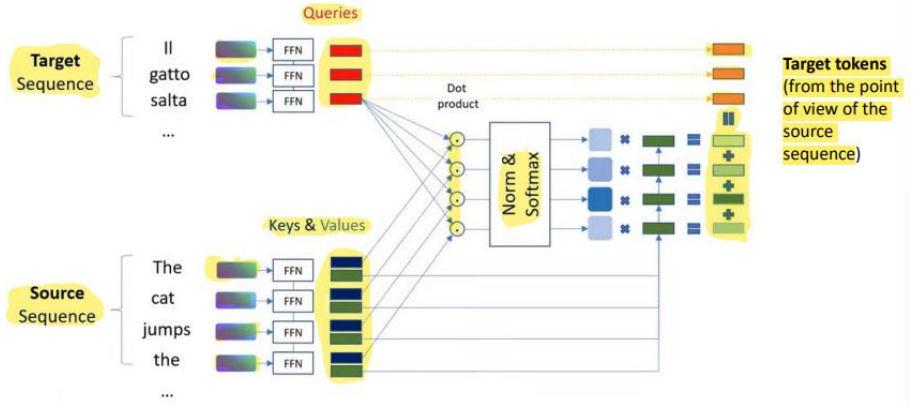
Chapter 15 – Transformers in Vision

Author: Gianmarco Scarano

gianmarcoscarano@gmail.com

1. RNNs

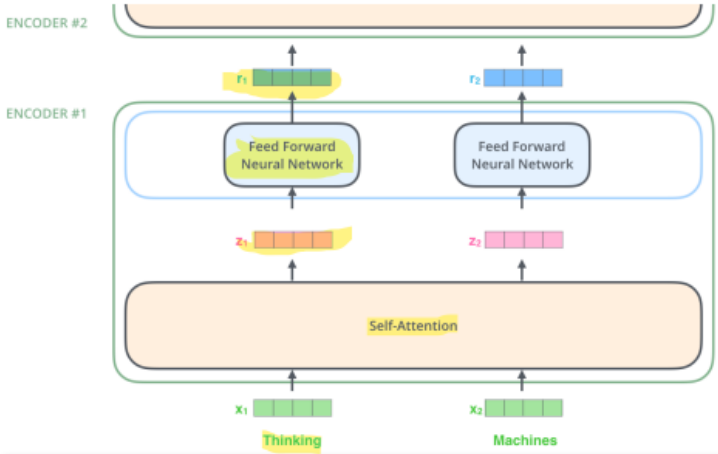
Recurrent models generate a sequence of hidden states h_t as a function of the previous hidden state h_{t-1} and the input for position t. There are quite few problems though, such as forgetting token too far in the past and we the need to wait the previous token to compute the next hidden state.

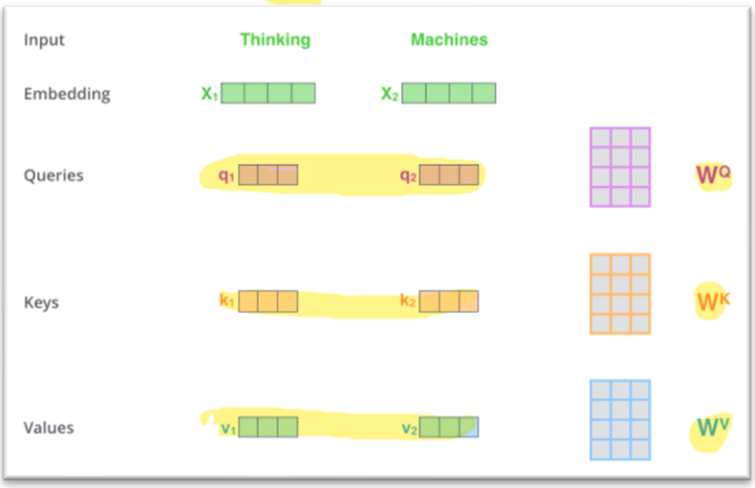


This has been solved by the paper "Attention is all you need" where we add a very powerful attention mechanism which is a milestone in the Transformer architectures.

2. Encoder / Self-Attention / Multi-head Attention

When we are dealing with an Attention mechanism, we can't forget the Encoder part, which englobes the Self-Attention method itself and the FFN (Feed-Forward Neural Network). The Encoders are all identical in structure.





Taking a quick look at the Self-Attention layer, it helps the Encoder to look at other words coming from the input sentence while it is encoding a specific word. The output of the Self-Attention Layer is then fed into the FFN and then into the next Encoder.

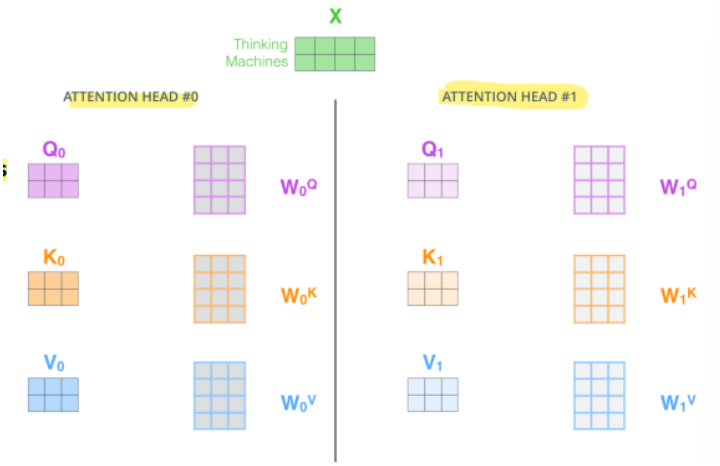
We can see how each word is treated separately (basically a list of vectors as input) into the Self-Attention mechanism, whose task is to look at other position in the input sequence for clues that can help into a better encoding of that specific word (x_1, x_2 etc.).

This works through a method called **<Query, Key, Value>.** The Self-Attention mechanism creates three vectors from the embeddings of each word by multiplying the embedding itself by three matrices that we train during the training process.

These vectors are abstractions that are useful for calculating attention. The concept is the same when we find a video on YouTube. The

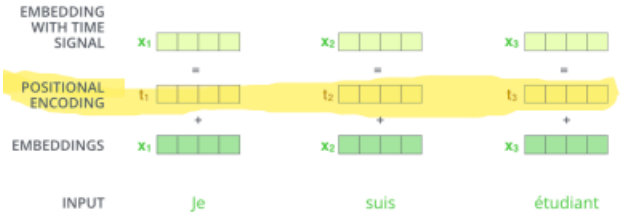
search engine will map your **QUERY** (text in the search bar) against a set of **KEYS** (video title, description, etc.) associated with candidate videos in their database, then present you the best matched videos (**VALUES**).

Afterwards, we calculate a score for each embedding by taking the dot product of the **query** vector with the **key** vector of the respective word we're scoring, then dividing the scores by the square root of the dimension of the **key** vectors. We pass this value through a SoftMax operation, which outputs a score (we call it **SoftMax score**). Finally, we multiply each **Value** by the **SoftMax score** and sum up the weighted value vectors.



The Multi-head Attention (**MHA**) is an instantiation of multiple Attention mechanisms. At the end, we concatenate all the attention heads, multiplying it with a weight matrix that was trained jointly with the model (which is then passed into the Feed Forward Network).

One thing that's missing from the model is a way to keep track of the words order in the input sequence! To address this, the transformer adds a vector to each input embedding which helps it determine the position of each word.



Doing a summary of the Self-Attention Layers, they were found to be faster than recurrent layers for shorter sequence lengths, while they were also lighter than Convolutional Layers since they would require the use of large filters and stack of Convolutional Layers.

Key points for the transformers:

- The encoder attends to all words in the input sequence.
- The decoder receives as input its own predicted y_{t-1}.
- Self- Attention calculation is **O(n^2)**

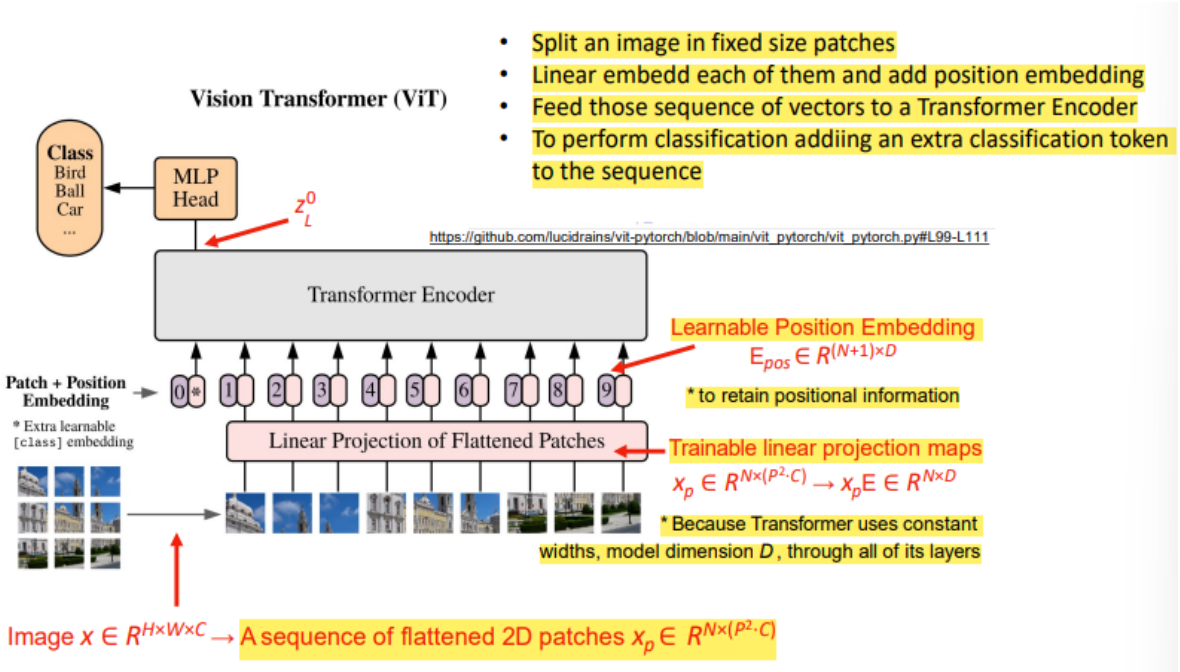
3. Attention to Vision (ViT – Vision Transfomers)

As Attention to words bring focus on specific words, Attention to images concentrate on relevant parts of the image or on Computer Vision tasks such as *Classification, Detection and Segmentation*.

The idea is to treat each pixel as a token and pass it to the transformer, but this scales quadratically in the number of pixels (256x256 image -> 62.500 pixel -> 3.906.250.000 calculations) so it's very expensive. Also scaling the pixels values will bring us to loss of detail.

The key points of the ViT are to apply a Transformer block directly to images with the fewest possible modifications, along with having excellent results when pre-trained and transferred (Transfer Learning).

Here, I leave a very self-explanatory overview of its architecture and its main points.



At the end, we pretrain the model with image labels, along with finetuning on a smaller dataset for image classification. This induces the training to be faster, to achieve larger models and better performances across vision and language tasks.

Transformers have been used for Object Detection (DETR), panoptic segmentation and video recognition.

Famous ViT which need to be mentioned are *Swin Transformer*, *Dino* & *Efficient Transformer.*