Which Pre-Trained Transformer Model to use for Fine-Tuning ?

In choosing a pre-trained Transformer model, there are many choices

- Architecture type: Encoder, Decoder, Encoder/Decoder
- Size
- Training set

We offer some guidelines, assuming our goal is to Fine-Tune the Pre-Trained model to a new Target task.

Architecture

Encoder

An Encoder converts the sequence $\mathbf{x}_{(1:\bar{T})}$ to the sequence $\bar{\mathbf{h}}_{(1:\bar{T})}$

- transforming the "raw" sequence (each position in isolation) $\mathbf{x}_{(1:\bar{T})}$
- into the "processed" sequence $\bar{\mathbf{h}}_{(1:\bar{T})}$

The salient feature of the Encoder is the *unmasked* Attention.

- $\bar{\mathbf{h}}_{(\bar{t})}$: the *contextualized representation* at position $\bar{t}$
    - is a function of the entire input $\mathbf{x}_{(1:\bar{T})}$

That is: the output positions are *context-sensitive* representations.

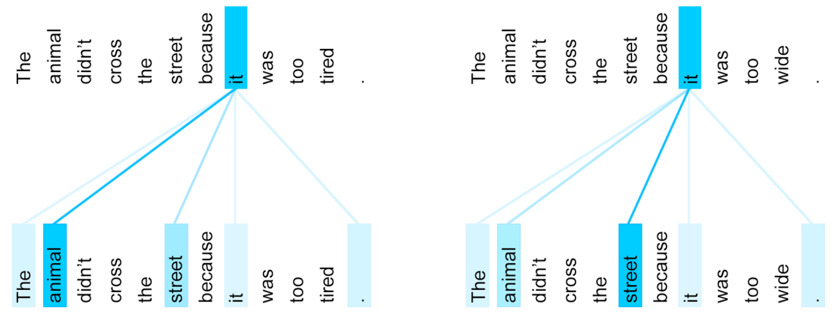
- where the context include everything prior to and after position $\bar{t}$

For example: the position corresponding to the word "it"

- in each of the two sentences
- is ambiguous in the raw sequence
- is unambiguous in the processed sequence
    - it has been informed by the tokens preceding and following its position

Attention weights

Thickness of the blue lines indicate the strength of attention to other tokens



Picture from: https://1.bp.blogspot.com/-AVGK0ApREtk/WaiAuzddKVI/AAAAAAAAB_A/WPV5ropBU-cxrcMpqJBFHg73K9NX4vywwCLcBGAs/s1600/image2.png

Thus, an Encoder-only model is useful for

- adding contextual information to the raw input
- as a source of a sequence that can be "pooled" into a single positions
    - that is a fixed length summary of the entire sequence
    - and thus can be feed to simpler NN layers (e.g., a Fully Connected layer acting as a Classifier)

Decoder

The Decoder produces an output $\hat{\mathbf{y}}_{(t)}$ at each position t that is a

- function of all preceding outputs $\hat{\mathbf{y}}_{(1:t-1)}$

Thus, the salient features of the Decoder are its *Autogregressive behavior* and *Causal masking*

This makes it appropriate for

- generative tasks
- where the output positions are generated one position at a time
- and each position is informed *only* by preceding (i.e., previously generated) positions

Encoder/Decoder

The Encoder/Decoder takes an input $\mathbf{x}_{(1:\bar{T})}$ and produces an output $\hat{\mathbf{y}}_{(1:T)}$

The output at each position t is a function of

- $\mathbf{x}_{(1:\bar{T})}$ using Cross-Attention
    - via the processed input $\mathbf{h}_{(1:\bar{T})}$
- $\hat{\mathbf{y}}_{(1:t-1)}$ via Self-Attention to the previously generated outputs

Most useful for Sequence to Sequence tasks where

- the input sequence may be fully accessed at all positions (**no** masking)
- the output sequence is generated autoregressively
    - causal attention to the previously generated output
    - with full access to the "processed" input (i.e., output of the Decoder)

Examples include

- language translation
- question answering

Some of these tasks might be able to be handled by a Decoder-only architecture using *Causal with Prefix* Attention

- the conditioning input $\mathbf{x}_{(1:\bar{T})}$ is prepended to the Decoder output $\hat{\mathbf{y}}_{(1:T)}$
- the Decoder, when producing output $\hat{\mathbf{y}}_{(t)}$ at position t has
    - *unmasked* Attention to the conditioning prefix $\mathbf{x}_{(1:\bar{T})}$
    - *causal masked* Attention to $\hat{\mathbf{y}}_{(1:t-1)}$

The full access to all of $\mathbf{x}_{(1:\bar{T})} atalloutputpositions$\tt$

- is essentially replicated what the Encoder of an Encoder/Decoder architecture would do

Forms of Attention

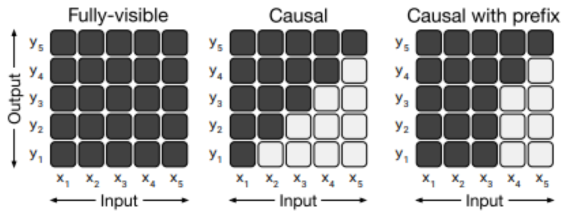


Figure 3: Matrices representing different attention mask patterns. The input and output of the self-attention mechanism are denoted x and y respectively. A dark cell at row i and column j indicates that the self-attention mechanism is allowed to attend to input element j at output timestep i. A light cell indicates that the self-attention mechanism is *not* allowed to attend to the corresponding i and j combination. Left: A fully-visible mask allows the self-attention mechanism to attend to the full input at every output timestep. Middle: A causal mask prevents the ith output element from depending on any input elements from "the future". Right: Causal masking with a prefix allows the self-attention mechanism to use fully-visible masking on a portion of the input sequence.

Attribution: https://arxiv.org/pdf/1910.10683.pdf#page=15

Per the diagram

- when producing output $\hat{\mathbf{y}}_{(t)}$ for $1 \le t \le 3$
- the Decoder has access to the conditioning info $\mathbf{x}_{(1:3)}$

Size

Many models come in different size variations: small, medium, large.

Obviously the memory size and computing ability of the processor you are using may influenced your choice of model.

Training set

Fine-tuning a model that has been pre-trained on

- a larger training set
- that is more similar to the Target task examples

would seem to be most beneficial for Transfer Learning.

In [2]:

```
print("Done")
```

```
Done
```