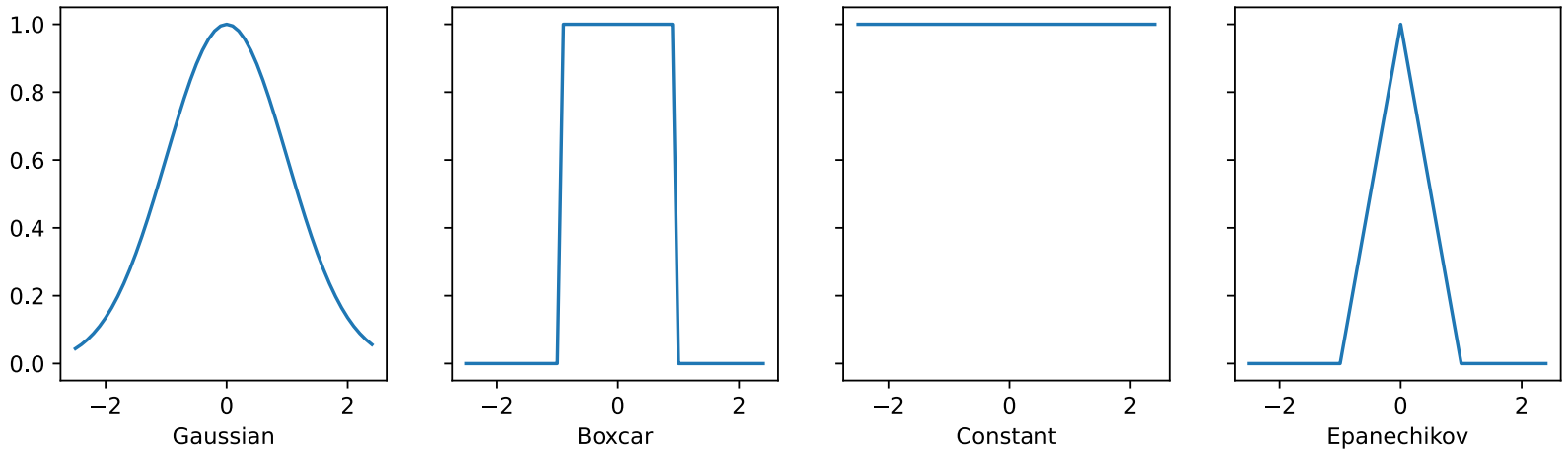
Transformers and Modern LLMs

Following D2L Chapter 11

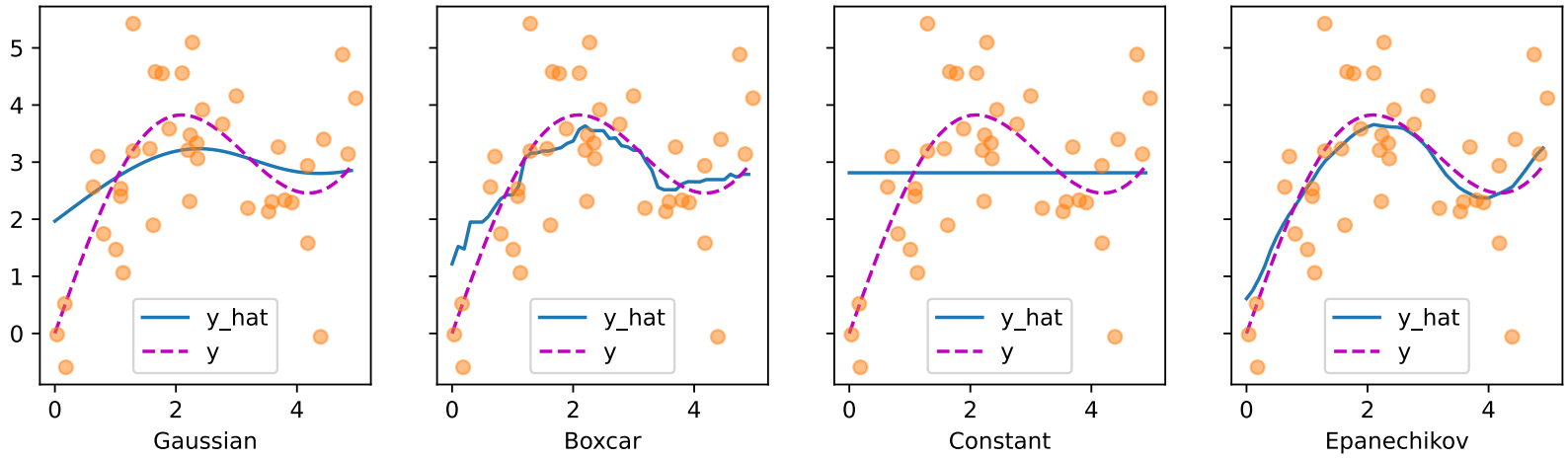
Kernel Density Estimation

A(nother) way to estimate non-linear regressions:

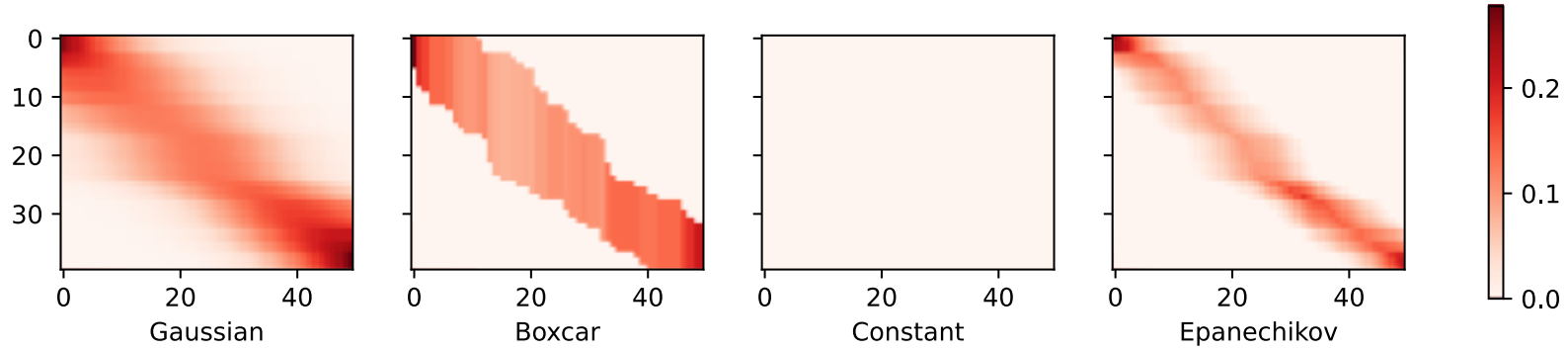
- Weight the observations based on their distance from the current location
- One version of this is LOWESS (local weighted sum of squares)



Regression Lines under KDE



How slope is estimated



So what?

While KDE methods are really great for semi-parametric models, some smart people (Vaswani et al, 2017) decided to generalize these concepts to even broader classes of models

- How can we selectively attend to different data points and inputs in a neural network?
- Can we make our model focus on more important words in a query, or critical characteristics in an image?

The answer was yes!

Defining Terms

$\mathcal{D}$: a database of key-value pairs

k: a key

v: a value

q: a query where we want to look up what the correct value is given a key

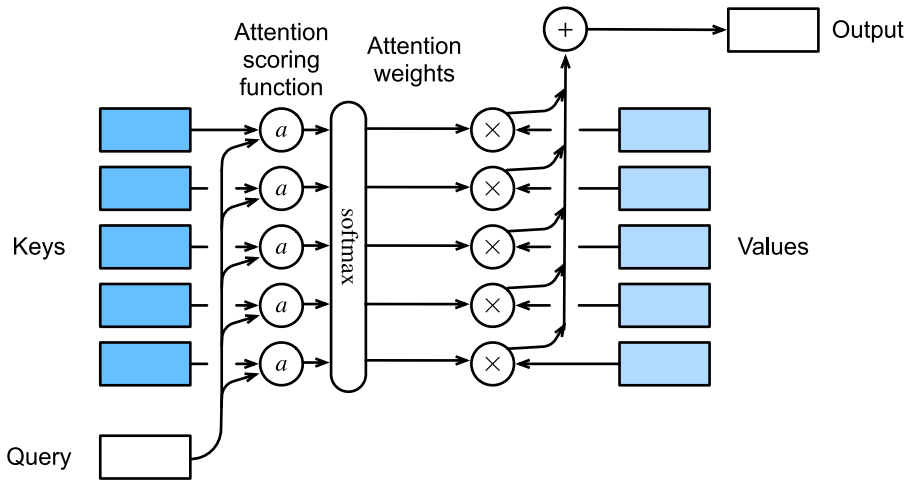
This is easy where all queries exist in $\mathcal{D}$, but what do we do when we have a query that doesn't exist?

Attention

$$\alpha(q, k_i) = softmax(a(q, k_i)) = \frac{exp(q^\intercal k_i / \sqrt{d})}{\sum_j exp(q^\intercal k_j / \sqrt{d})}$$

In English, we provide the input that our weights indicate is most applicable across our set of inputs as the relevant feature for our model to leverage at any particular point

Attention Diagram



Our weights help us to choose which pieces of information we pass forward through our network, and softmax makes us focus on a single key for each query in a given layer

Encoder and Decoder

Our attention models can have two parts

1. Encoder - Where we take our query and encode the information in it into our information set (embedding). Trained to **ingest** data
2. Decoder - Where we take our ingested data and use it to generate an output

So an attention model is actually TWO neural networks that are connected to each other: one encoder and one decoder

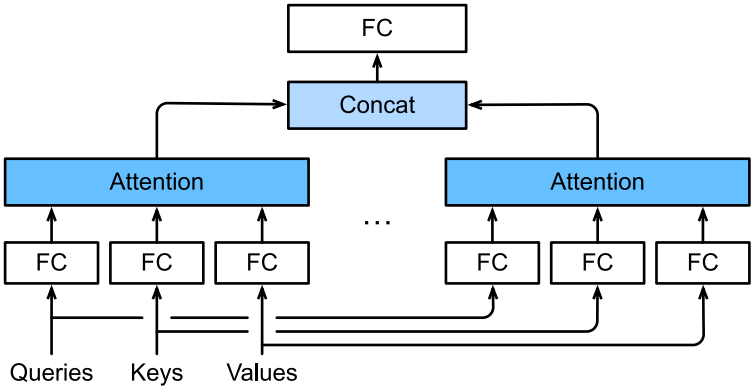
So what does Attention do?

Our model will get REALLY big if we try to make it memorize all of the input information in order to remember how to generate output information.

Instead, we teach our model how to pay attention to the important parts to streamline its calculations and memory requirements

Multi-head Attention

Just like our inception models, we can create parallel streams to model our inputs so that we can explore the data from different angles

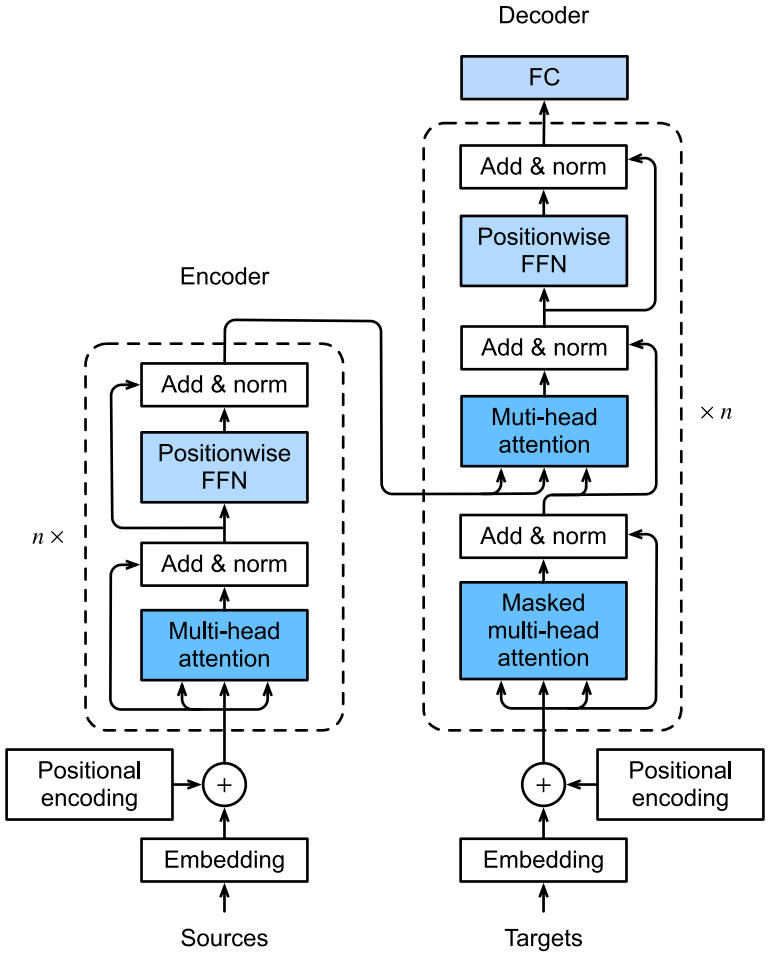


Positional Encoding

We also include information about WHERE a specific token appears in a sequence as **separate inputs**

- This allows us to parallel process our data rather than work sequentially!
- Reduces computation time drastically, and allows for much greater scaling of models than was previously possible

The Transformer



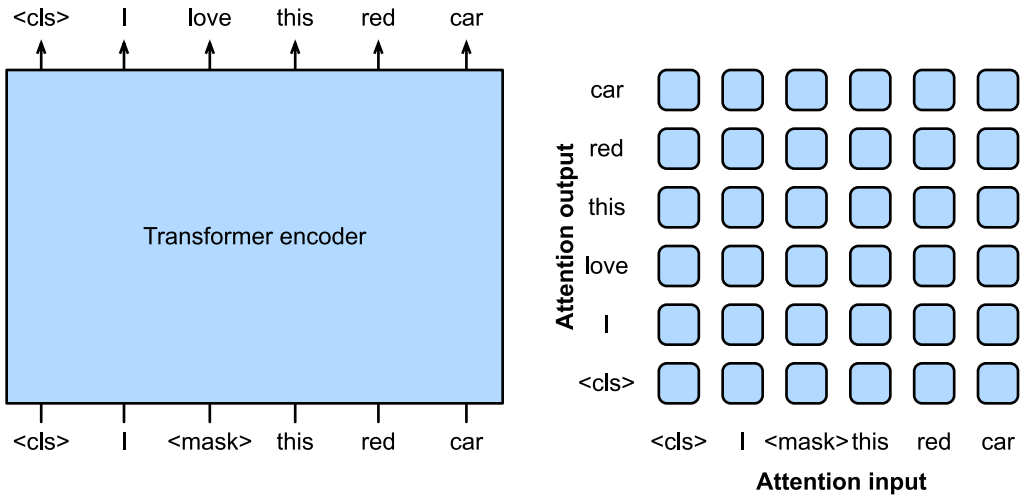
Let's Make One!

Follow the code here:

https://d2l.ai/chapter_attention-mechanisms-and-transformers/transformer.html

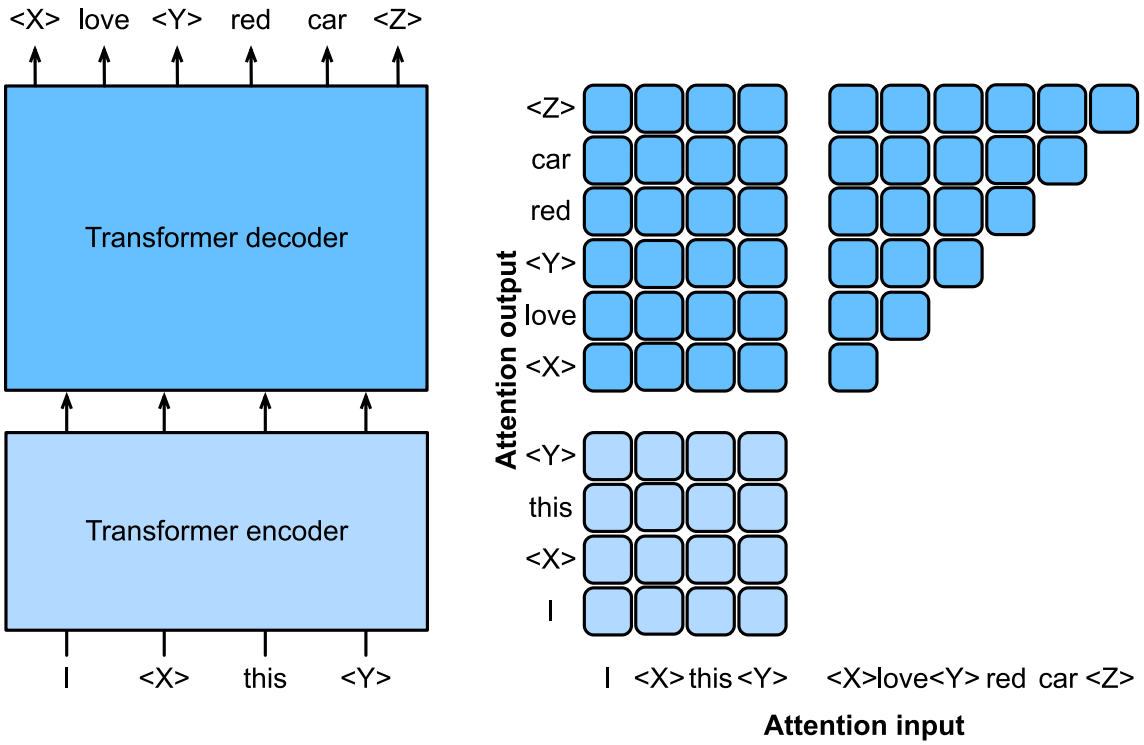
Modern Models - BERT

Encoder only!

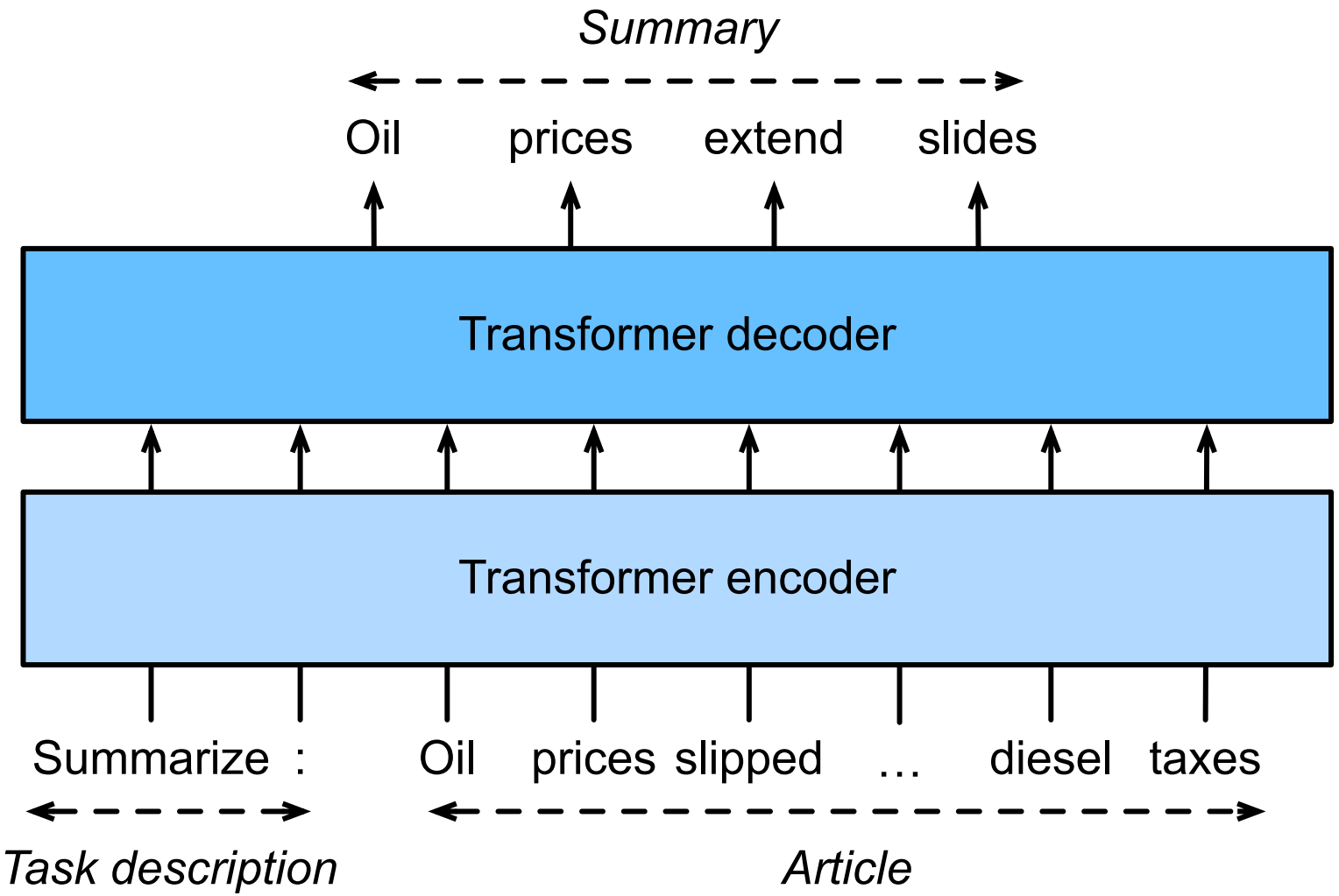


T5

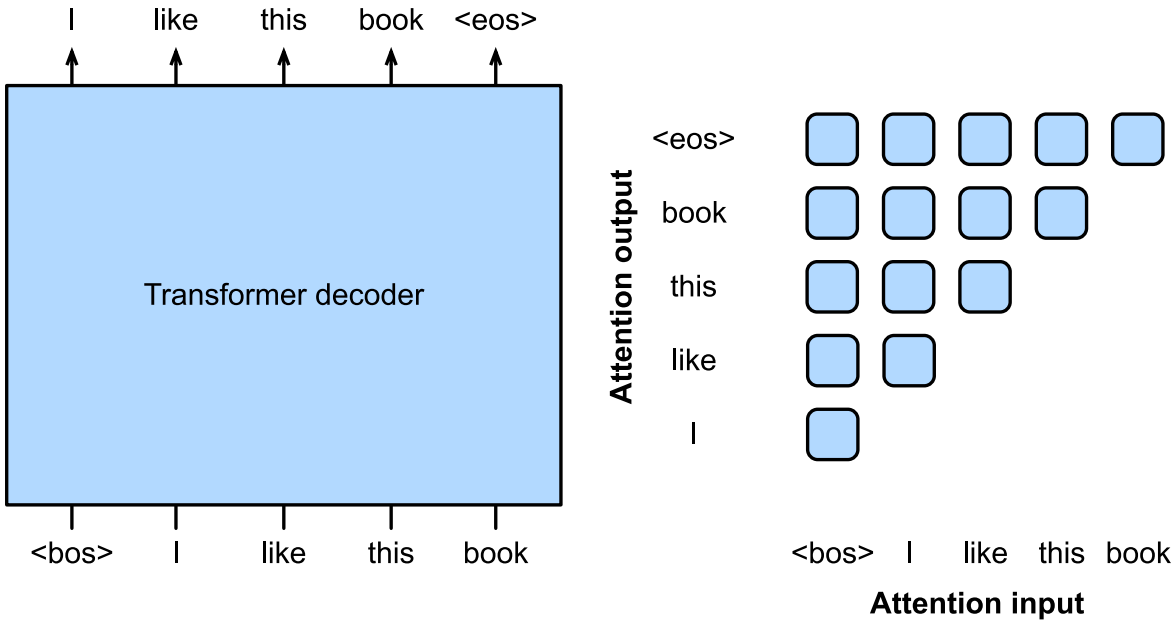
Encoder-Decoder



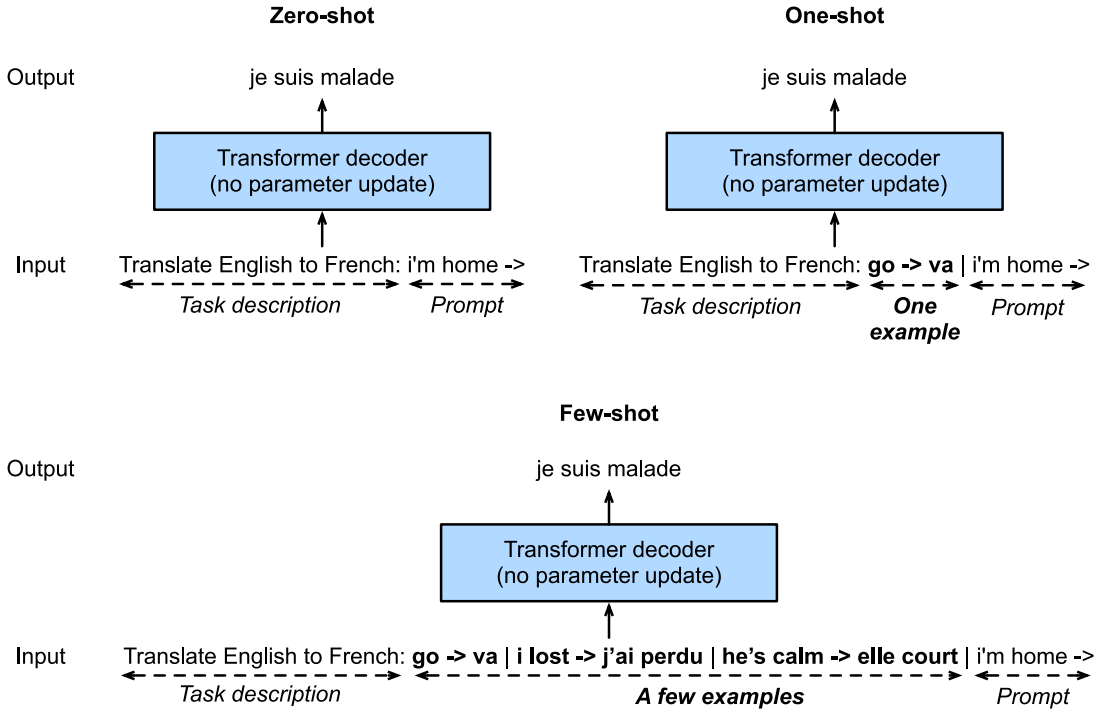
T5 Training


Summary
Oil
prices
extend
slides
Transformer decoder
Transformer encoder
Summarize
:
Oil
prices
slipped
…
diesel
taxes
Task description
Article

ChatGPT (1 and 2)



Training GPT3, and "shots"



Lab time!