EE 046746 - Technion - Computer Vision

Hila Manor

Tutorial 12 - Attention and Transformers

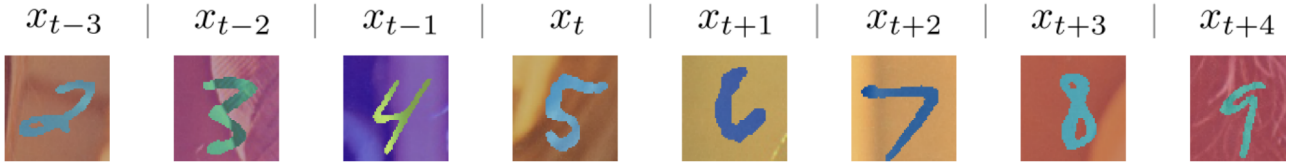
Prompt: university students learning about transformers and attention, you can clearly see a self-attention title in the slides

- Do it yourselves (https://stablediffusionweb.com/#demo/)

Agenda

Processing Sequences

- Many domains have data that is sequential in nature.
  - Natural language processing (NLP), speech and audio processing, financial data (such as stock prices), etc.
- In sequential data, each data point depends on the ones that come before it.



- So we want **context**.
  - A 5 that came after a 4, which came after a 3, and so on...
  - But how long should our context be?

- It turns out that "forever" is too long of a time period (Go figure).

Do we actually need "forever?"

- **Definition of Computer Vision** from Wikipedia:
  - Computer vision is an interdisciplinary field that deals with how computers can be made to gain high-level understanding from digital images or videos . From the perspective of engineering, it seeks to automate tasks that the human visual system can do. "Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. It involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding." As a scientific discipline, computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as video sequences, views from multiple cameras, or multi-dimensional data from a medical scanner. As a technological discipline, computer vision seeks to apply its theories and models for the construction of computer vision systems.

Attention

- Lets teach a model to **pay attention** to the more important features and understand the relationships between them.
- An example:
  - Task: Translate the English sentence "I am a student" to French.
  - Let's assume we have a dictionary where each English word (a **key**) has a direct translation to French (a **value**)

Keys	Values
student	étudiant
I	je
...	...
painting	tableau

  - To translate the English query: "I am a student", we can look up the translations for each word (a **query**) in the dictionary.
- Will every key have a unique translation?
  - The correct translation is "Je suis étudiant". Only 3 words!
  - What is the role of the word "a" in this context?
- Maybe the translation depends on more than just one word?
  - I am a female student -> Je suis étudiante

- Formally:
  - **Query** - The input we now want to search in our database: $\mathbf{q}_i \in \mathbb{R}^{d_q}$
    - $i \in \{1, \ldots, n\}$, where n is the length of our query.
  - **Key** - The entries in our database: $\mathbf{k}_j \in \mathbb{R}^{d_k}$
    - They're from the same "family" so usually $d_q = d_k$
  - **Value** - A data-point in the destination domain we're searching for- $\mathbf{v}_j \in \mathbb{R}^{d_v}$
    - $j \in \{1, \ldots, m\}$, where m is the size of our database.
- To query our database, we need to find the key in the dictionary that is the most similar to our query.
  - The most common similarity function is the scaled dot product:

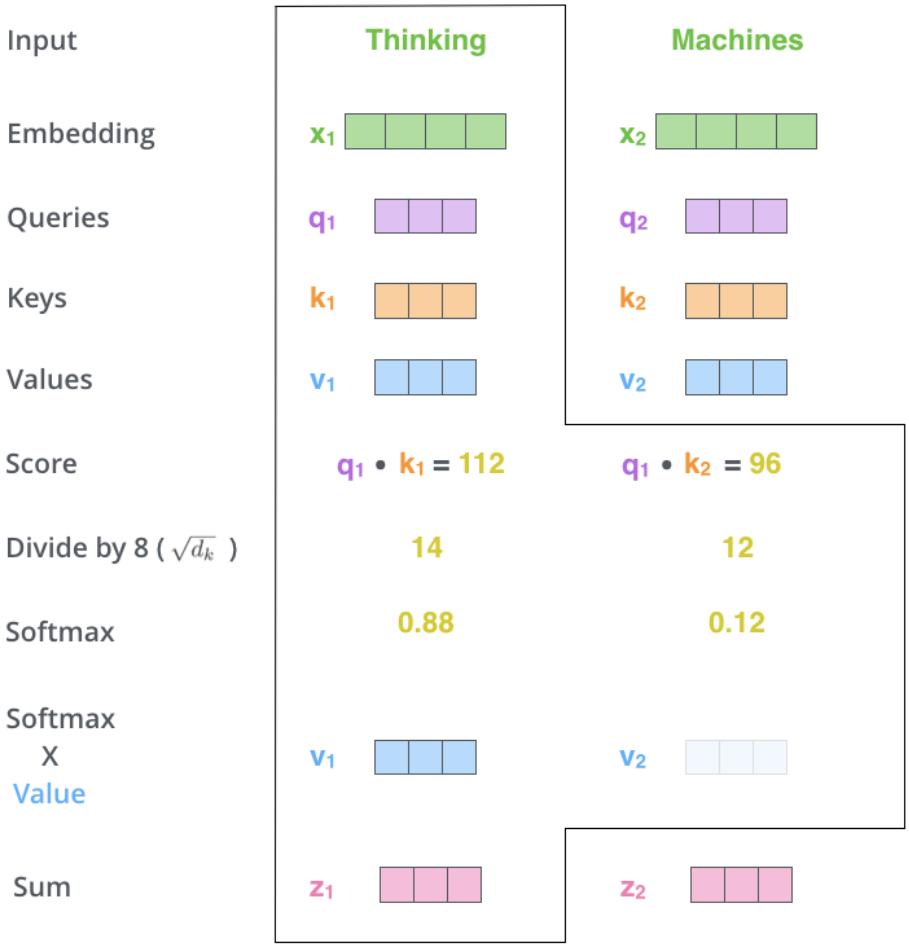
$$similarity(\mathbf{q}_i, \mathbf{k}_j) = \frac{\mathbf{q}_i^T \mathbf{k}_j}{\sqrt{d_k}}$$

- Then we weigh the result according to how simliar the key was to the original query:

$$Attention(\mathbf{q}_i, \{\mathbf{k}_j\}_{j=1}^m, \{\mathbf{v}_j\}_{j=1}^m) = \sum_j similarity(\mathbf{q}_i, \mathbf{k}_j) \cdot \mathbf{v}_j = \sum_j \frac{\mathbf{q}_i^T \mathbf{k}_j}{\sqrt{d_k}} \cdot \mathbf{v}_j$$

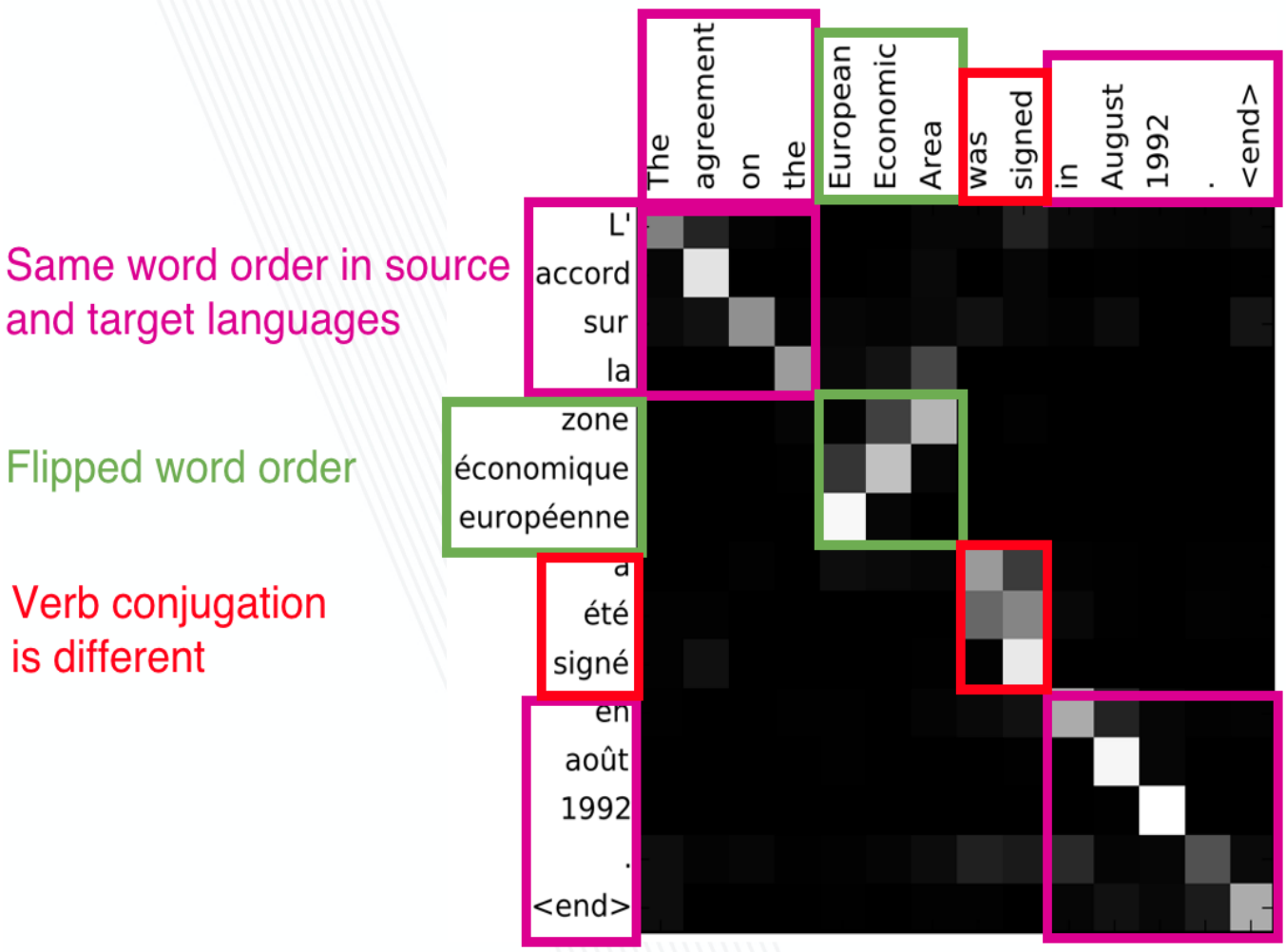
- We can stack keys-vectors on top of each other ($\mathbf{K} \in \mathbb{R}^{d_k \times m}$) to get all of the similarity values at once for a single query: $\frac{\mathbf{q}_i^T \mathbf{K}}{\sqrt{d_k}}$
  - We can add a softmax on top to normalize it: $softmax\left(\frac{\mathbf{q}_i^T \mathbf{K}}{\sqrt{d_k}}\right)$
- We can stack values similarly ($\mathbf{V} \in \mathbb{R}^{d_v \times m}$) to weigh all of the values at once: $softmax\left(\frac{\mathbf{q}_i^T \mathbf{K}}{\sqrt{d_k}}\right)\mathbf{V}$
- And we can also stack the queries ($\mathbf{Q} \in \mathbb{R}^{d_q \times n}$) to query everything at once:

$$Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax\left(\frac{\mathbf{Q}^T \mathbf{K}}{\sqrt{d_k}}\right)\mathbf{V}$$



- Image Source (https://jalammar.github.io/illustrated-transformer/)

Attention Weights Matrix Visualization



- Image Source: D. Bahdanau et al (https://arxiv.org/pdf/1409.0473.pdf)

A famous visualization example of attention from an image-captioning paper (Xu et al. 2015 (http://proceedings.mlr.press/v37/xuc15.pdf)):


A woman is throwing a frisbee in a park.


A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.


A little girl sitting on a bed with a teddy bear.


A group of people sitting on a boat in the water.

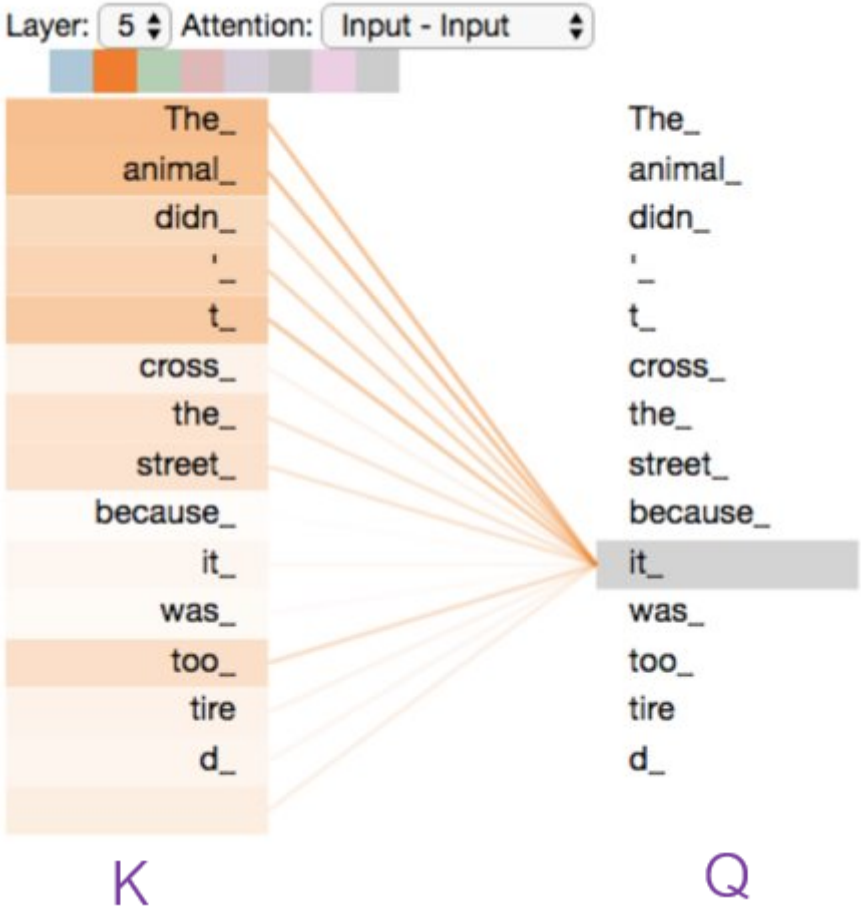

A giraffe standing in a forest with trees in the background.

- Image Source: K. Xu et al. (http://proceedings.mlr.press/v37/xuc15.html)

Self-Attention

- What if we want to find relationships between elements in the same input?
  - If it looks like a car, sounds like a car, but is on water... Then it's a boat.

- in self-attention we only have one domain.
- So we use the same "query" input for the keys as well
  - We query it against itself.
- And the destination domain also uses the same input (from English to English, to English again).



Who are Q, K, V?

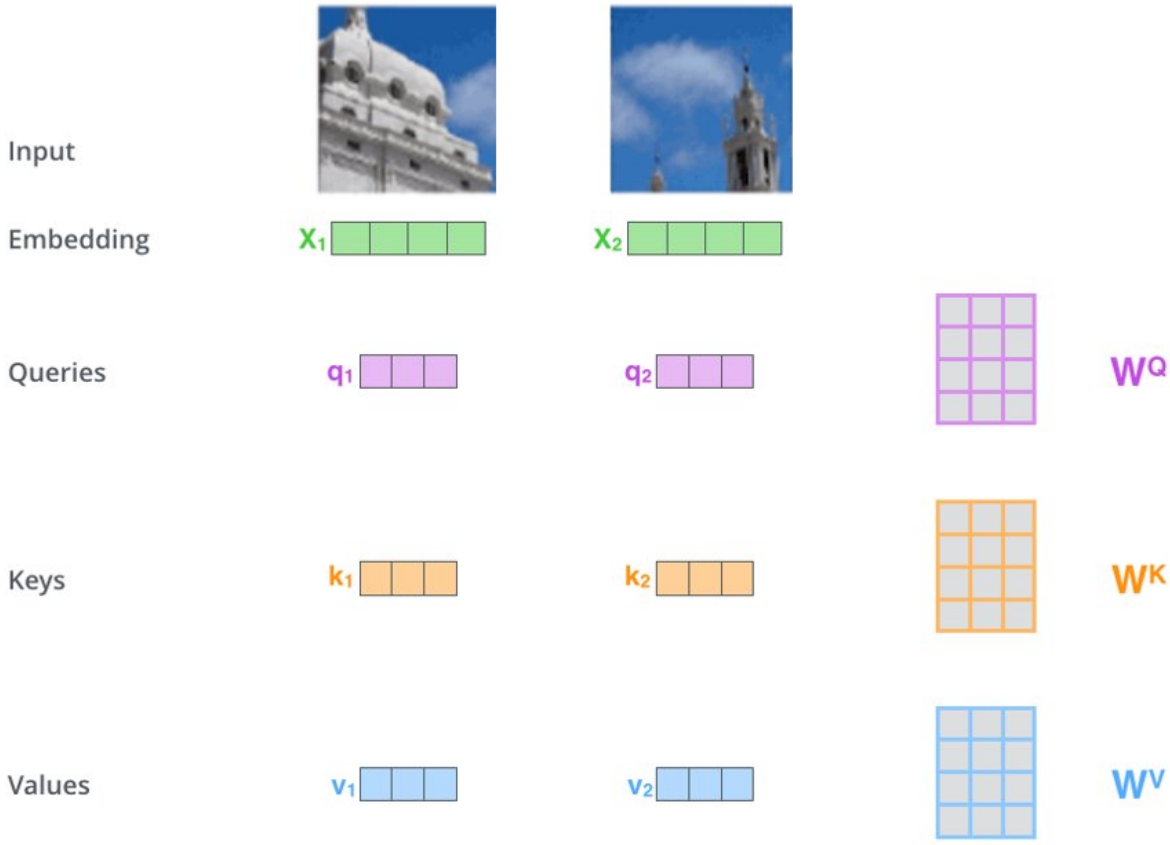
- How did we get from a word to these vectors?
  - How do we get from **an image** to these vectors?
- Cut the image to 16x16 patches, and *embed* each patch using an linear projection head (MLP).
  - Add some form of positional encoding, so we will have some sense of "where" each patch was.

- Now to create $\mathbf{Q}, \mathbf{K}, \mathbf{V}$, we multiply the embedded input $\mathbf{x}_i \in \mathbb{R}^{d_{emb}}$ y **learnable** matrices: $\mathbf{W_Q} \in \mathbb{R}^{d_q \times d_{emb}}, \mathbf{W_K} \in \mathbb{R}^{d_k \times d_{emb}}, \mathbf{W_V} \in \mathbb{R}^{d_v \times d_{emb}}$

$$\mathbf{q}_i = \mathbf{W_Q}\mathbf{x}_i$$

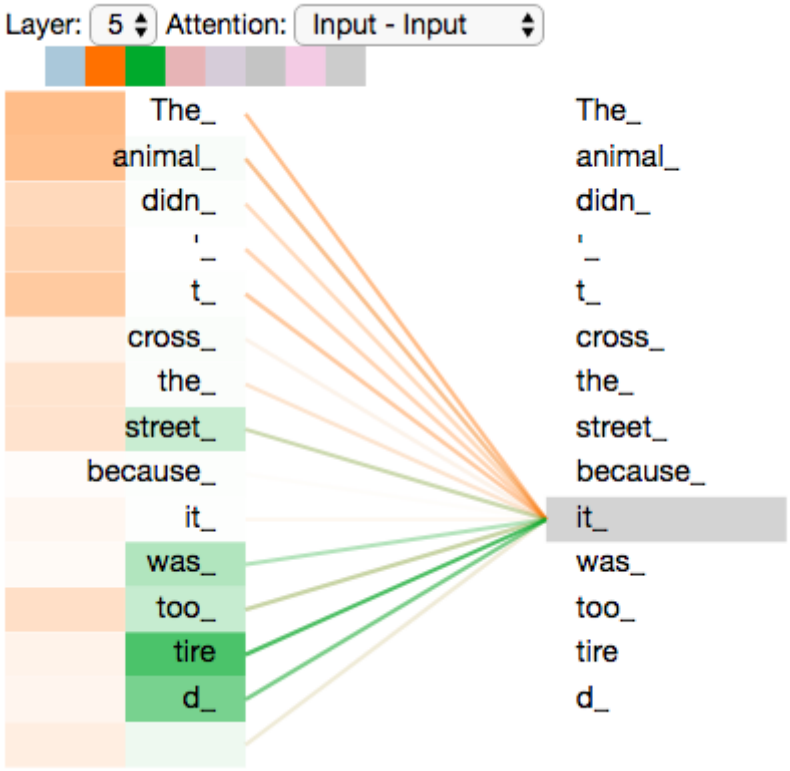
- In self-attention, the same input is used for the keys and values, remember?

$$\mathbf{k}_i = \mathbf{W_K}\mathbf{x}_i$$
$$\mathbf{v}_i = \mathbf{W_V}\mathbf{x}_i$$



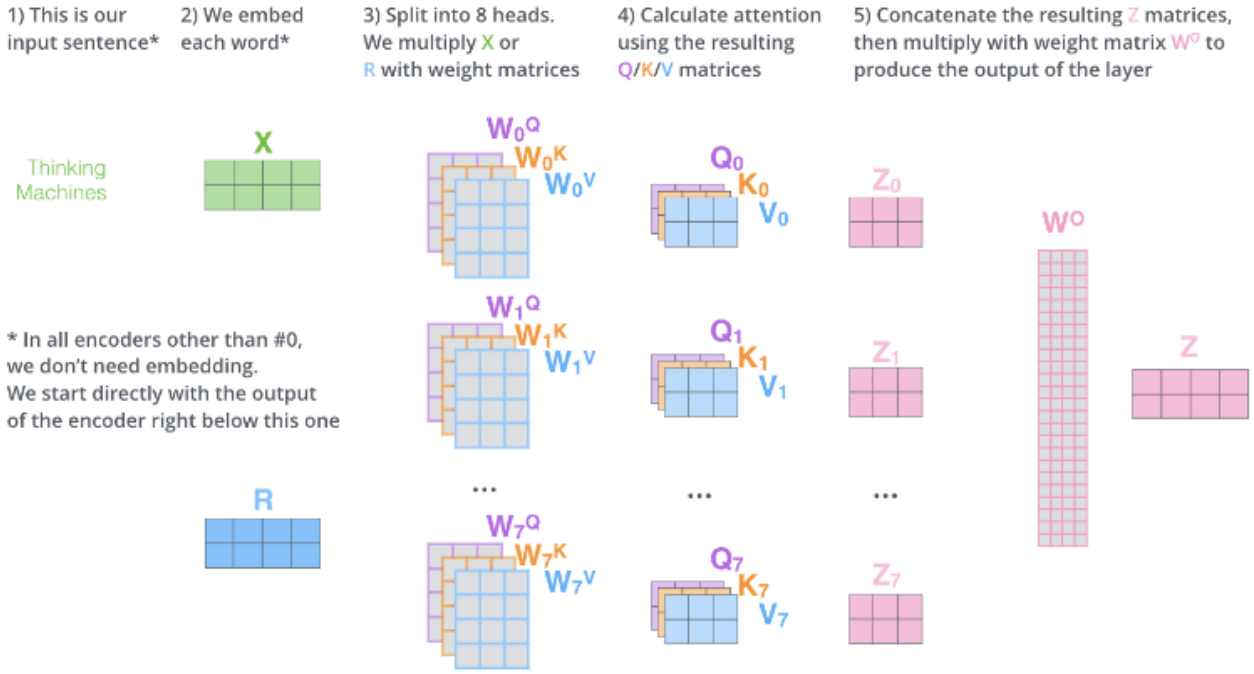
Multiple Heads

- Can we pay attention to different stuff for the same input?



- [Tensor2Tensor Notebook (https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/t

- Learn multiple $\mathbf{W_Q}$ $\mathbf{W_K}$, $\mathbf{W_V}$ matrices, and combine the results from the different "heads".



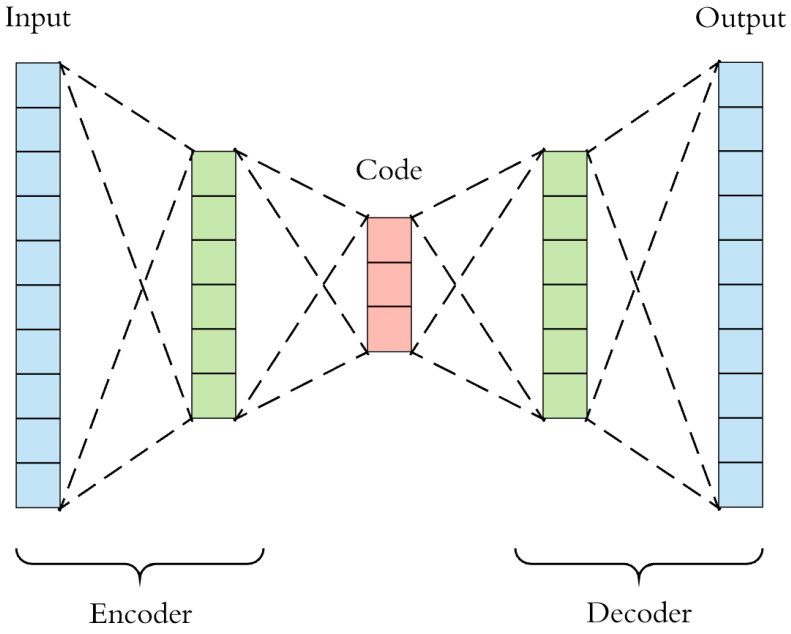
The Transformer

- The attention scheme received a lot of attention thanks to the paper [Attention Is All You Need (https://arxiv.org/abs/1706.03762)](https://arxiv.org/abs/1706.03762) (+61642 citations).
  - Before it, sequential data was processed... Sequentially.
    - Can't maximize the utilisation of the GPU!
  - Using attention, the data can be processed **all-at-once**.
    - Using the parallelization capabilities of GPUs.
- The transformer is composed of an **Encoder** and a **Decoder**.

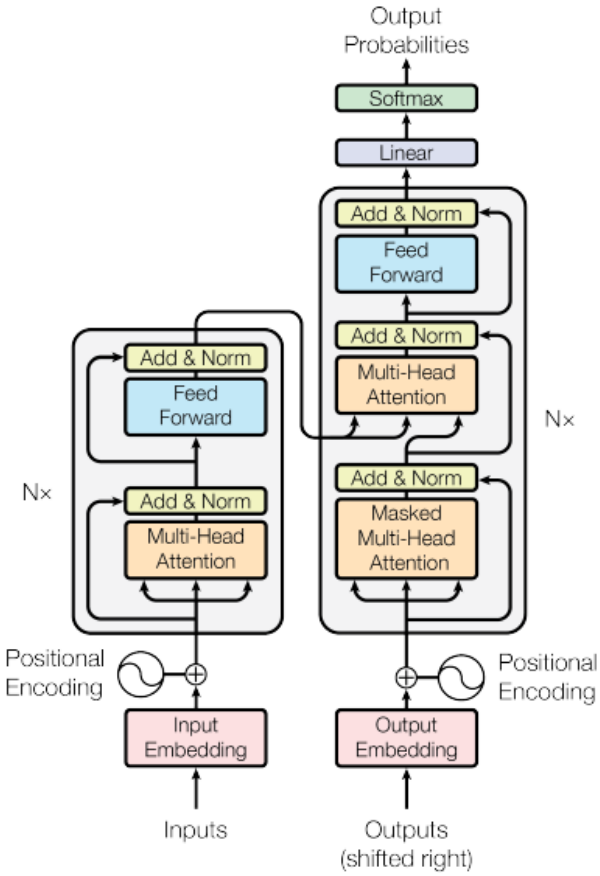
Encoder-decoder architectures

A common type of architecture used in many tasks.

- The **encoder** maps the input to some latent representation, usually of a low dimension.
  - In the context of our course - the encoder is some type of **feature extractor**
- The **decoder** applies a different mapping, from the latent space to some other space (sometimes back to the input space).
  - This can be used to *generate* **new** data.

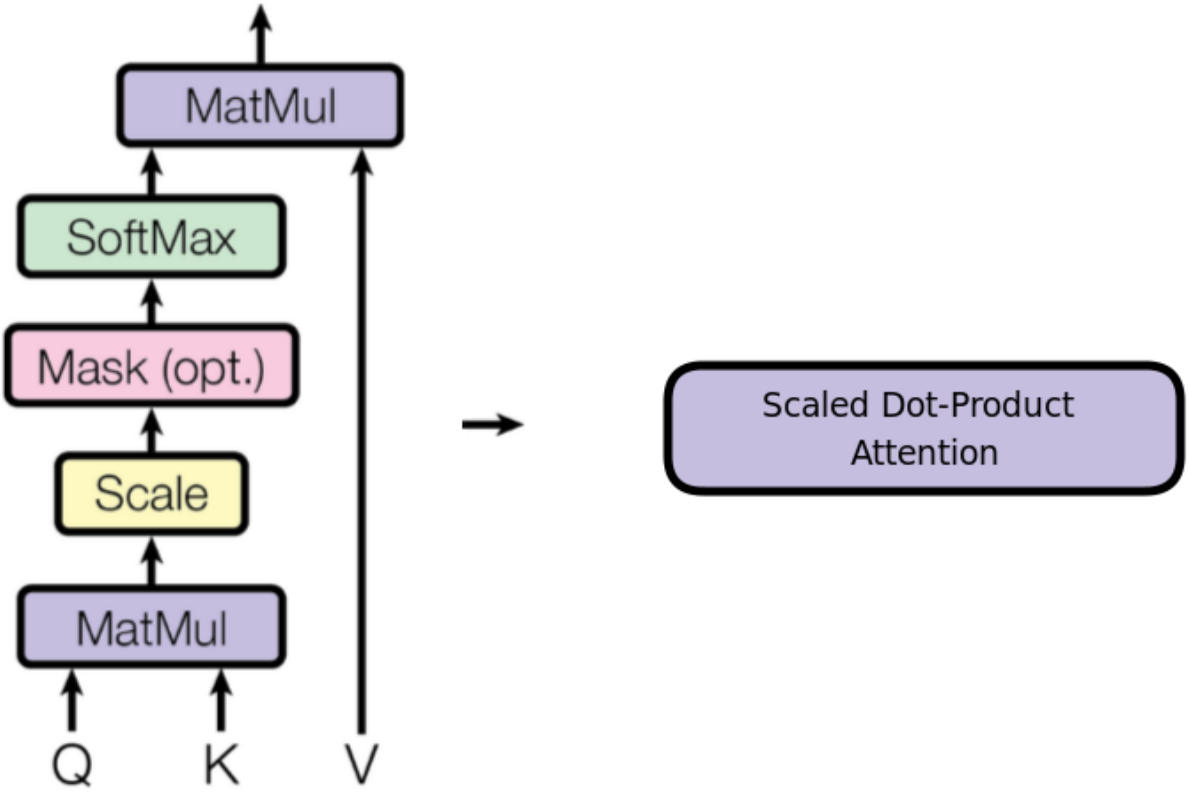


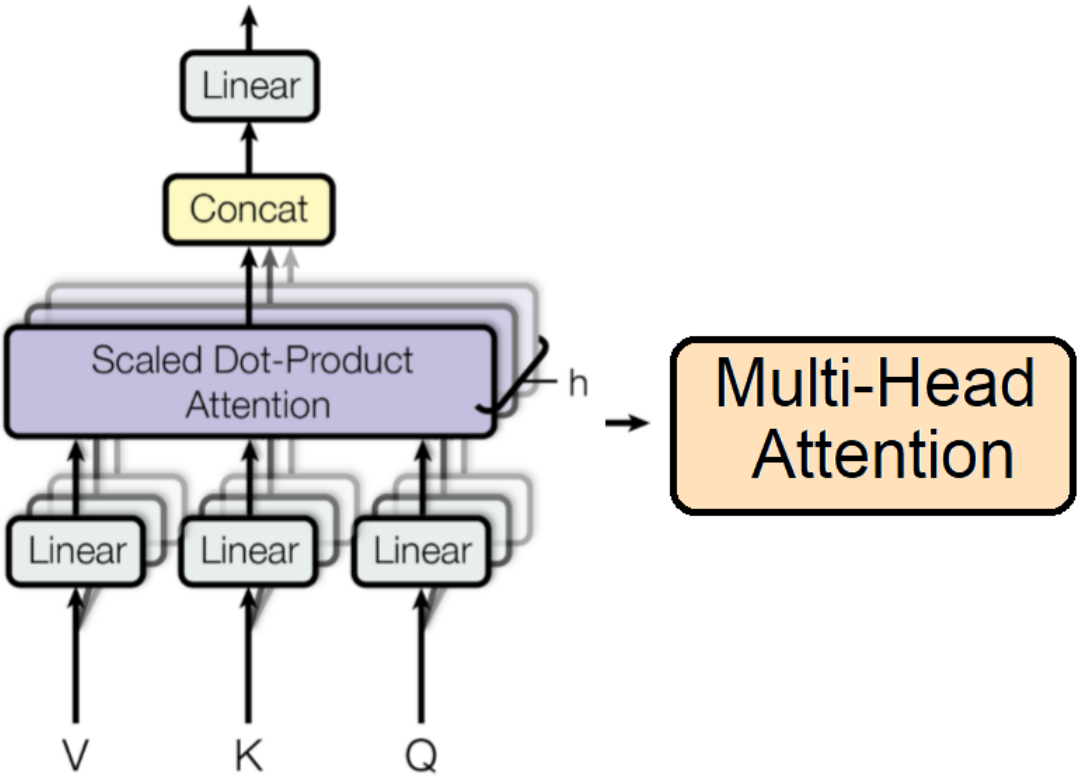
Scary Transformer Diagram



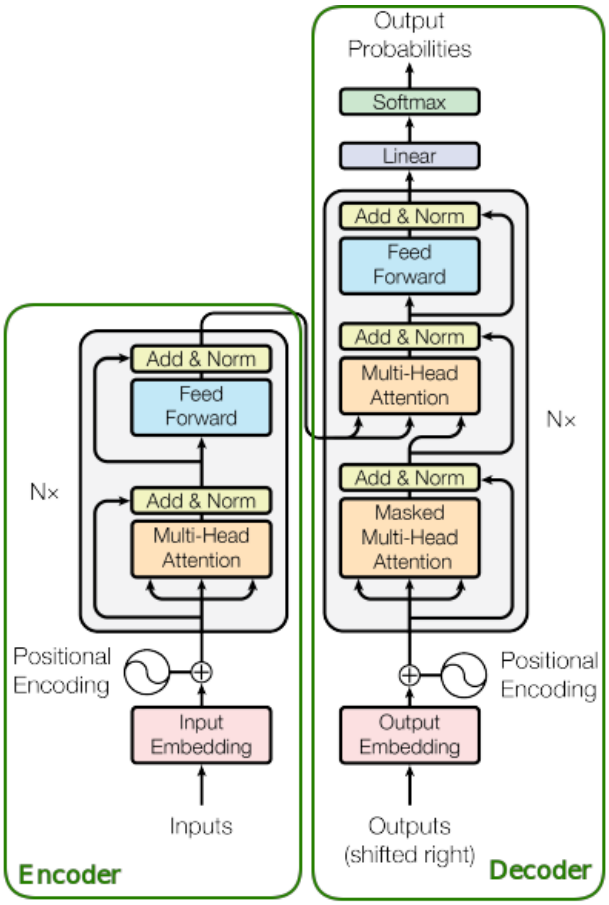
Let's Break it Down

- Let's start from what we know:

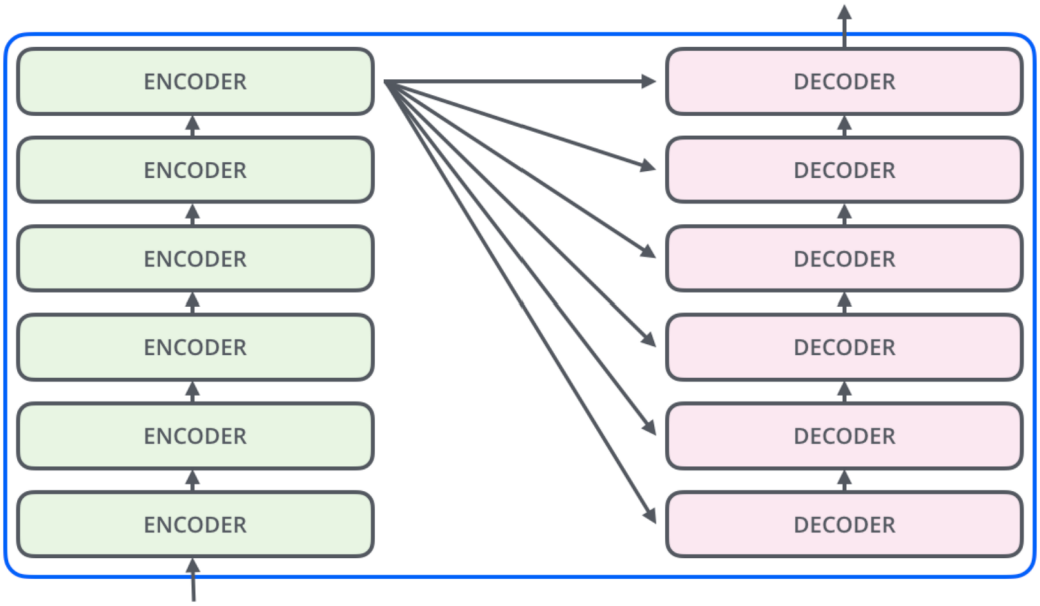




- In the transformer:
  - The encoder uses attention layers to create meaningful features.
  - The decoder uses the previous outputs to query which of those features is important for the next output (keys and values).



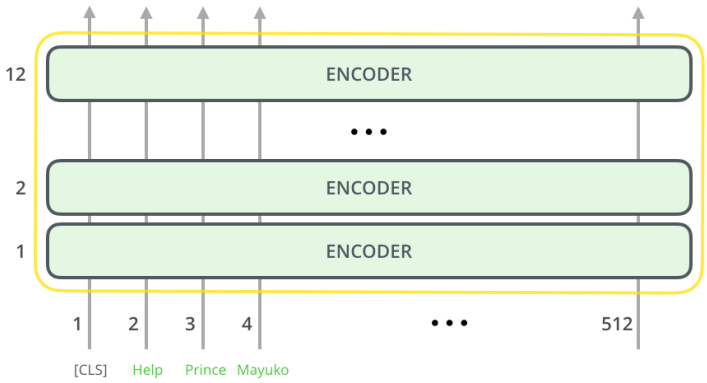
- In reality we stack encoders and decoders:



A Zoo of Transformers

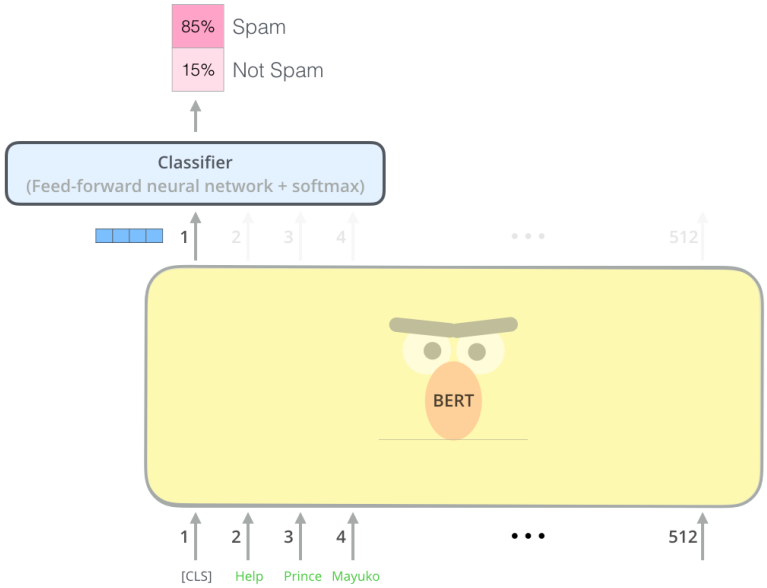
- Since that paper, people have used the transformer architecture in many different ways, for many different tasks.

- Just the encoder (BERT (https://arxiv.org/abs/1810.04805))



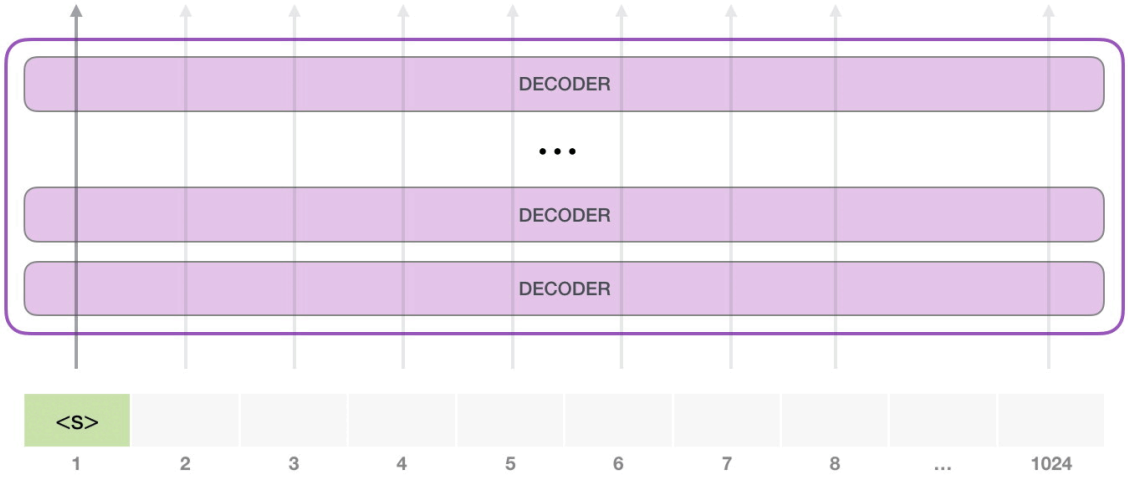
  - This is just a feature extractor!

- So we can add heads for specific tasks.
  - Can be used with pre-training and fine-tuning.



- Image Source: Jay Alammar's The Illustrated BERT, ELMo, and co. (https://jalammar.github.io/illustrated-bert/)

- Just the decoder (GPT-2 (https://openai.com/blog/better-language-models/))



  - Generate data - for many types of tasks.
- GIF adapted from: Jay Alammar's The Illustrated GPT-2 (https://jalammar.github.io/illustrated-gpt2/)

- Why not use them on everything, everywhere, all at once?

please state shortly and concisely the main disadvantages of transformers

The main disadvantages of transformers are:

1. High computational complexity and resource requirements.
2. Large data requirement.
3. Low interpretability and difficulty in debugging.
4. High memory requirement.
5. Sensitivity to initialization.

Recommended Videos

- Both of the channels below are very recommended for all things deep learning
  - Yannic Kilcher - Attention Is All You Need (https://www.youtube.com/watch?v=iDulhoQ2pro)
  - Aleksa Gordić - The AI Epiphany - Attention Is All You Need (Transformer) | Paper Explained (https://www.youtube.com/watch?v=cbYxHkgkSVs)

Credits

- ECE 046211 Winter 22-23 - Tal Daniel (https://taldatech.github.io/)
- CS 236781 Winter 22-23 - Moshe Kimhi, Aviv A. Rosenberg (https://vistalab-technion.github.io/cs236781/tutorials/)
- Jay Alammar's The Illustrated Transformer (https://jalammar.github.io/illustrated-transformer/)
- Jay Alammar's Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention) (https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/)
- Attention is all you need, Ashish Vaswani et al. (https://arxiv.org/abs/1706.03762)
- Icons from Icon8.com (https://icons8.com/) - https://icons8.com (https://icons8.com)