Applications of Transformers

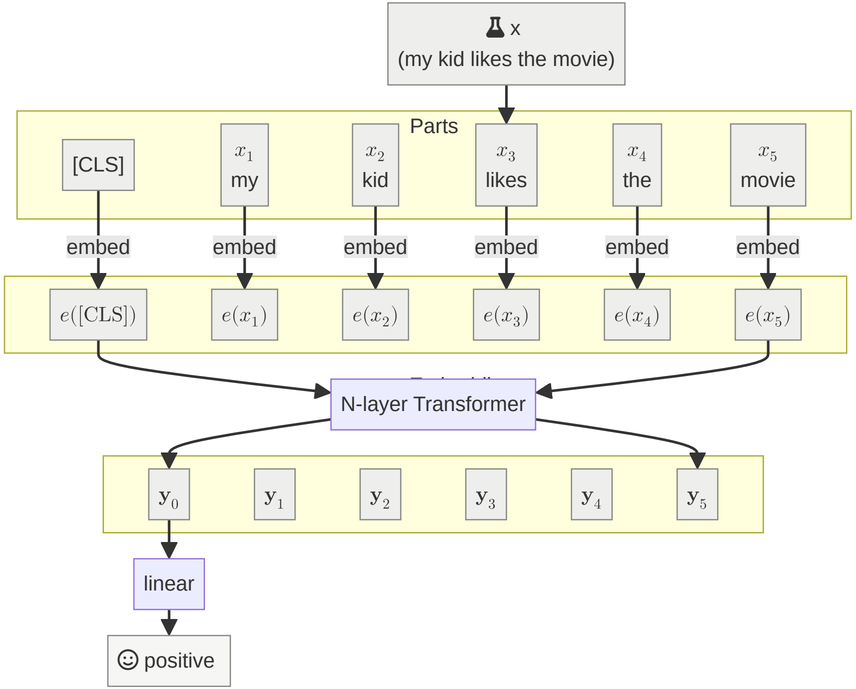
Recap: Applying Transformers to Sentiment Analysis

Input: a set of tokens $\{\mathbf{x}_i\}$

- prepended by a special token `[CLS]`

Output: another set of tokens $\{\mathbf{y}_i\}$

Transformer: stack of N transformer layers



What Else Can We Do With Transformers?

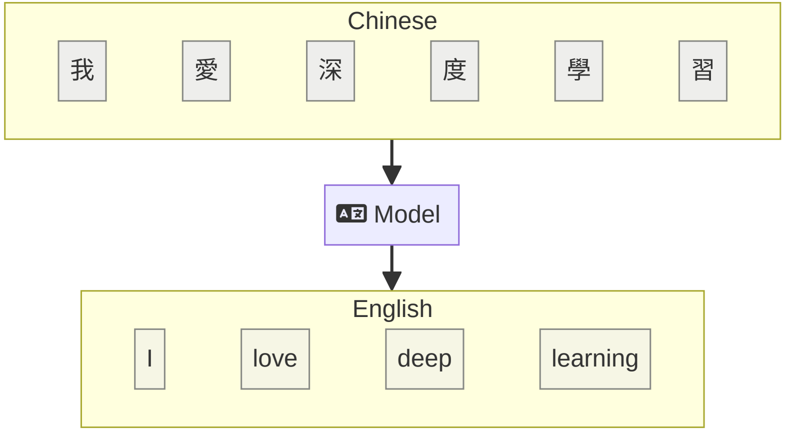
Machine Translation

Input: a sentence in a given language

Output: translation to another language

🈂 English: I love deep learning

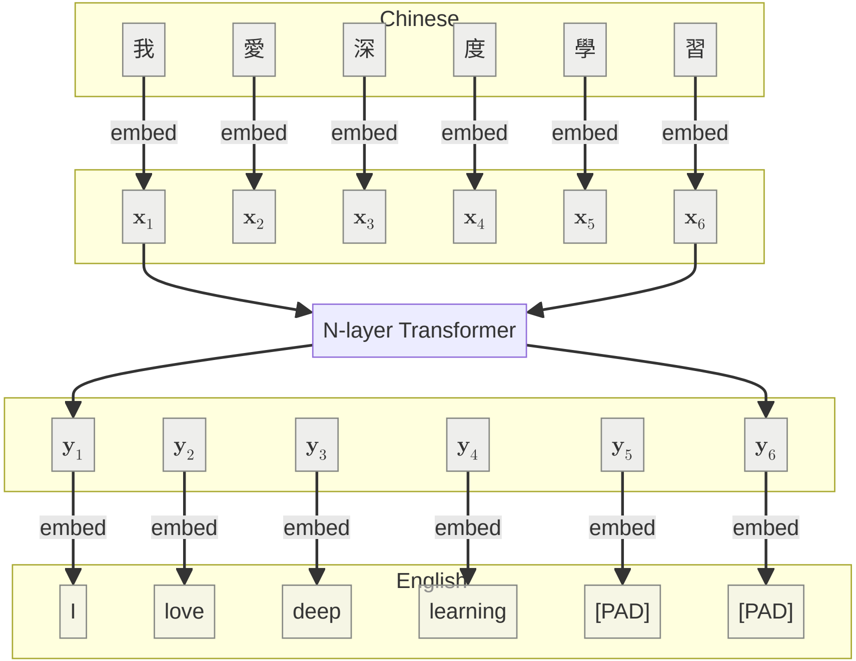
🈂 Chinese: 我愛深度學習



Challenges of Machine Translation

⚠ Length of input != length of output

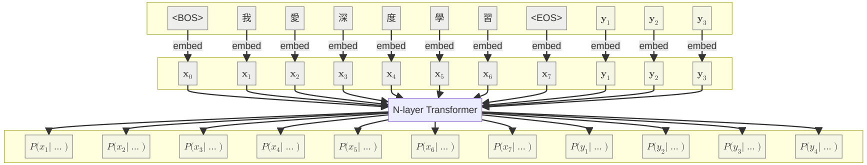
⚠ Hard to produce coherent output tokens simultaneously



Auto-Regressive Prediction

Predict one token (word) at a time

Br mult=4 />



1. $P(\tilde{\mathbf{y}}_1|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6)$
2. $P(\tilde{\mathbf{y}}_2|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \tilde{\mathbf{y}}_1)$
3. $P(\tilde{\mathbf{y}}_3|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2)$
4. $P(\tilde{\mathbf{y}}_4|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_3)$

...

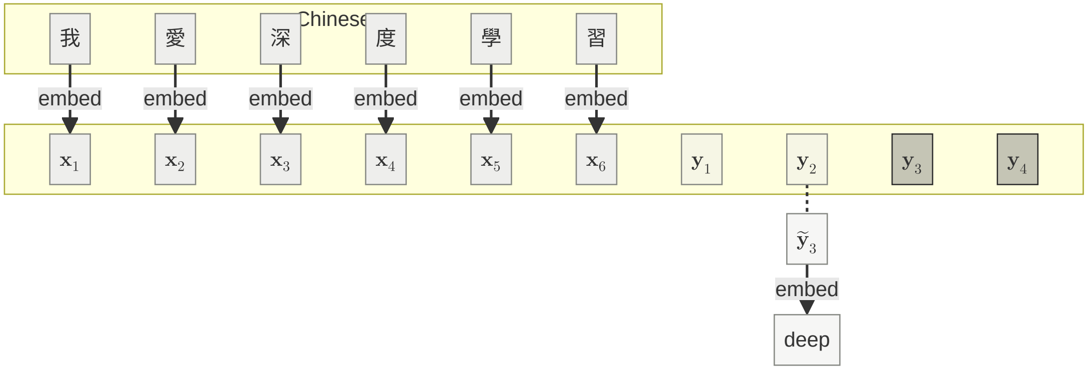
Until $\tilde{\mathbf{y}}_t$ hits an end-of-sequence (EOS) token

Here, $p(\tilde{\mathbf{y}}_t|\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \tilde{\mathbf{y}}_{t-1})$ is modeled by an N-layer Transformer

Masked Attention

Output sequence is offset by one compared to input

Model can easily cheat by looking at future tokens

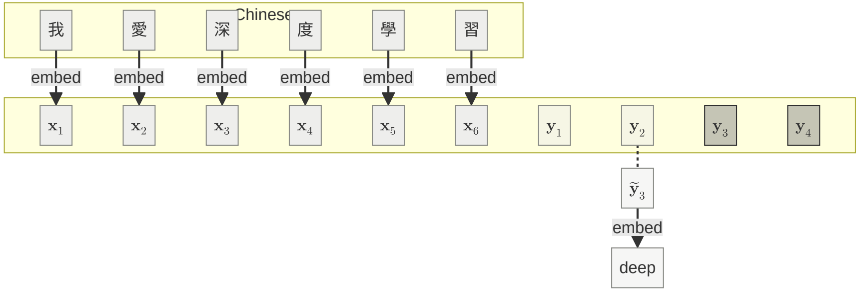


Masked Attention

$$\begin{aligned}\mathbf{O} &= \mathrm{MaskedAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \\ &= \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{C}} + \mathbf{M}\right)\mathbf{V}\end{aligned}$$

where the mask $\mathbf{M}$ is defined by

$$\mathbf{M} = \begin{bmatrix} 0 & -\infty & \cdots & -\infty \\ 0 & 0 & \cdots & -\infty \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$



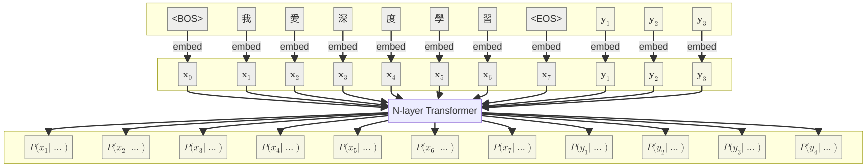
Auto-Regressive Prediction

Test Time:

Sample one token (word) at a time

$$P(\tilde{\mathbf{y}}_t | \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \tilde{\mathbf{y}}_1, \tilde{\mathbf{y}}_2, \cdots, \tilde{\mathbf{y}}_{t-1})$$

until $\tilde{\mathbf{y}}_t$ hits an end-of-sentence (`<EOS>`) token



☹ Very slow during training

Teacher Forcing

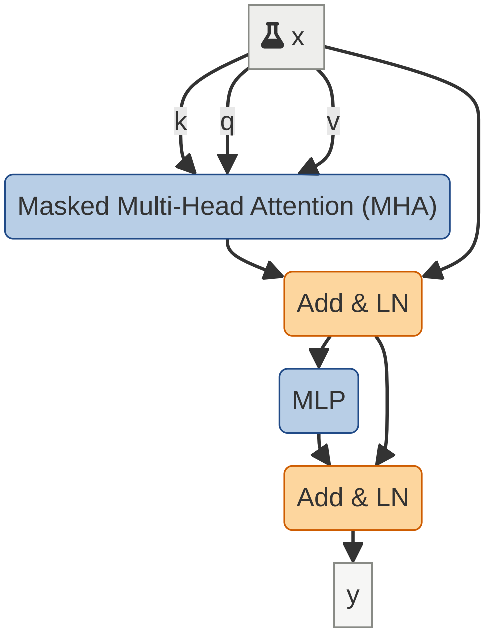
Fast training

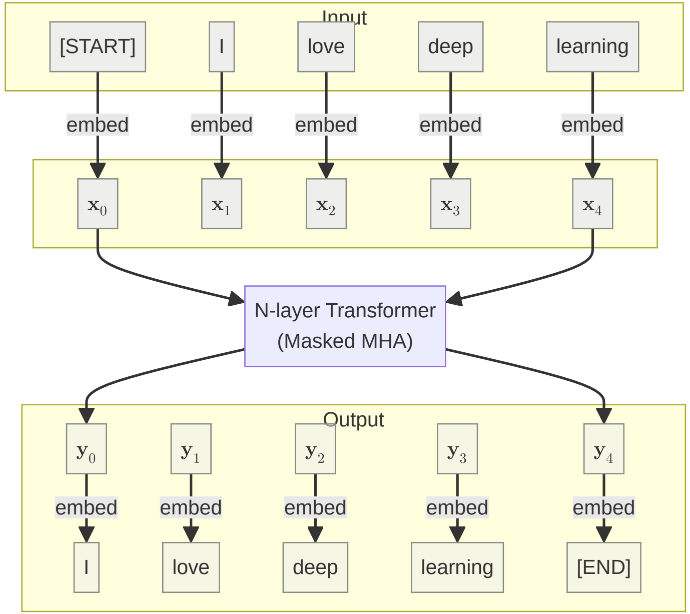
- Condition on ground truth inputs
- Different from what is seen during generation (sampling vs ground truth)
- Fine in practice
- Parallel training of all predictions

$$P(\tilde{\mathbf{y}}_t | \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_{t-1})$$

$\mathbf{y}_0$ is a special end-of-sentence (`<EOS>` = start-of-translation) token

Transformer Layer With Masked Attention





Types of Transformers

Decoder-only

Masked auto-regressive prediction

Encoder-only

No prediction, just understanding

Encoder-Decoder

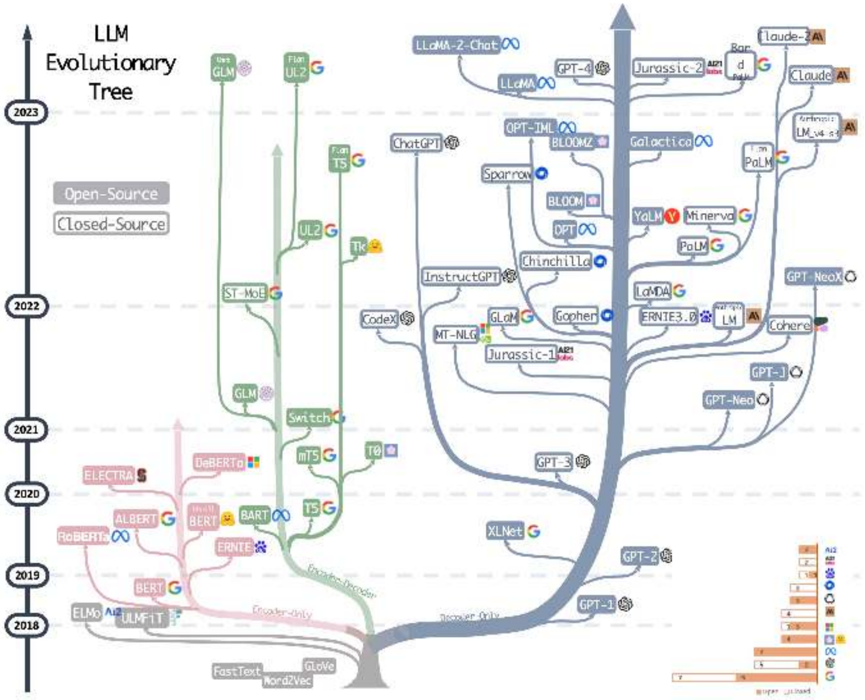


Image credit https://github.com/Mooler0410/LLMsPracticalGuide ↩

Types of Tokens

Tokens

- words or sub-words (tokenization)
- visual (e.g. image patches)
- discrete or continuous

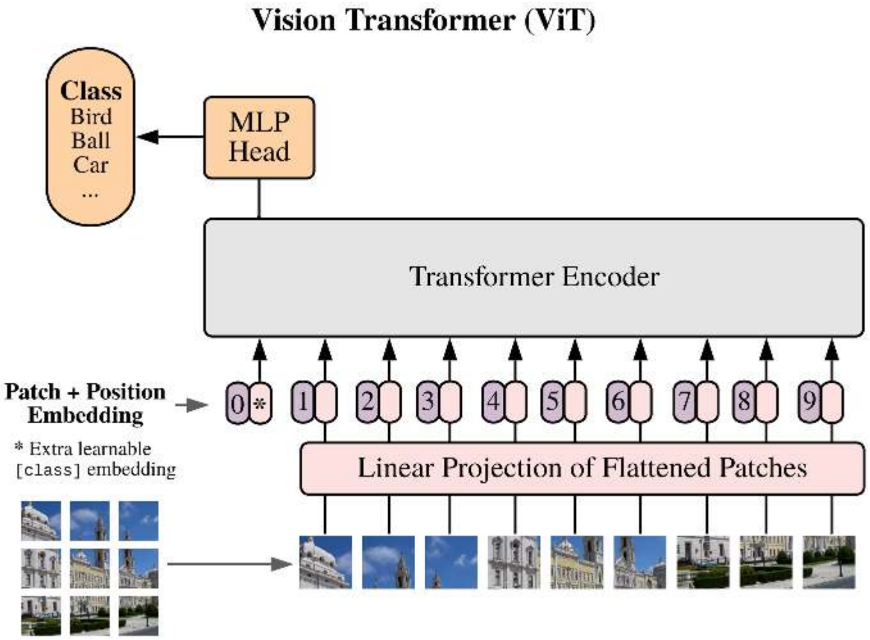


Image credit: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale ↩️

Applications of Transformers - TL;DR

Transformers are suitable language models

Auto-regressive next word prediction

Efficient parallel training through teacher forcing

Transformers process many forms of tokens