Let's jump in to talk about the self-attention mechanism of transformers. If you can get the main idea behind this video, you'll understand the most important core idea behind what makes transformer networks work. Let's jump in. You've seen how attention is used with sequential neural networks such as RNNs. To use attention with a style more like CNNs, you need to calculate self-attention, where you create attention-based representations for each of the words in your input sentence. Let's use our running example, Jane, visite, l'Afrique, en, septembre, our goal will be for each word to compute an attention-based representation like this. So we'll end up with five of these, since our sentence has five words. When we've computed them we'll call the five representations with these five words A1through A5. I know you're starting to see a bunch of symbols Q, K, and V, we'll explain what these symbols mean in a later slide so don't worry about them for now. The running example I'm going to use is take the word l'Afrique in this sentence. We'll step through on the next slide how the transformer network's self-attention mechanism allows you to compute A3 for this word, and then you do the same thing for the other words in the sentence as well. Now you learn previously about word embeddings. One way to represent l'Afrique would be to just look up the word embedding for l'Afrique. But depending on the context, are we thinking of l'Afrique or Africa as a site of historical interests or as a holiday destination, or as the world's second largest continent. Depending on how you're thinking of l'Afrique, you may choose to represent it differently, and that's what this representation A(3) will do.

It will look at the surrounding words to try to figure out what's actually going on in how we're talking about Africa in this sentence, and find the most appropriate representation for this.

In terms of the actual calculation, it won't be too different from the attention mechanism you saw previously as applied in the context of RNNs, except we'll compute these representations in parallel for all five words in a sentence.

When we're building attention on top of RNNs, this was the equation we used. With the self-attention mechanism, the attention equation is instead going to look like this. You can see the equations have some similarity. The inner term here also involves a softmax, just like this term over here on the left, and you can think of the exponent terms as being akin to attention values.

Exactly how these terms are worked out you'll see in the next slide. So again, don't worry about the details just yet. But the main difference is that for every word, say for l'Afrique, you have three values called the query, key, and value. These vectors are the key inputs to computing the attention value for each words. Now, let's step through the steps needed to actually calculate A3. On this slide, let's step through the computations you need to go from the words l'Afrique to the self-attention representation A3. For reference, I've also printed up here on the upper-right that softmax-like equation from the previous slide. First, we're going to associate each of the words with three values called the query key and value pairs. If X3 is the word embedding for l'Afrique, the way that's Q3 is computed is as a learned matrix, which I'm going to write as WQ times X3, and similarly for the key and value pairs, so K3 is WK times X3 and V3 is WV times X3. These matrices, WQ, WK, and WV, are parameters of this learning algorithm, and they allow you to pull off these query, key, and value vectors for each word. So what are these query key and value vectors supposed to do? They were indeed using a loose analogy to a concept and databases where you can have queries and also key-value pairs.

If you're familiar with those types of databases, the analogy may make sense to you, but if you're not familiar with that database concept, don't worry about it. Let me give one intuition behind the intent of these query, key, and value of vectors. Q3 is a question that you get to ask about l'Afrique. Q3 may represent a question like, what's happening there? Africa, l'Afrique is a destination. You might want to know when computing A^3, what's happening there. What we're going to do is compute the inner product between q^3 and k^1, between Query 3 and Key 1, and this will tell us how good is an answer where it's one to the question of what's happening in Africa. Then we compute the inner product between q^3 and k^2 and this is intended to tell us how good is visite an answer to the question of what's happening in Africa and so on for the other words in the sequence.

The goal of this operation is to pull up the most information that's needed to help us compute the most useful representation A^3 up here. Again, just for intuition building if k^1 represents that this word is a person, because Jane is a person, and k^2 represents that the second word, visite, is an action, then you may find that q^3 inter producted with k^2 has the largest value, and this may be intuitive example, might suggest that visite, gives you the most relevant contexts for what's happening in Africa. Which is that, it's viewed as a destination for a visit. What we will do is take these five values in this row and compute a Softmax over them. There's actually this Softmax over here, and in the example that we've been talking about, q^3 times k^2 corresponding to word visite maybe has the largest value. I'm going to shade that blue over here. Then finally, we're going to take these Softmax values and multiply them with v^1, which is the value for word 1, the value for word 2, and so on, and so these values correspond to that value up there. Finally, we sum it all up. This summation corresponds to this summation operator and so adding up all of these values gives you A^3, which is just equal to this value here. Another way to write A^3 is really as A, this A up here of q^3, k, v. But sometimes it will be more convenient to just write A^3 like that. The key advantage of this representation is the word of l'Afrique isn't some fixed word embedding. Instead, it lets the self-attention mechanism realize that l'Afrique is the destination of a visite, of a visit, and thus compute a richer, more useful representation for this word. Now, I've been using the third word, l'Afrique as a running example but you could use this process for all five words in your sequence to get similarly rich representations for Jane, visite, l'Afrique, en, septembre. If you put all of these five computations together, denotation used in literature looks like this, where you can summarize all of these computations that we just talked about for all the words in the sequence by writing Attention(Q, K, V) where Q, K, V matrices with all of these values, and this is just a compressed or vectorized representation of this equation up here. The term in the denominator is just to scale the dot-product, so it doesn't explode. You don't really need to worry about it. But another name for this type of attention is the scaled dot-product attention. This is the one represented in the original transformer architecture paper, Attention Is All You Need As Well.

That's the self-attention mechanism of the transformer network. To recap, associated with each of the five words you end up with a query, a key, and a value. The query lets you ask a question about that word, such as what's happening in Africa. The key looks at all of the other words, and by the similarity to the query, helps you figure out which words gives the most relevant answer to that question. In this case, visite is what's happening in Africa, someone's visiting Africa. Then finally, the value allows the representation to plug in how visite should be represented within A^3, within the representation of Africa. This allows you to come up with a representation for the word Africa that says this is Africa and someone is visiting Africa. This is a much more nuanced, much richer representation for the world than if you just had to pull up the same fixed word embedding for every single word without being able to adapt it based on what words are to the left and to the right of that word. We've all got to take into account and in the context. Now, you have learned about the self-attention mechanism. We're going to put a big four-loop over this whole thing and that will be the multi-headed attention mechanism. Let's dive into the details of that in the next video.