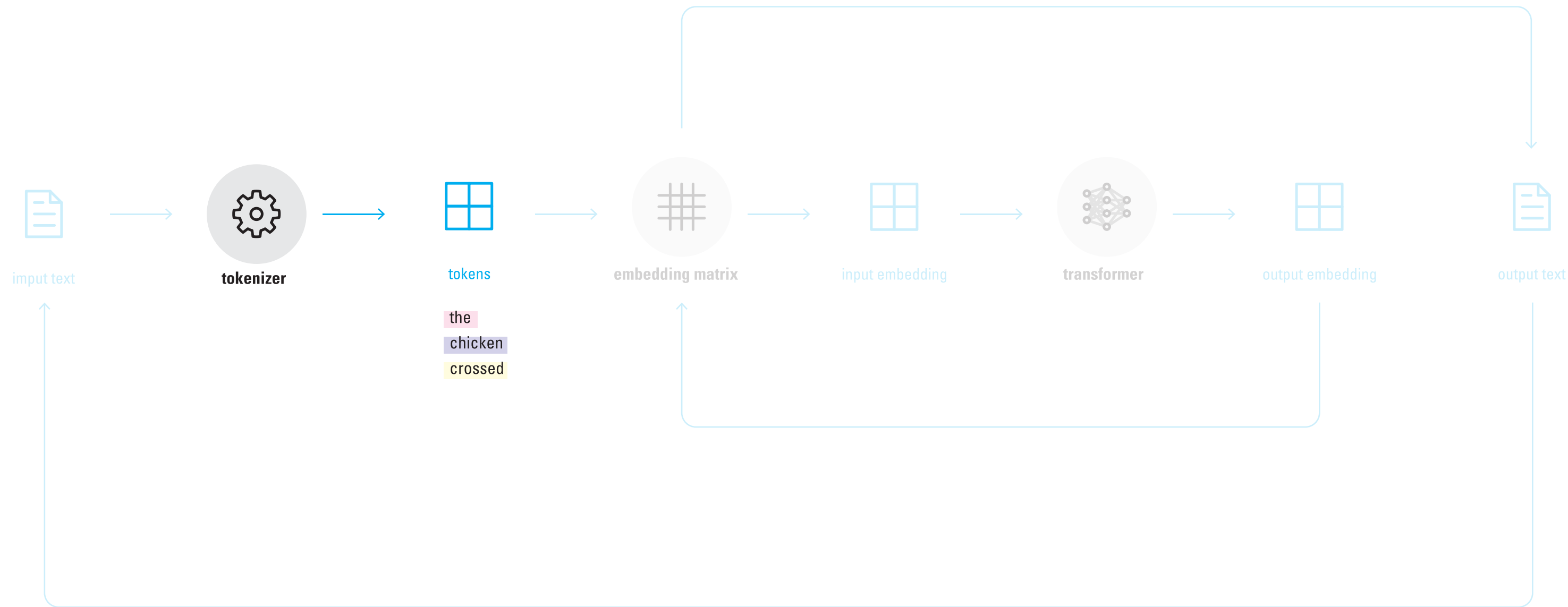


Let's follow the input "The chicken crossed" through the model.

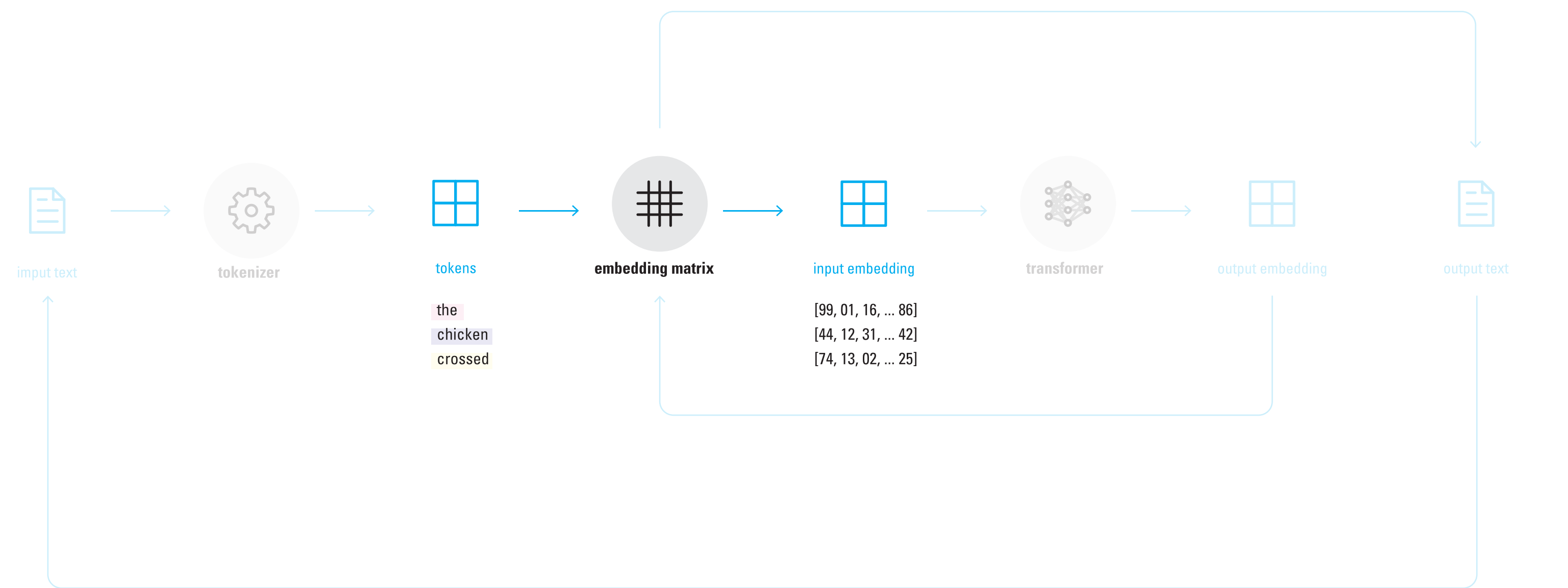
First, it is input as a prompt by the user.



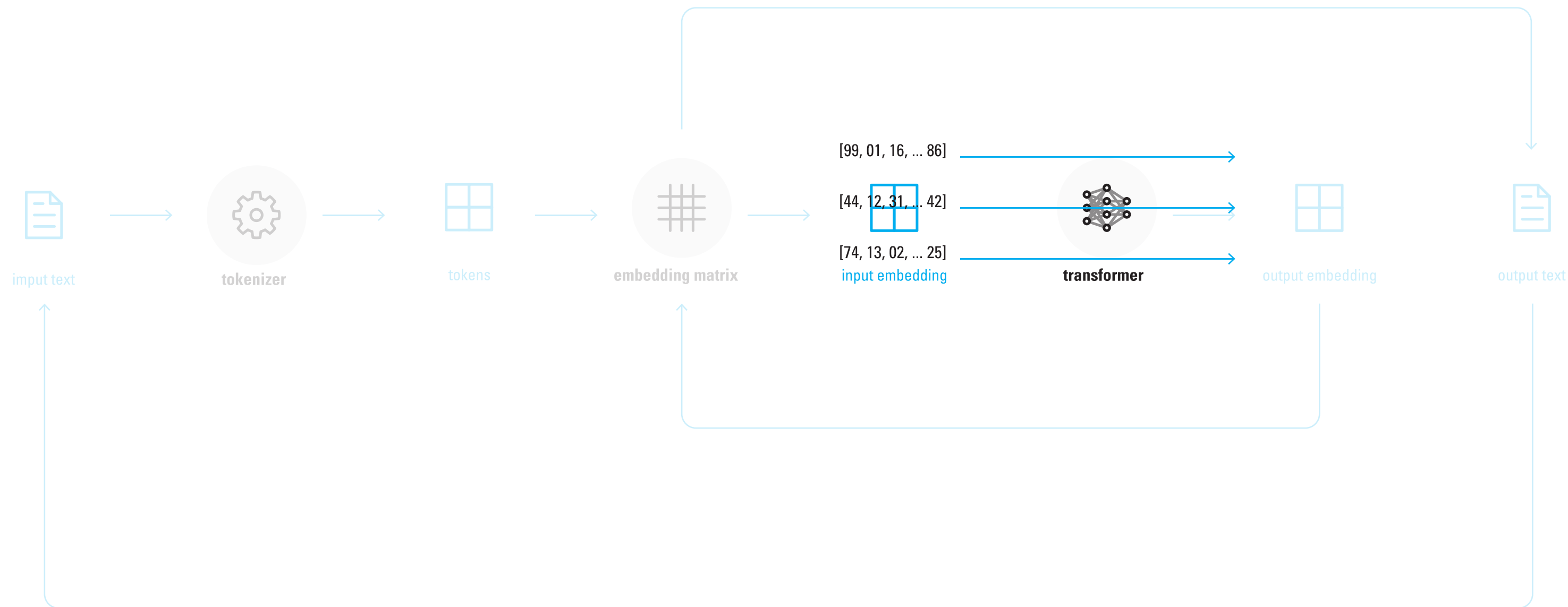
Then it is split up by the tokenizer



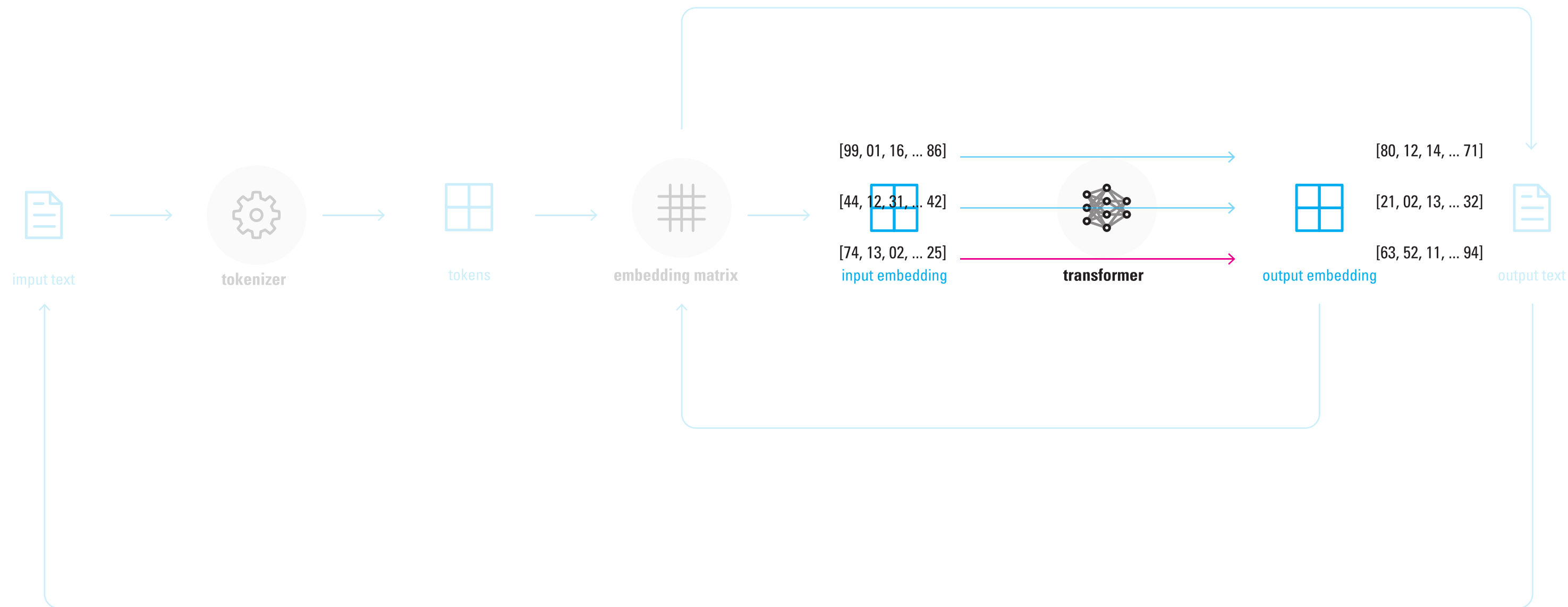
Each token is converted into its vector embedding by the embedding matrix.



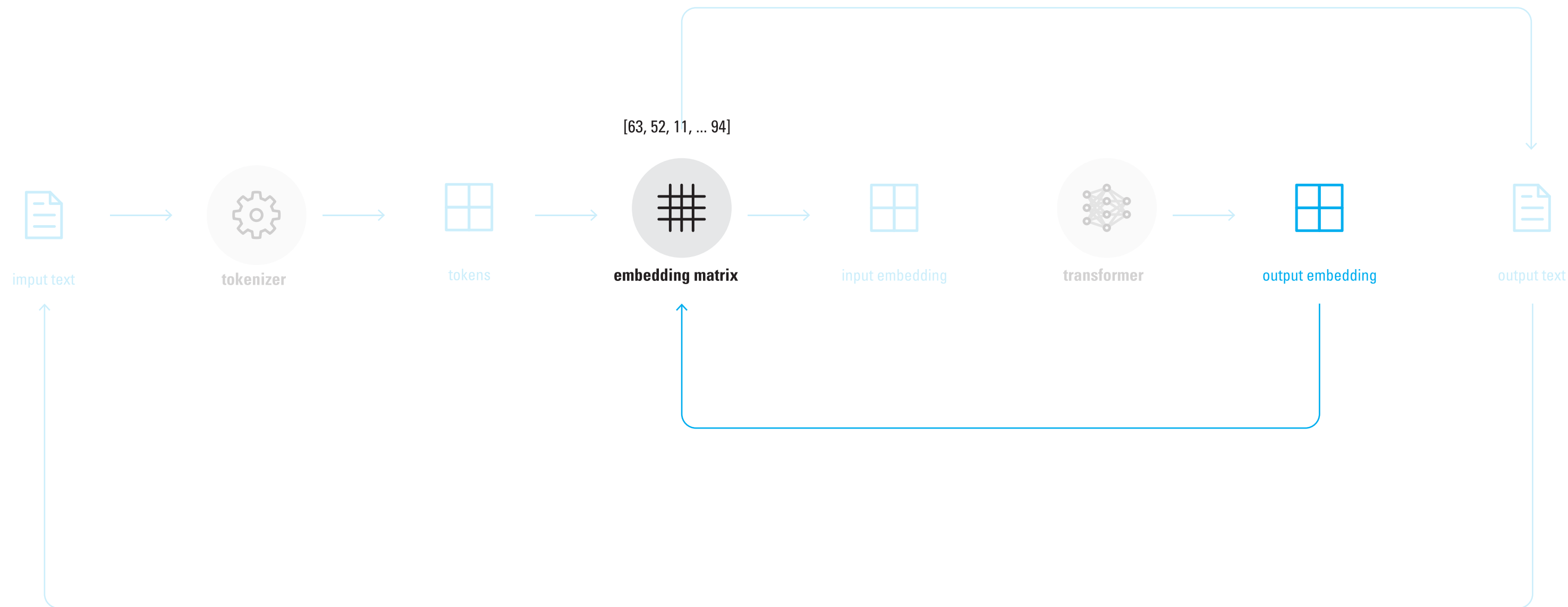
Then, each token embedding is processed by the transformer on its own path.



Each embedding is transformed, but only the last one is used to predict the next token.



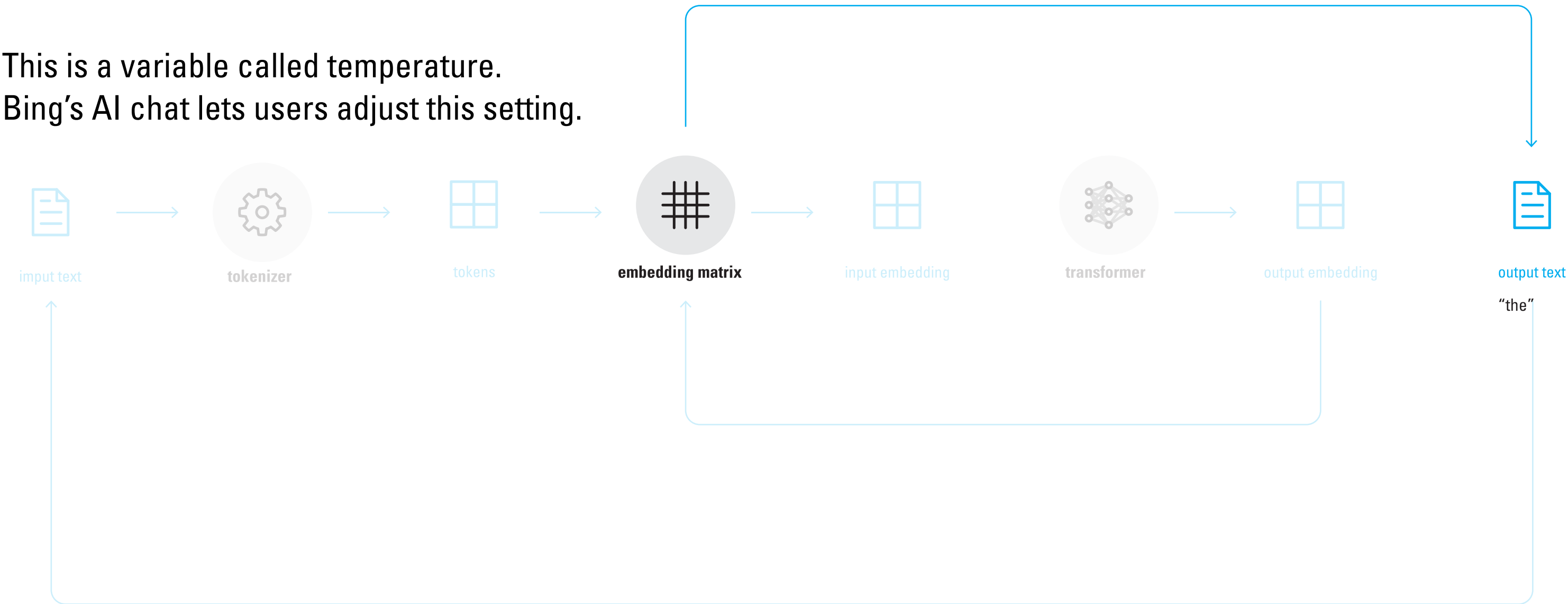
The output token is passed back and multiplied by the embedding matrix, to generate a list of probabilities for all tokens in the models vocabulary.



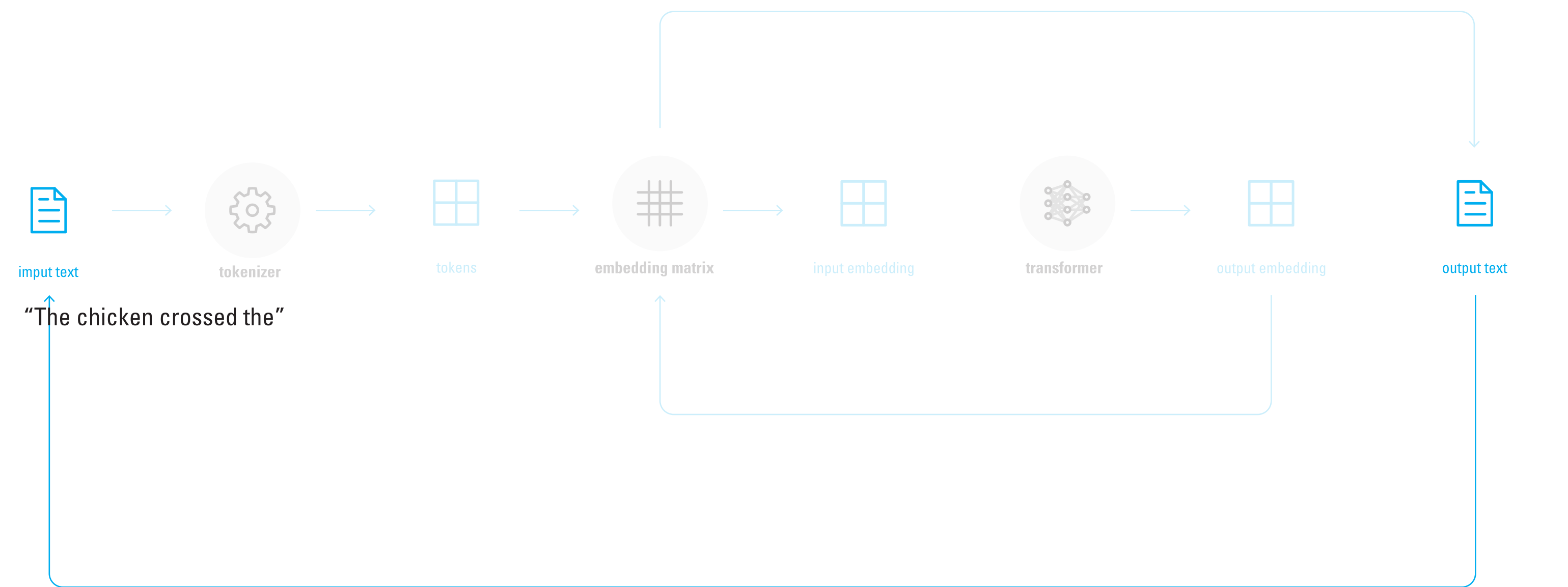
The one of the most probable tokens in the list is chosen as the next token in the sequence, and outputted by the model.

Choosing the most probable token every time yields very repetitive and robot like text.
Occasionally choosing from the runner ups yields more creative sounding results.

This is a variable called temperature.
Bing’s AI chat lets users adjust this setting.

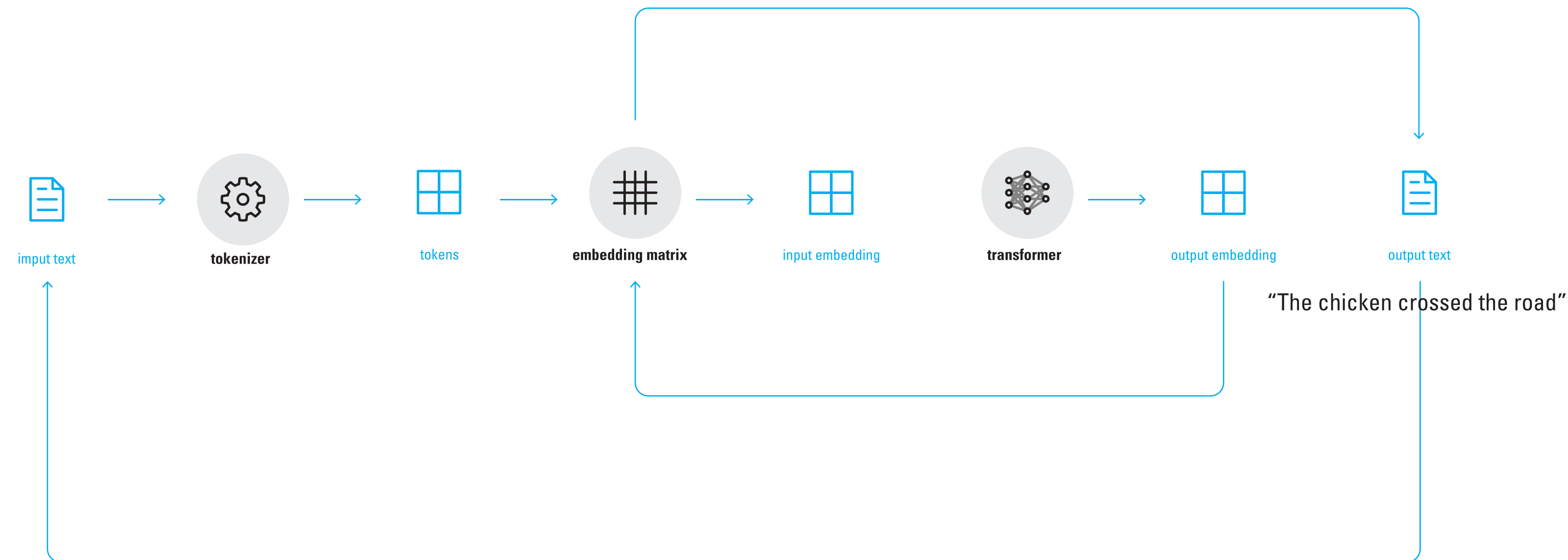


**The output is appended to the input,
and the process starts again with the new phrase “The chicken crossed the”**

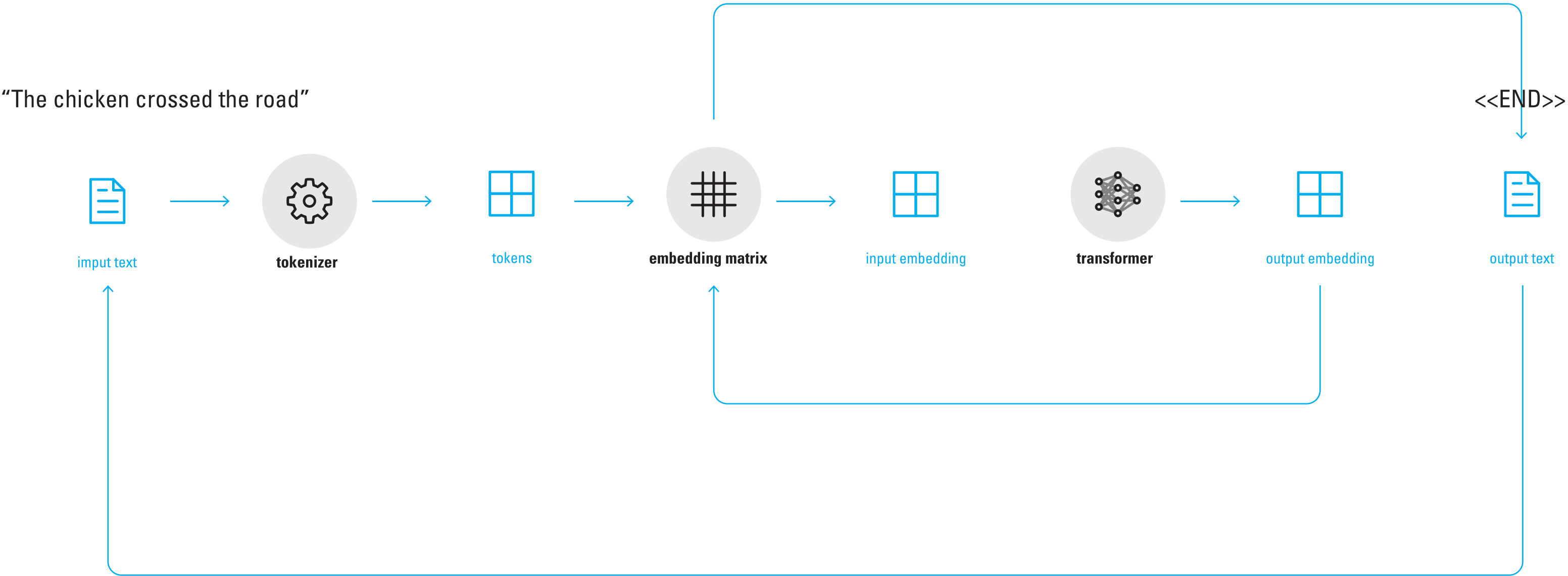


Through the system again, we get the complete phrase

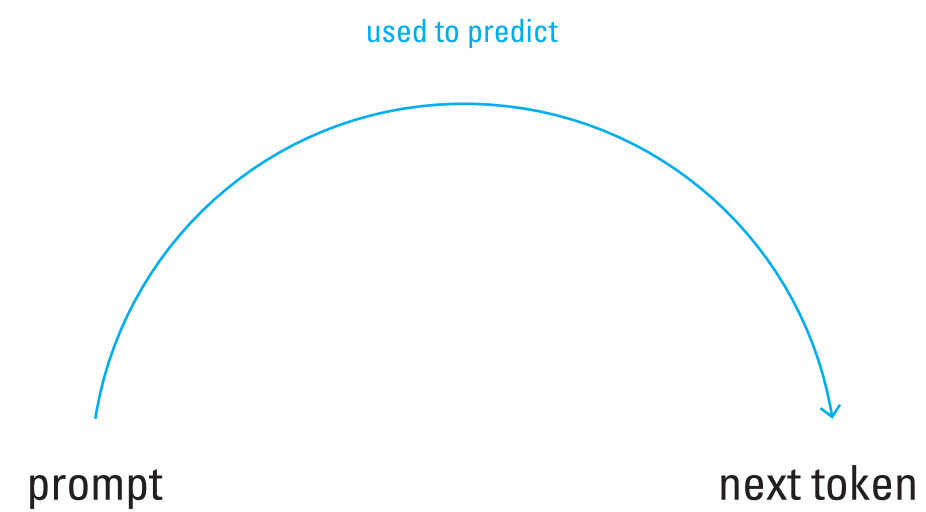
“The chicken crossed the road”



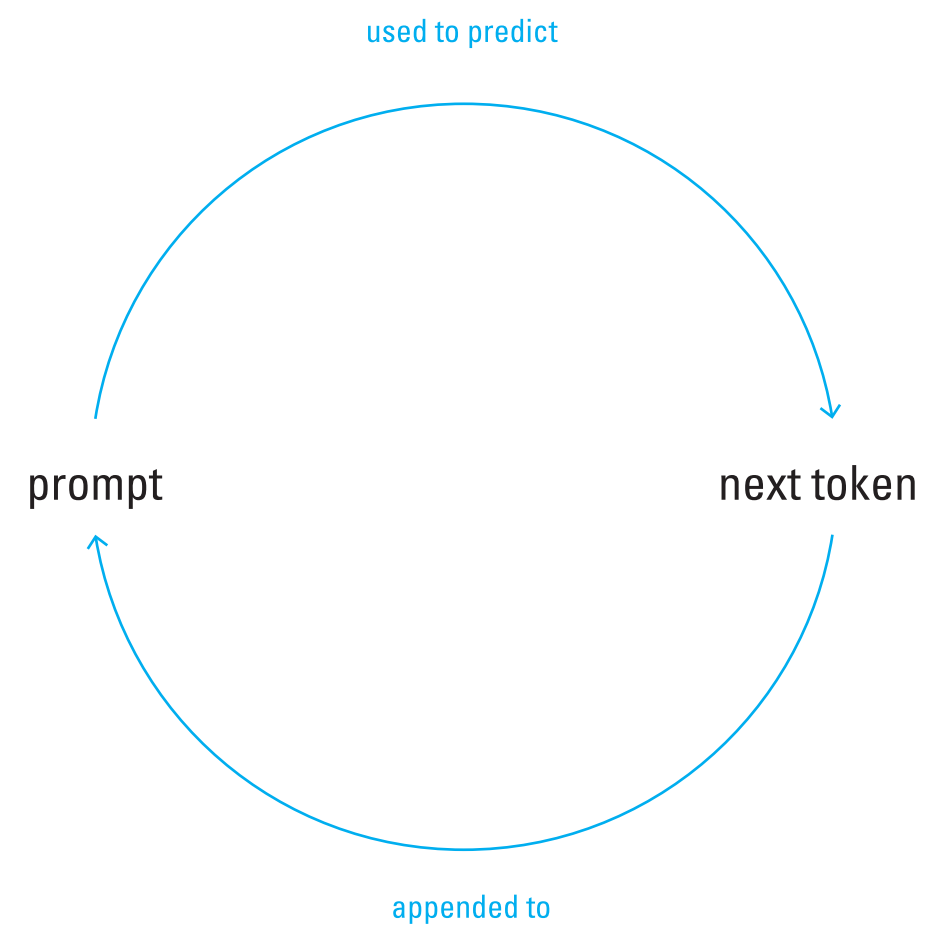
Once more through the cycle, and the model predicts the end token, ending the text generation, and leaving us with the final output.



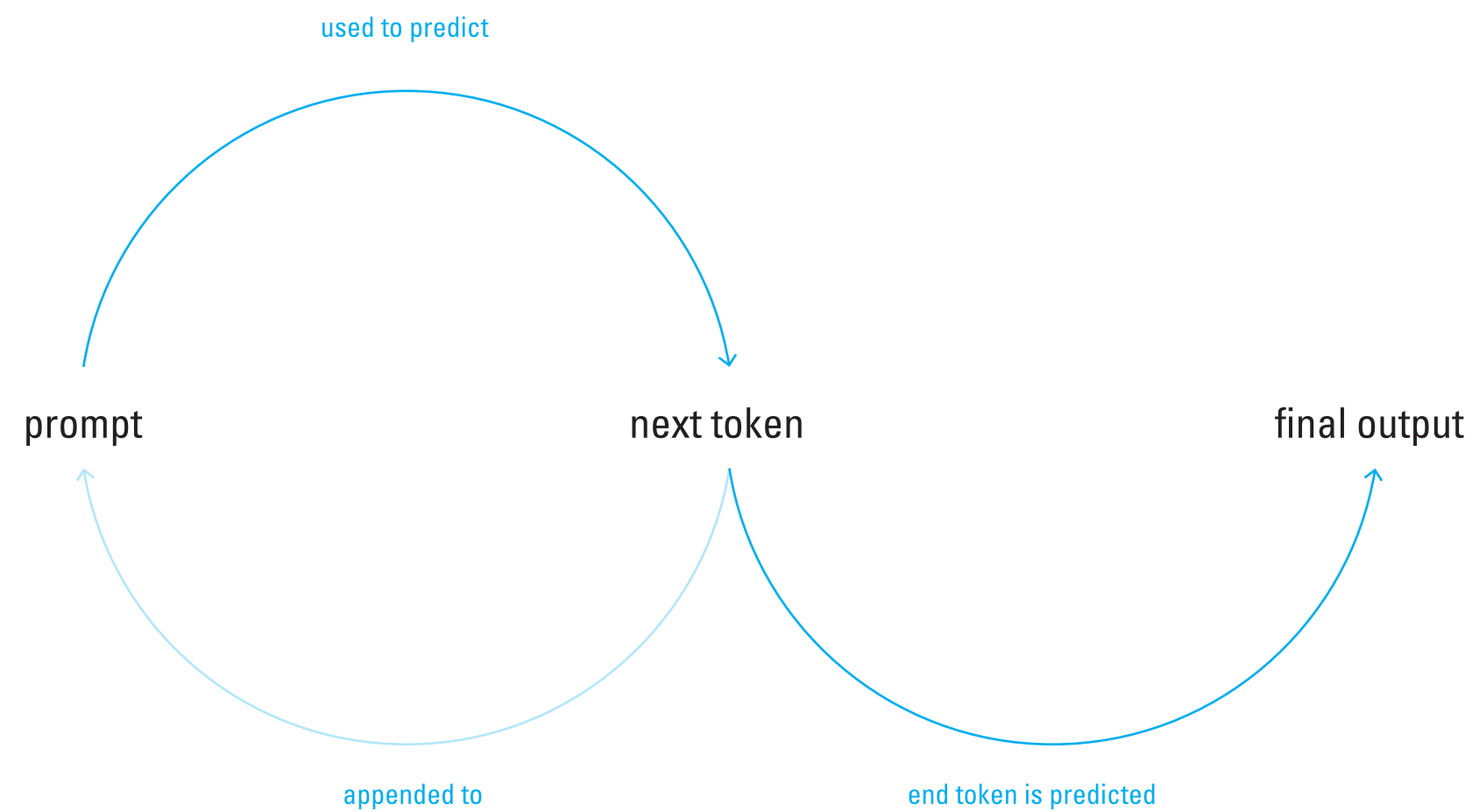
Simply put, the initial prompt is used to predict the next token.



The next token is appended to the prompt, and the cycle repeats.

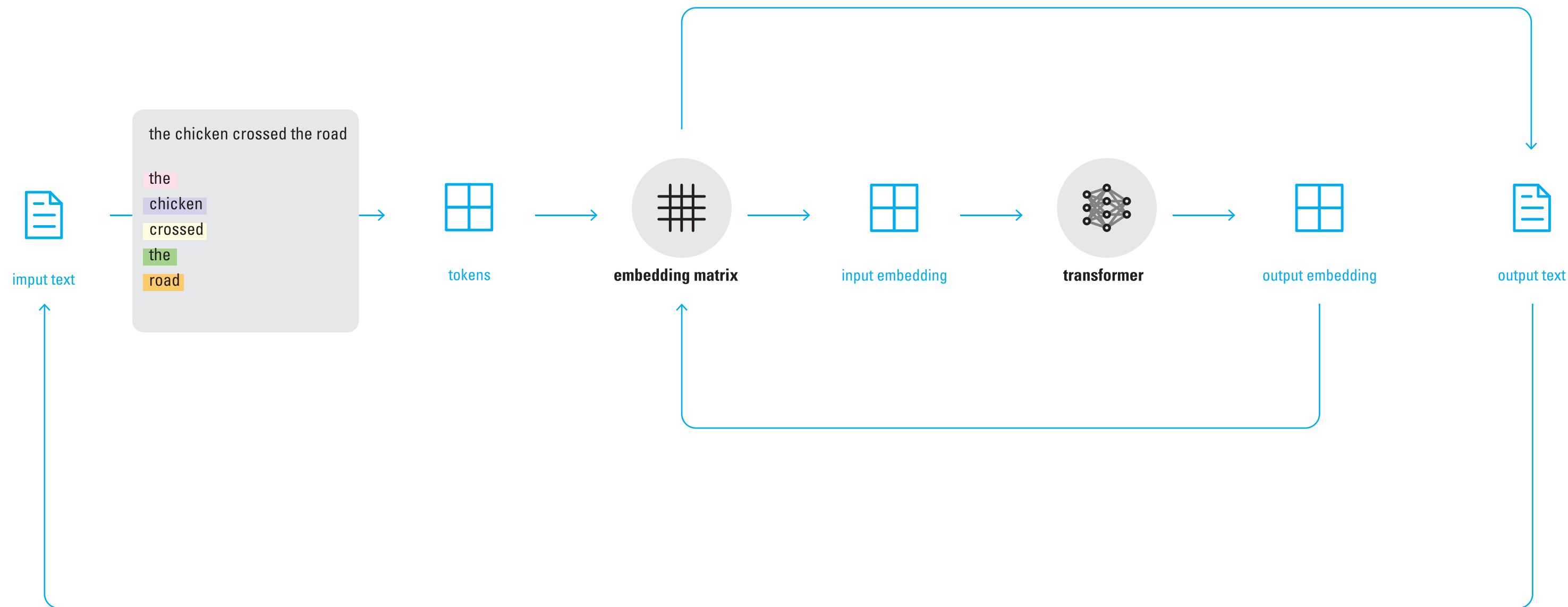


Until the end token is predicted, and the text generation stops.

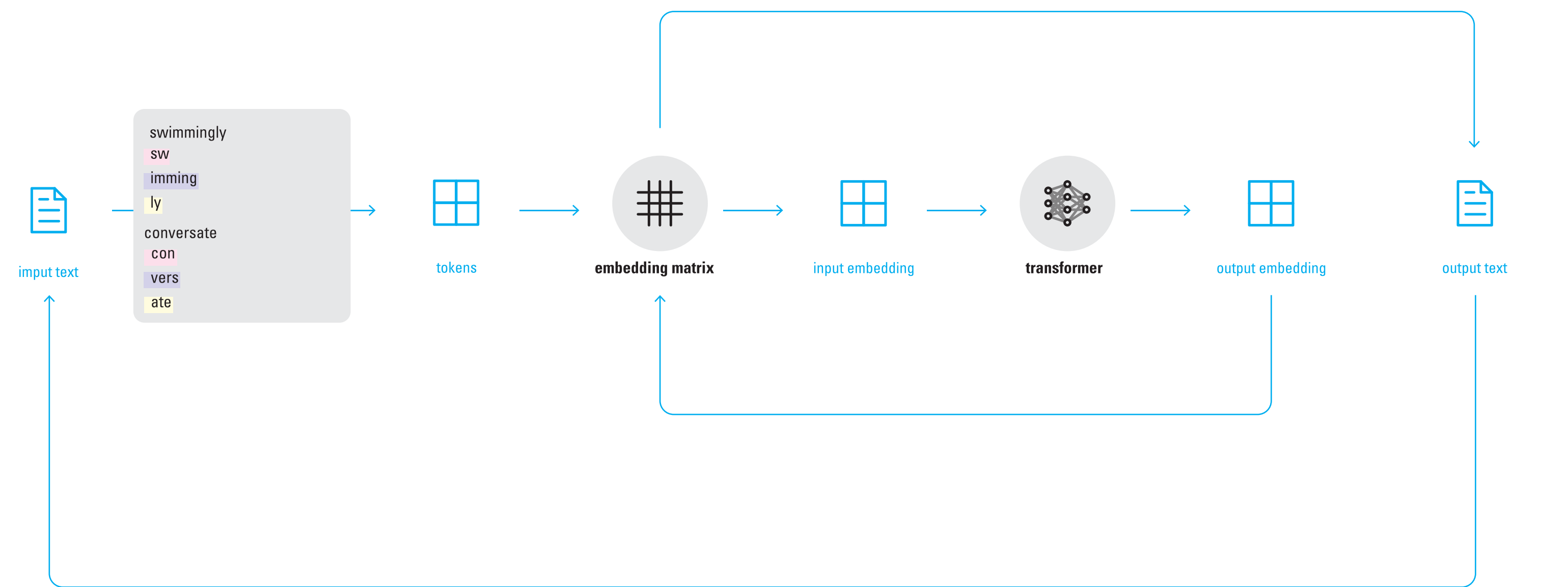


The tokenizer

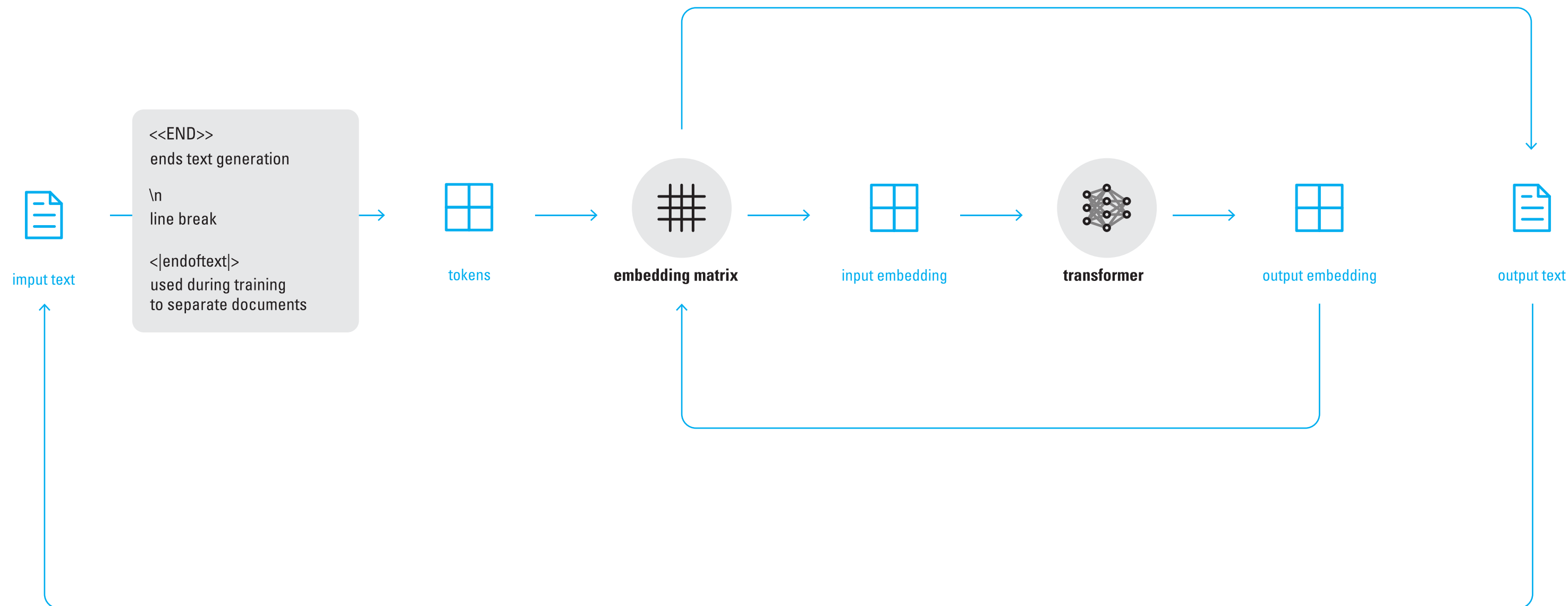
The tokenizer breaks up the input into individual tokens from the system's vocabulary. Often, each word is a token.



Longer words can also be made up of multiple tokens.



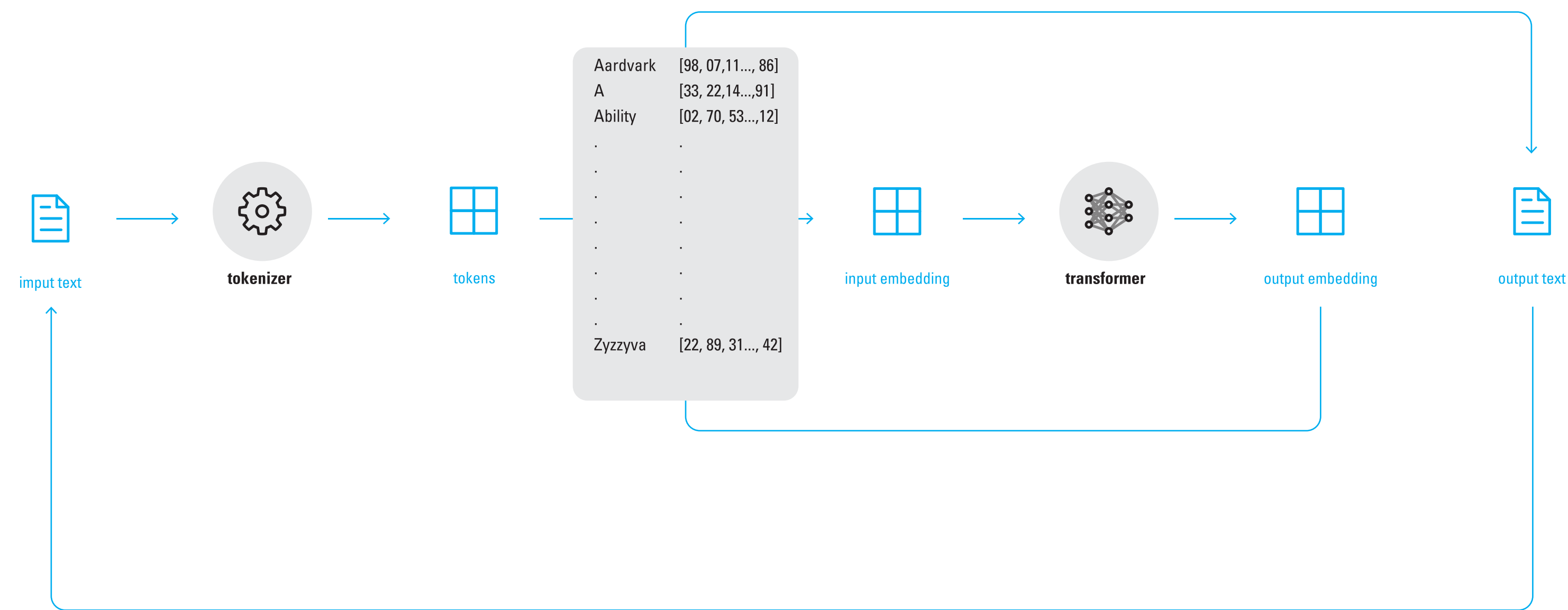
There are also special tokens, such as the end token to stop text generation, and the line break token, to start a new line.



The embedding matrix

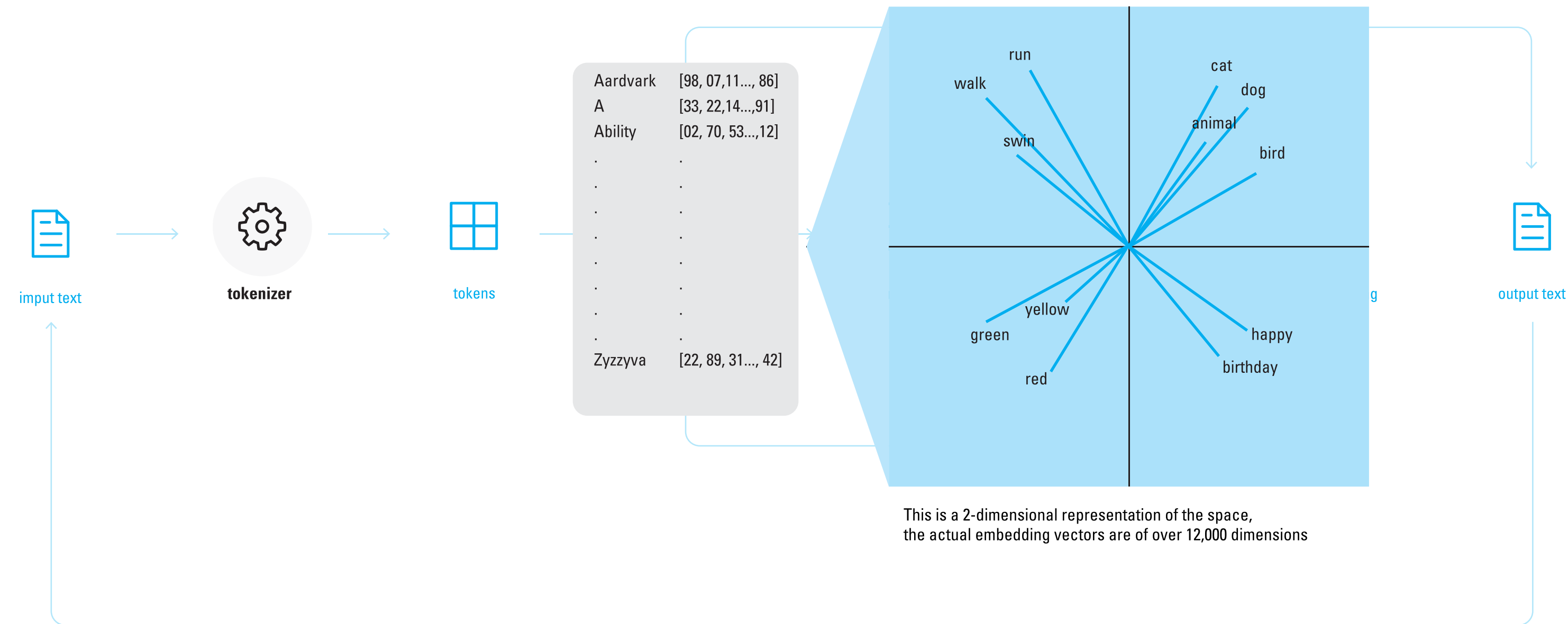
The embedding matrix a two column matrix.

The first column is every token in the system's vocabulary, and the second is each token's corresponding vector embedding.



The vector embedding describe a high dimensional meaning space.

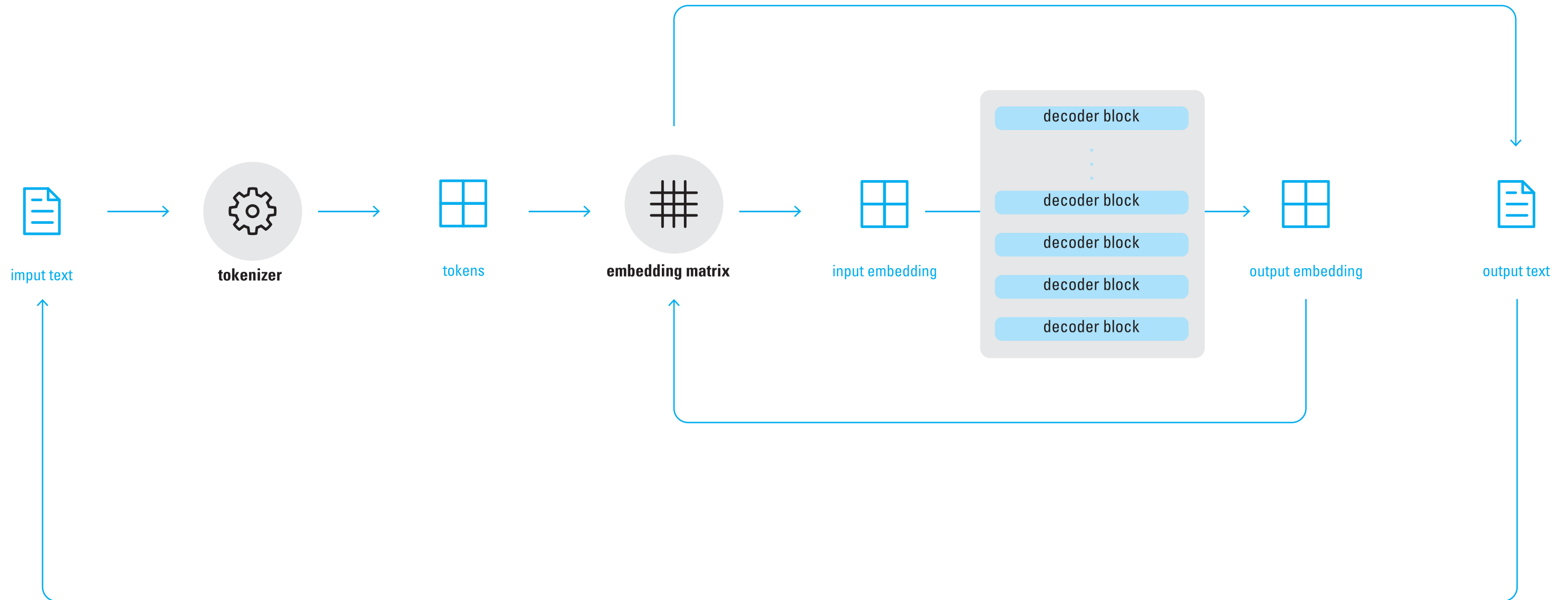
Each embedding is a set of coordinate that define a tokens meaning in relation to all other tokens



The transformer

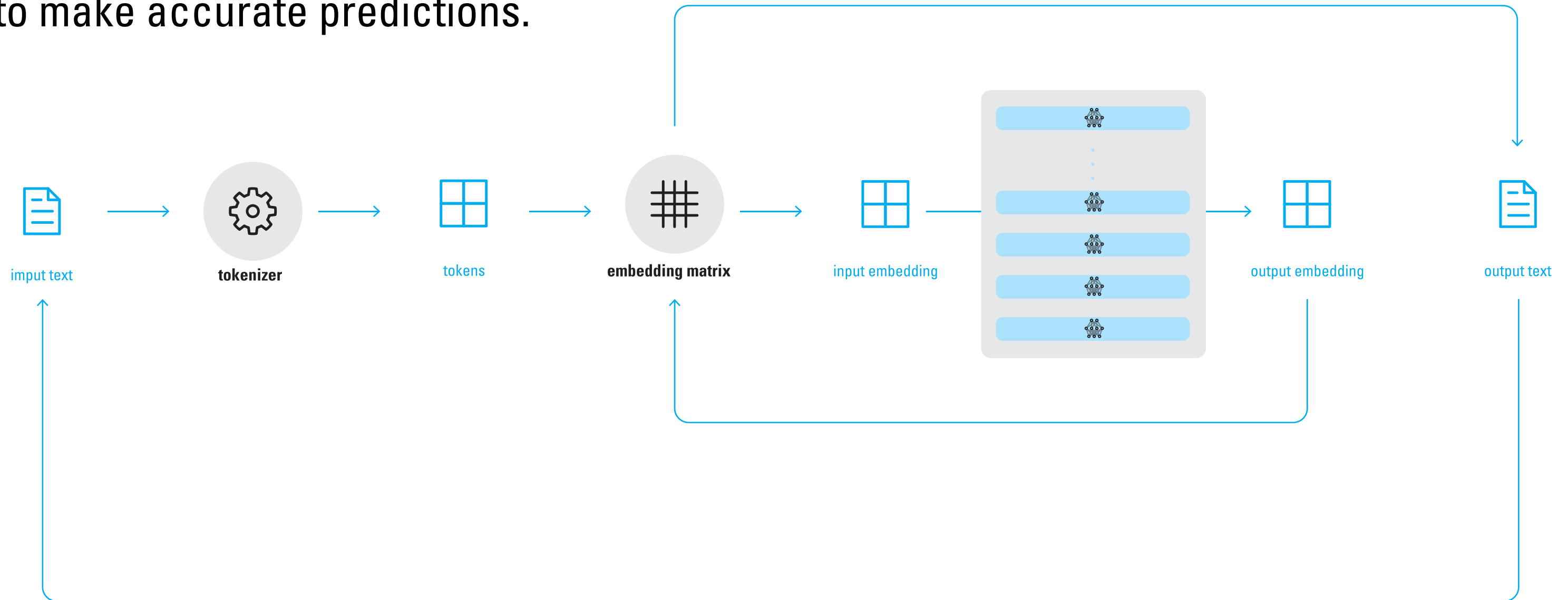
The transformer is a stack of decoder blocks.

Each block takes the input of the previous one and passes it up the chain to the next one. The final block outputs the prediction for the next token.



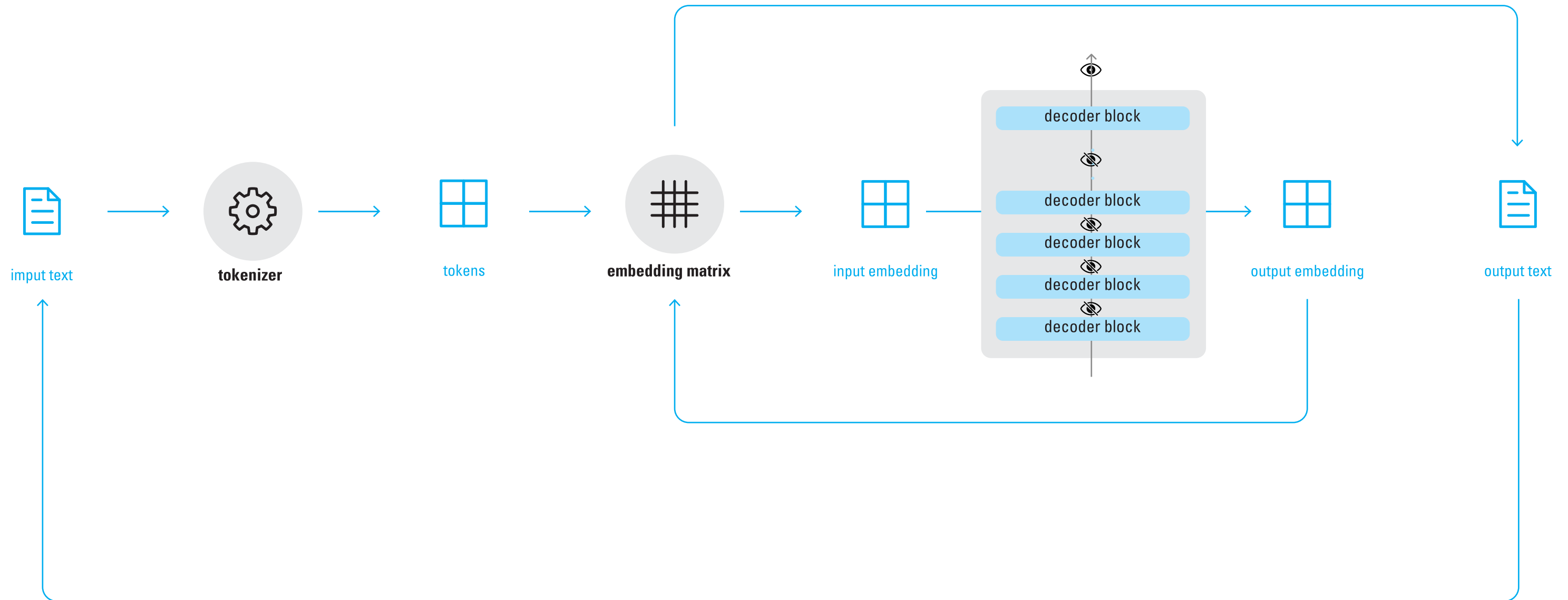
Each one of these layers contains a neural network with millions of parameters.

These parameters are initialized randomly,
and updated during training to allow the system
to make accurate predictions.



The decoder block architecture is the same for each block, but only the last output is used for the prediction.

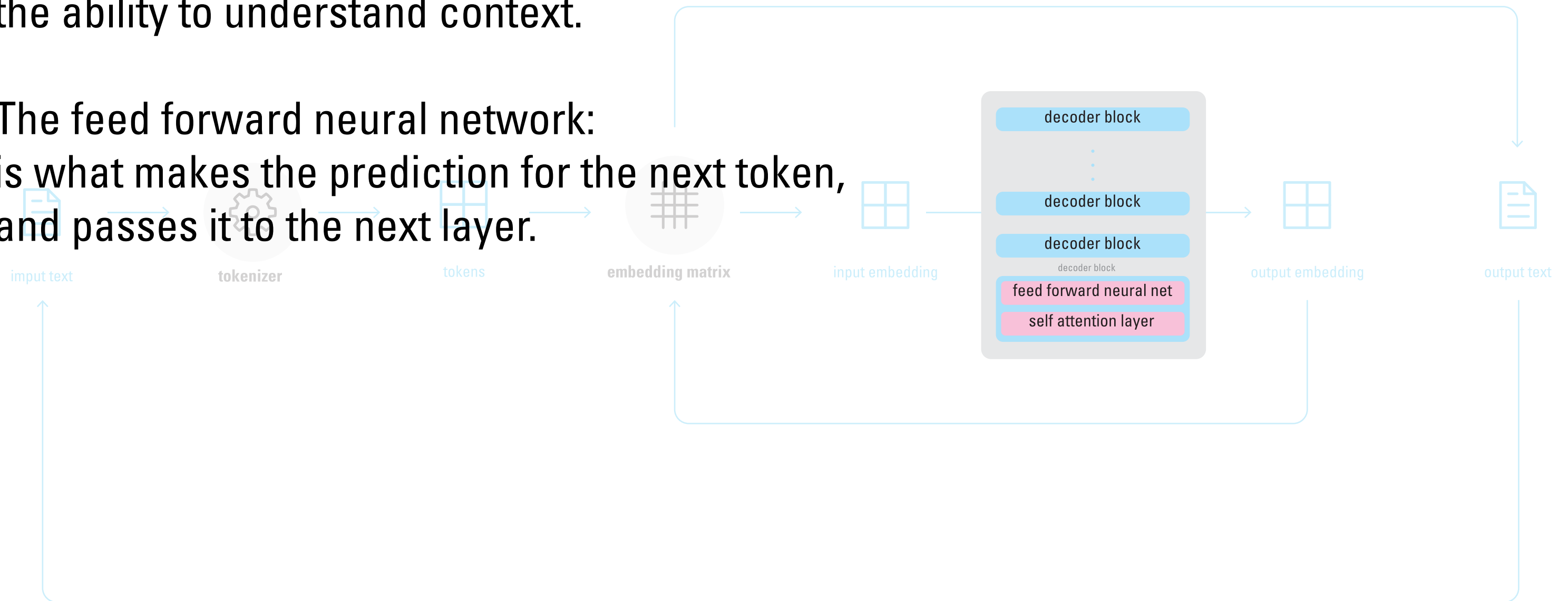
The middle decoder outputs are referred to as hidden states.



Each decoder block contains two sublayers.

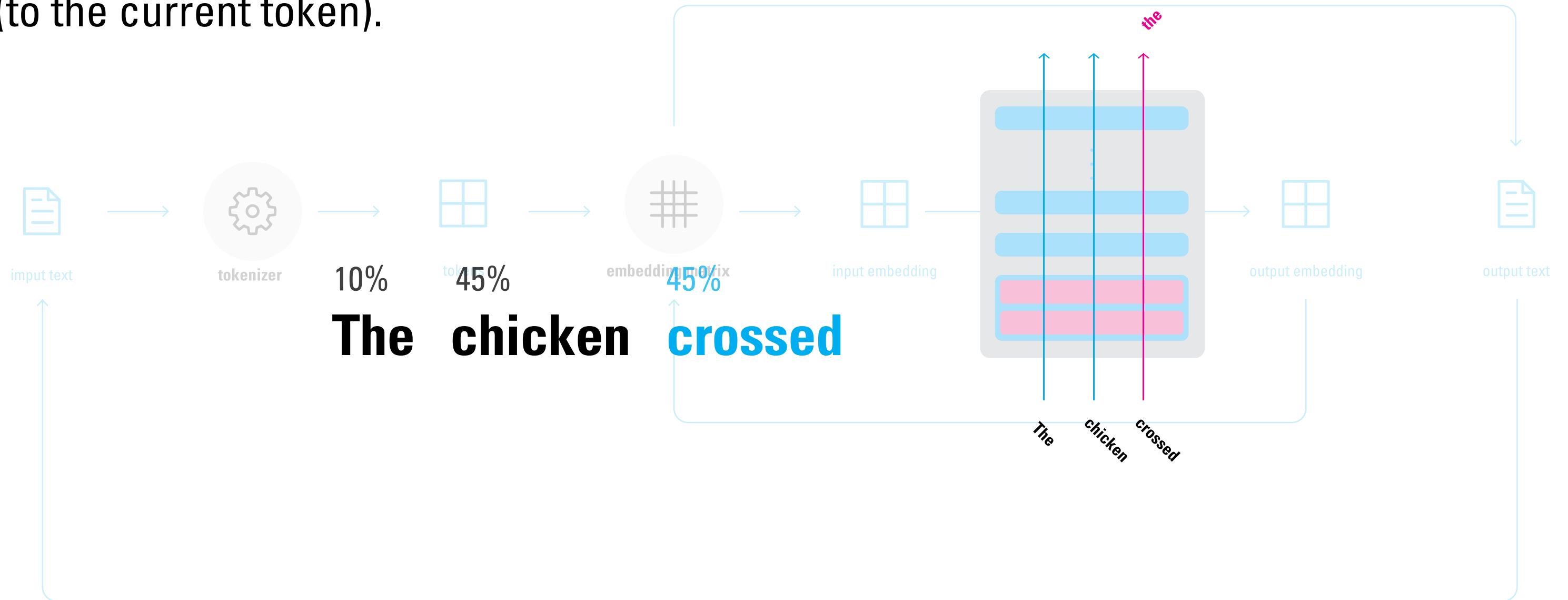
The self attention layer:
looks back on previous tokens, it is what gives the model
the ability to understand context.

The feed forward neural network:
is what makes the prediction for the next token,
and passes it to the next layer.



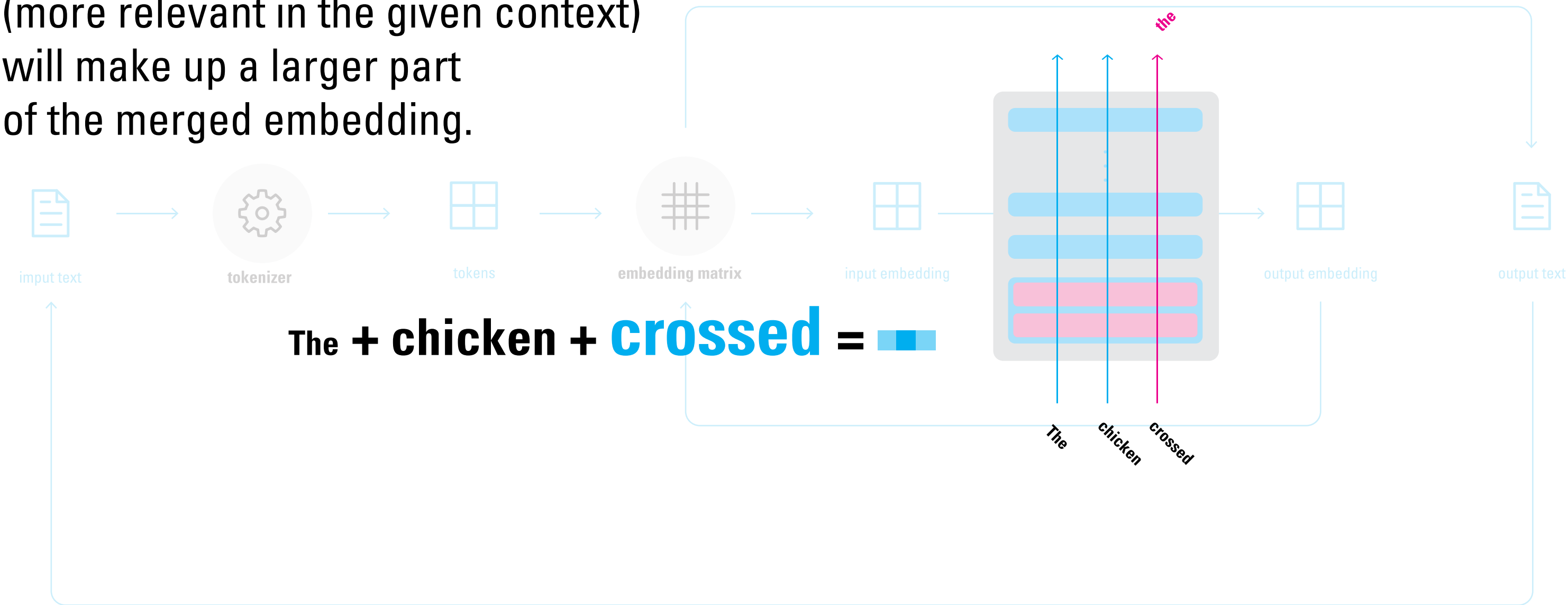
Self attention

For each track, all tokens to the left of the current token are weighted in terms of relevance (to the current token).

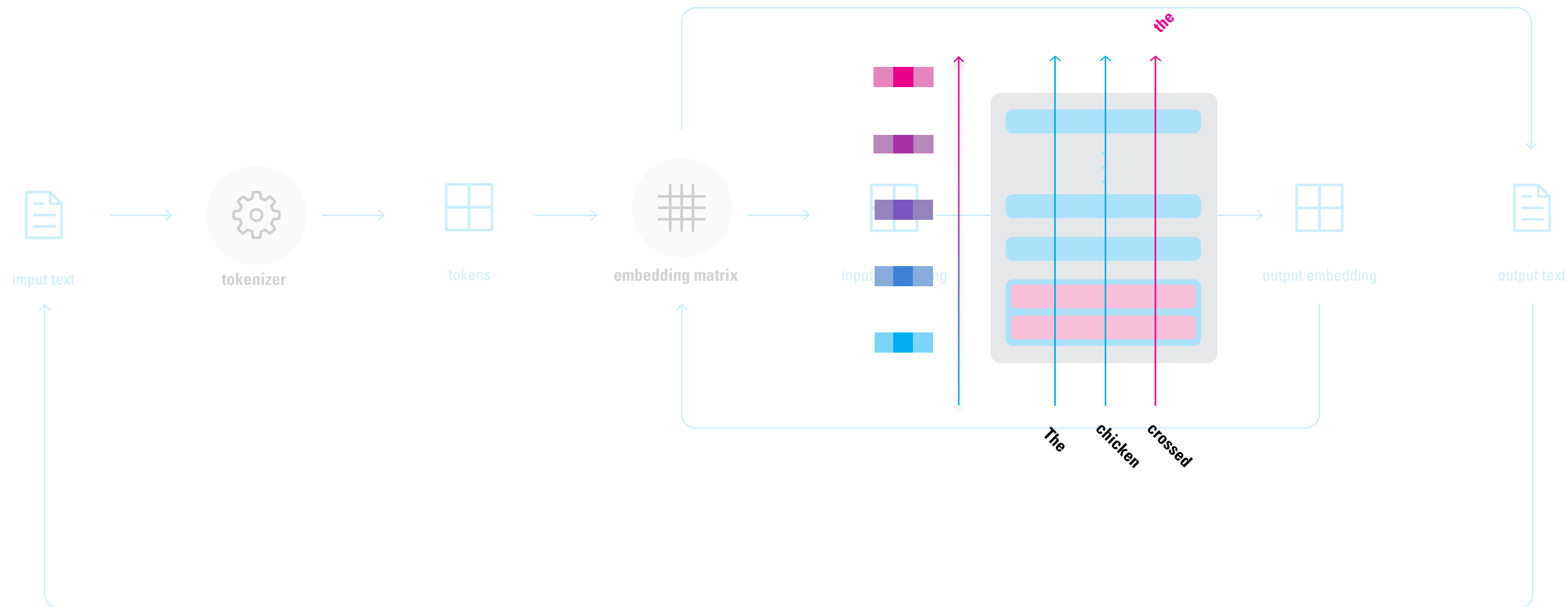


The token embeddings are multiplied by their weights, and concatenated to the current token embedding.

Tokens with greater weights (more relevant in the given context) will make up a larger part of the merged embedding.



**As the embeddings are passed up the stack,
the meaning within context is baked into the model.**



**This series of simple steps and calculations
creates the complex results we see from ChatGPT.**



Through the training and adjusting of its billions of parameters, GPT is able to display remarkable emergent behaviors.

Such as:

- writing a poem with a specified structure
- writing computer code in a specific language
- summarizing and writing essays
- emulating styles of writing
- making jokes
- editing its own writing

Appendix

Prompt:

can you prove infinite primes?

The statement that there are infinitely many prime numbers is a well-known result in mathematics and can be proven using a technique called Euclid's proof by contradiction. Here is a simplified explanation of the proof:

Assume that there are only finitely many prime numbers, and let's denote them as $p_1, p_2, p_3, \dots, p_n$ (where n is a finite number).

Consider the number $N = p_1 * p_2 * p_3 * \dots * p_n + 1$. This number N is larger than any of the primes $p_1, p_2, p_3, \dots, p_n$.

Now, either N is a prime number itself or it is composite. We consider both cases:

If N is a prime number, then it is a prime that is not in the original list of primes $p_1, p_2, p_3, \dots, p_n$. This contradicts the assumption that the list includes all prime numbers.

If N is composite, it must have prime factors. However, none of the primes $p_1, p_2, p_3, \dots, p_n$ can divide N without leaving a remainder (because N leaves a remainder of 1 when divided by any of the primes). Therefore, N must have at least one prime factor that is not in the original list of primes, which again contradicts the assumption.

Since the assumption leads to a contradiction in both cases, we can conclude that there must be infinitely many prime numbers.

This proof demonstrates that assuming a finite number of primes leads to a contradiction, implying that there must be an infinite number of primes. It's important to note that Euclid's proof provides evidence for the existence of infinitely many primes, but it does not provide an explicit list or formula for generating all prime numbers.

Prompt:

now can you prove infinite primes while rhyming?

Certainly! Here's a lighthearted rhyming proof
for the existence of infinitely many prime numbers:

In a realm of numbers, let's take a stroll,
Where primes reside, they play a role.
Assume finiteness, a limited few,
Primes so rare, but is it true?

Imagine a list, with primes in line,
 p_1, p_2, p_3 , and so on, they shine.
Now let's create a number, unique and neat,
 N equals one plus their product complete.

If N itself is prime, oh what a sight,
A new prime discovered, shining so bright!
It breaks our list, expands its span,
Another prime, not part of the plan!

But if N 's composite, its factors we seek,
None from our list, no primes they speak
A new prime must dwell, unknown to us,
Expanding our list, without a fuss.

In both scenarios, we're faced with a twist
A contradiction, the assumption dismissed.
Thus, we conclude with a joyous cheer,
Infinitely many primes are forever near!

So rejoice in numbers, infinite in flight,
Prime numbers dancing, a wondrous sight.
With Euclid's proof, we're bound to see,
Primes that stretch to eternity!