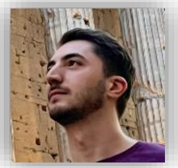
NLP and the Web – WS 2024/2025

Lecture 10
Neural Language Modeling 2


Dr. Thomas Arnold
Hovhannes Tamoyan
Kexin Wang

Ubiquitous Knowledge Processing Lab
Technische Universität Darmstadt

Syllabus (tentative)

Nr.	Lecture
01	Introduction / NLP basics
02	Foundations of Text Classification
03	IR – Introduction, Evaluation
04	IR – Word Representation
05	IR – Transformer/BERT
06	IR – Dense Retrieval
07	IR – Neural Re-Ranking
08	LLM – Language Modeling Foundations, Tokenization
09	LLM – Neural LLM
10	**LLM – Adaptation**
11	LLM – Prompting, Alignment, Instruction Tuning
12	LLM – Long Contexts, RAG
13	LLM – Scaling, Computation Cost
14	Review & Preparation for the Exam

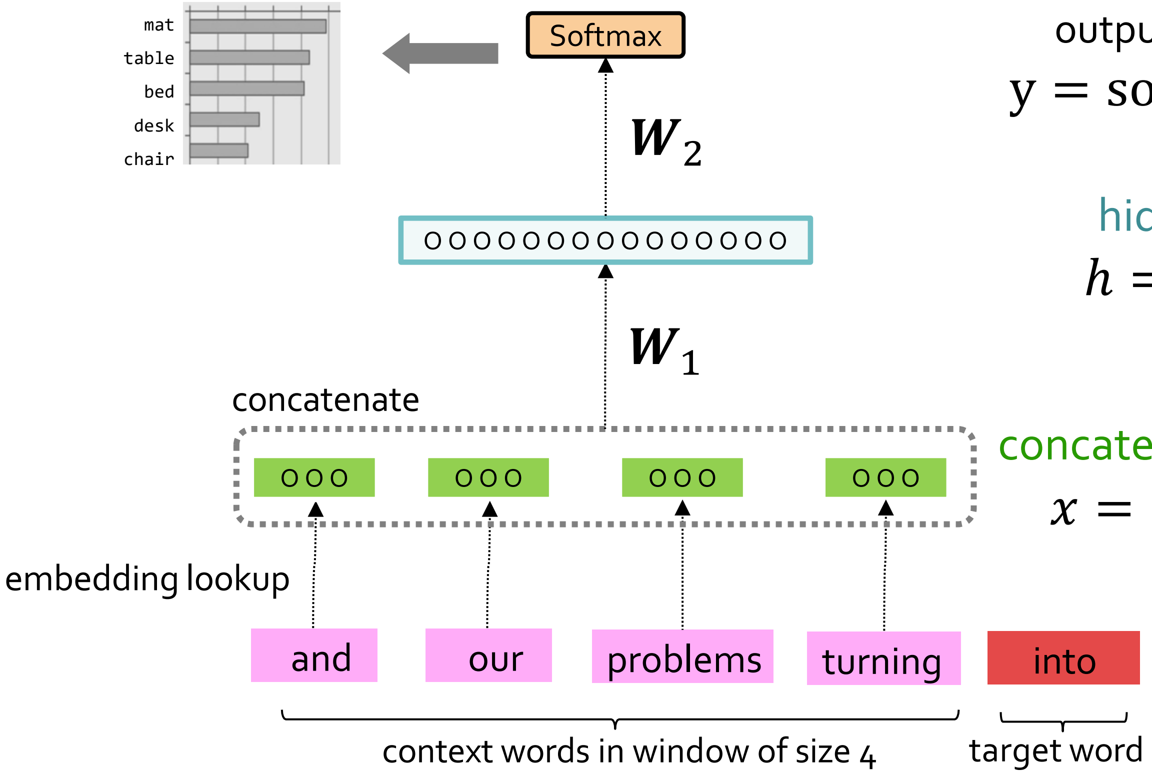
Outline

Transformer LM (cont.)

Adaptation

Lecture Evaluation, Quiz

A Fixed-Window Neural LM



output distribution

$$y = \text{softmax}(W_2 h)$$

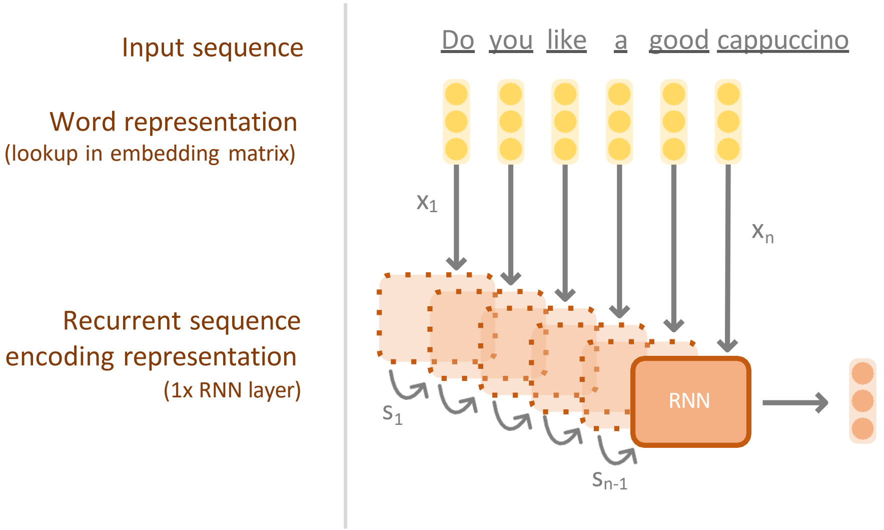
hidden layer

$$h = f(W_1 x)$$

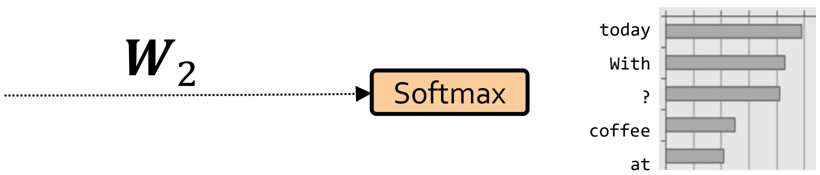
concatenated word embeddings

$$x = [v_1, v_2, v_3, v_4]$$

RNN Language Model

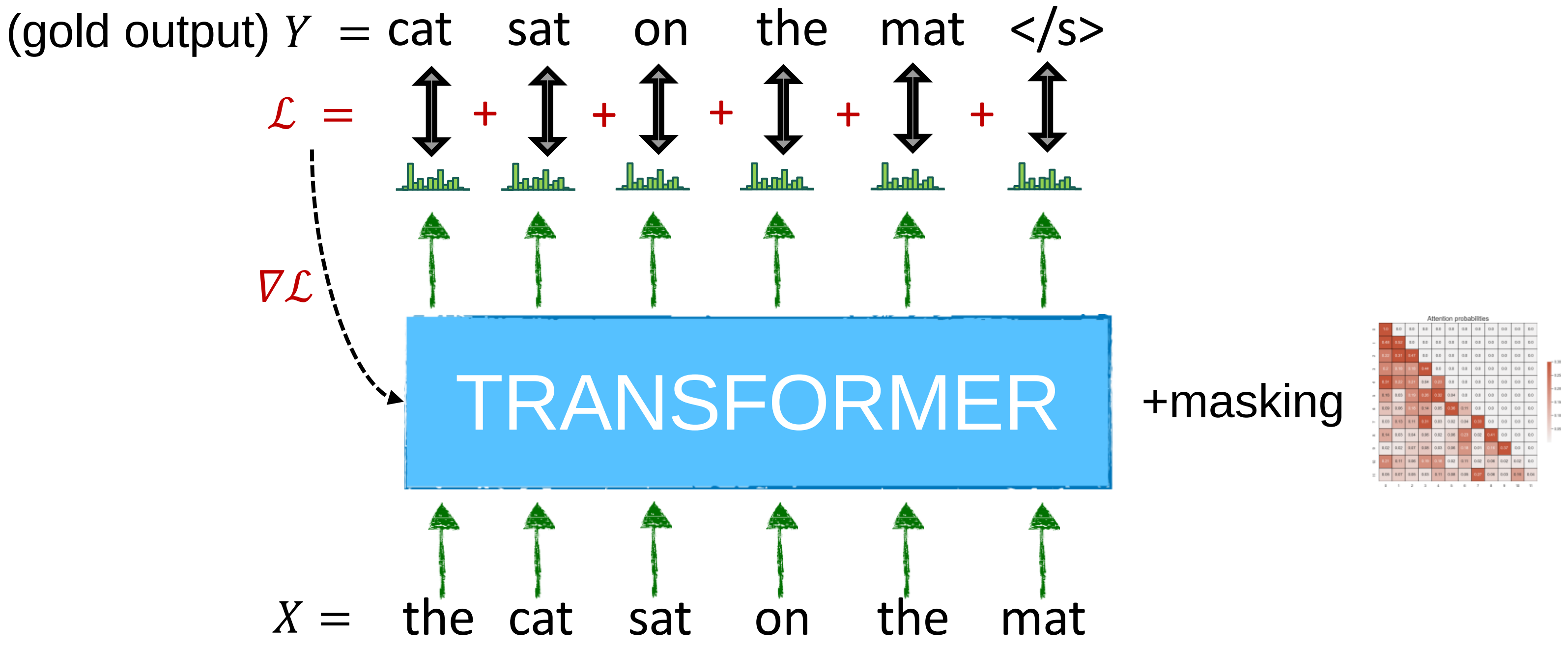


- Sequence as the input
- Single vector (last state) as output
  - Part of a larger network



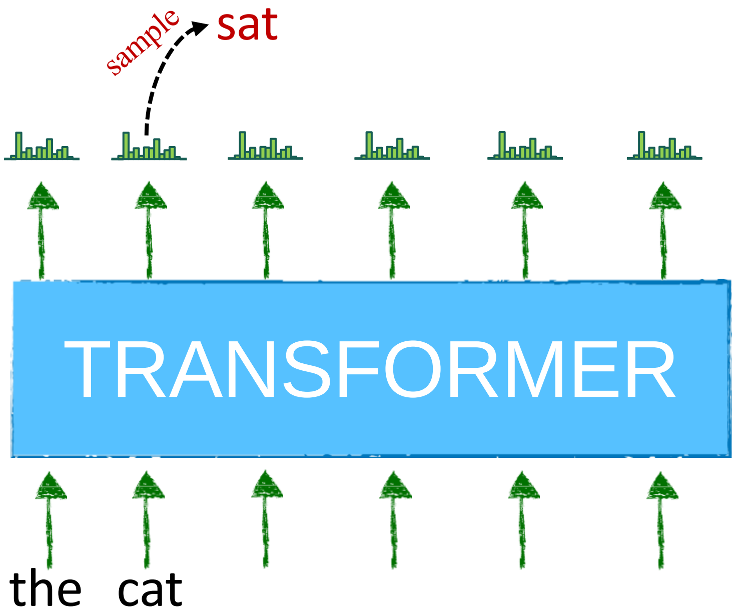
Training a Transformer Language Model

- We need to **prevent information leakage** from future tokens! How?



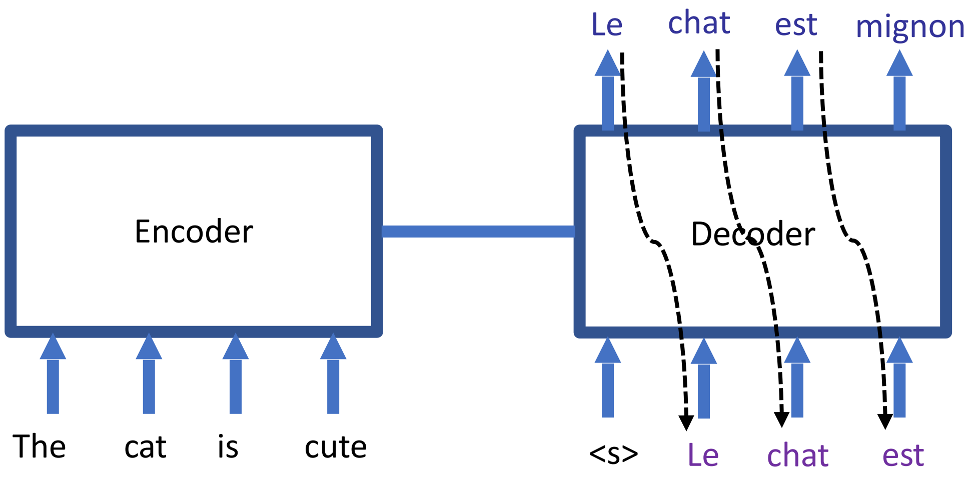
How to use the model to generate text?

- Use the output of previous step as input to the next step repeatedly



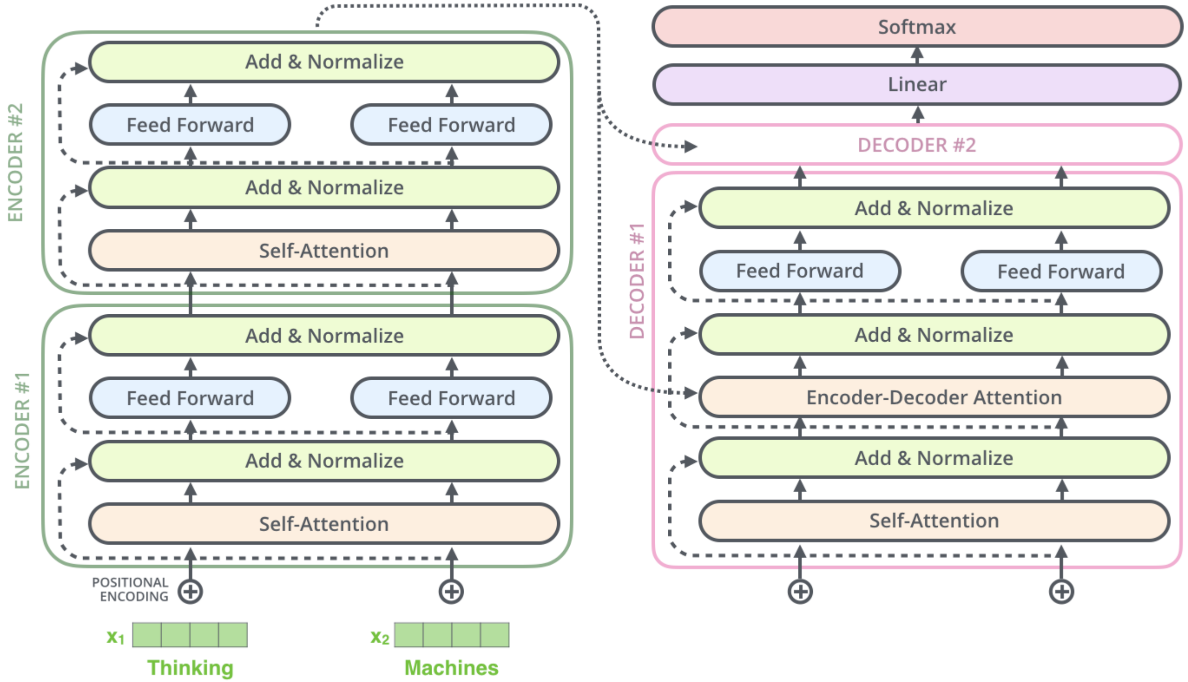
Encoder-decoder models

- Encoder = read or encode the input
- Decoder = generate or decode the output



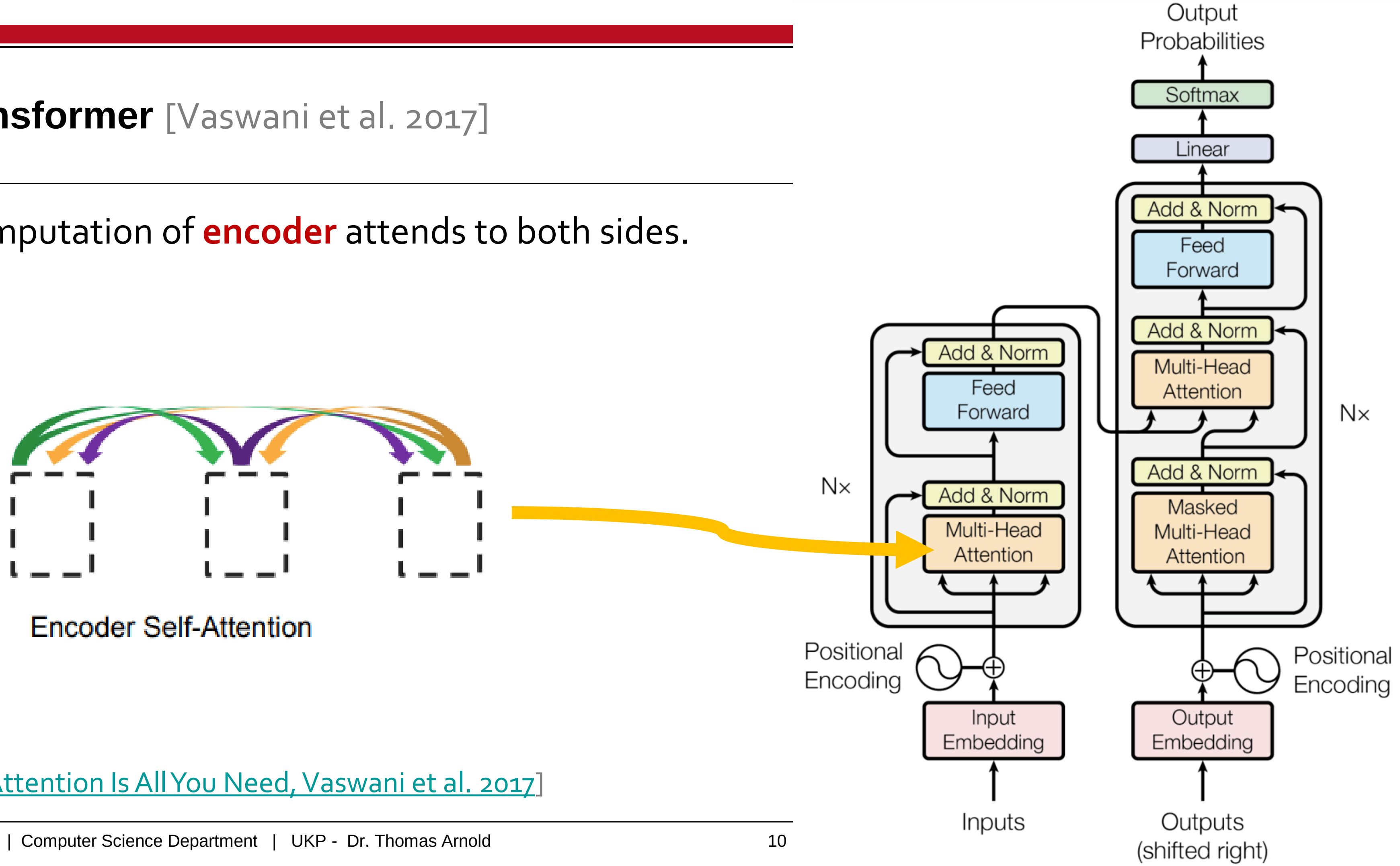
Transformer [Vaswani et al. 2017]

- An encoder-decoder architecture built with attention modules.



Transformer [Vaswani et al. 2017]

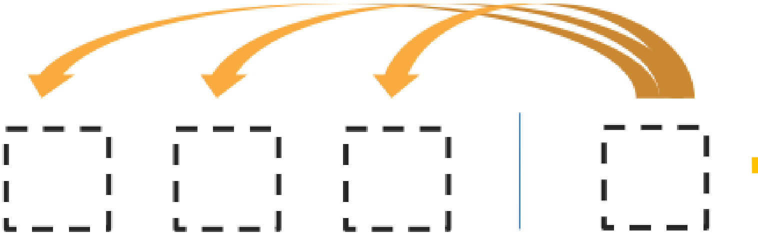
- Computation of **encoder** attends to both sides.



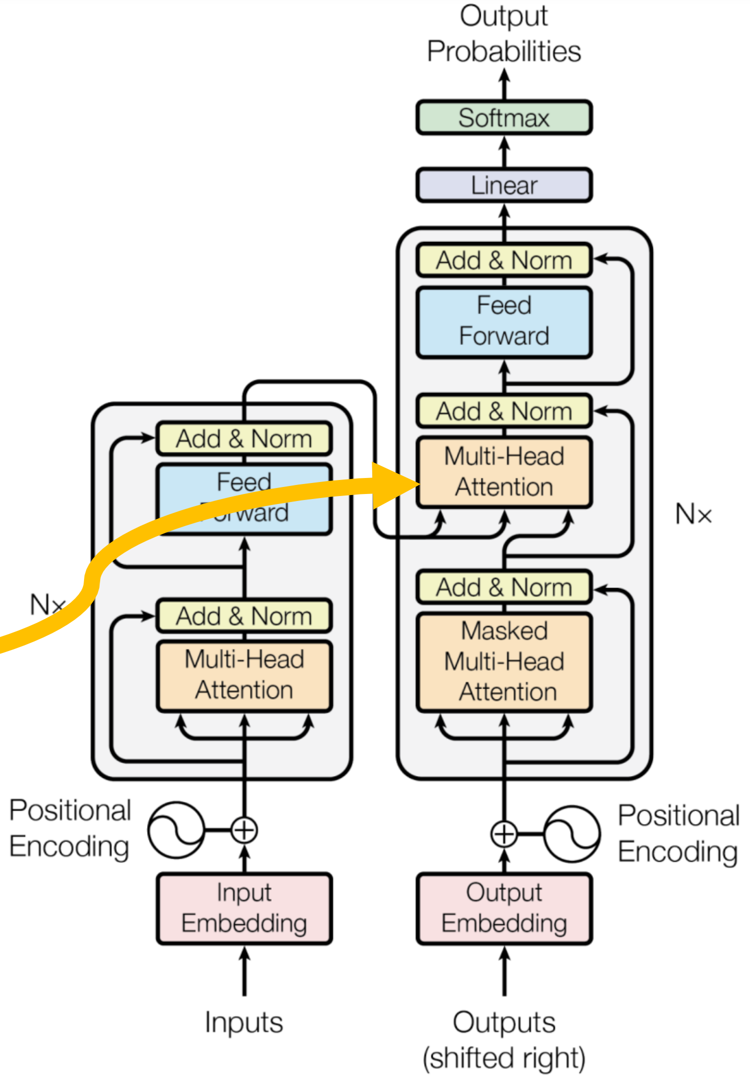
[Attention Is All You Need, Vaswani et al. 2017]

Transformer [Vaswani et al. 2017]

- At any step of **decoder**, it attends to previous computation of **encoder**

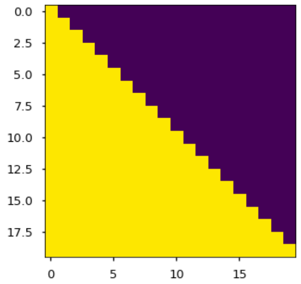


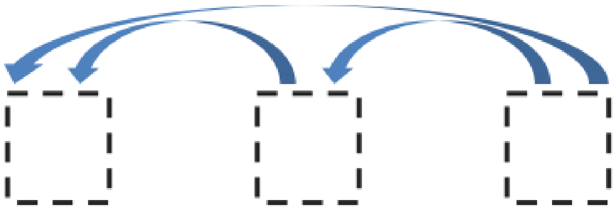
Encoder-Decoder Attention



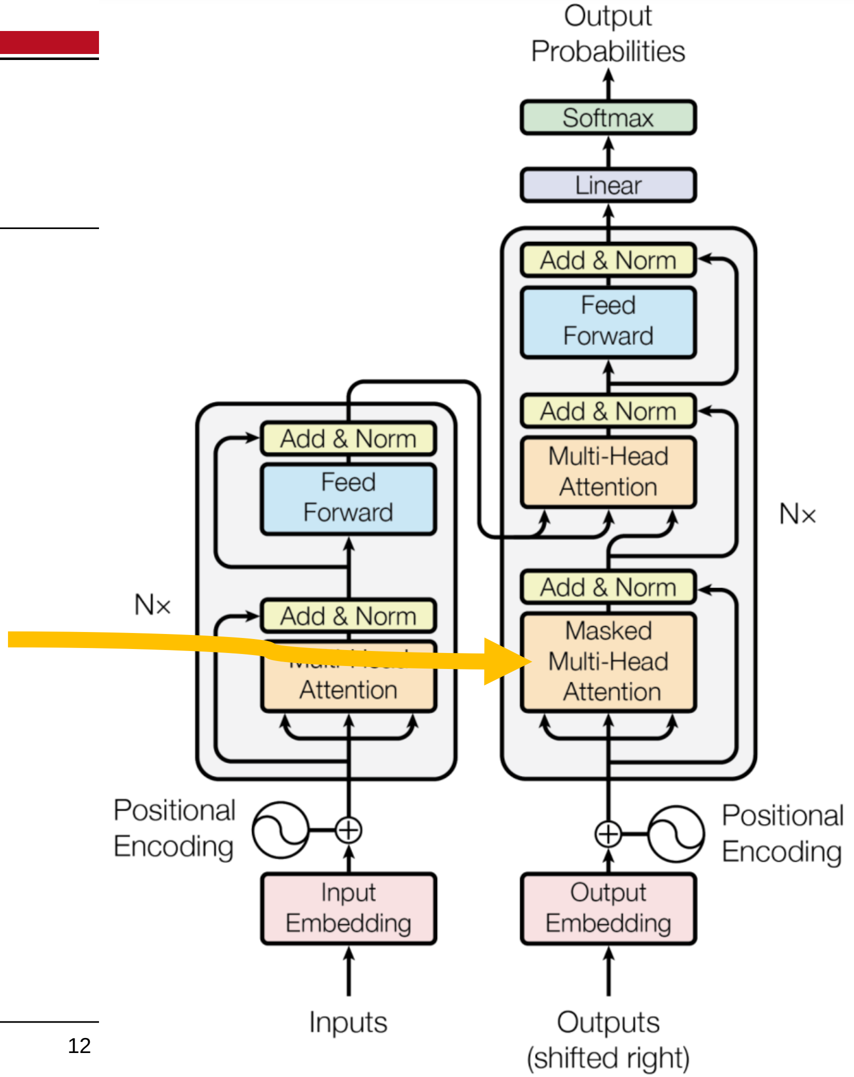
Transformer [Vaswani et al. 2017]

- At any step of **decoder**, it attends to **decoder's previous generations**





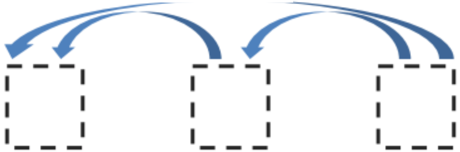
MaskedDecoder Self-Attention



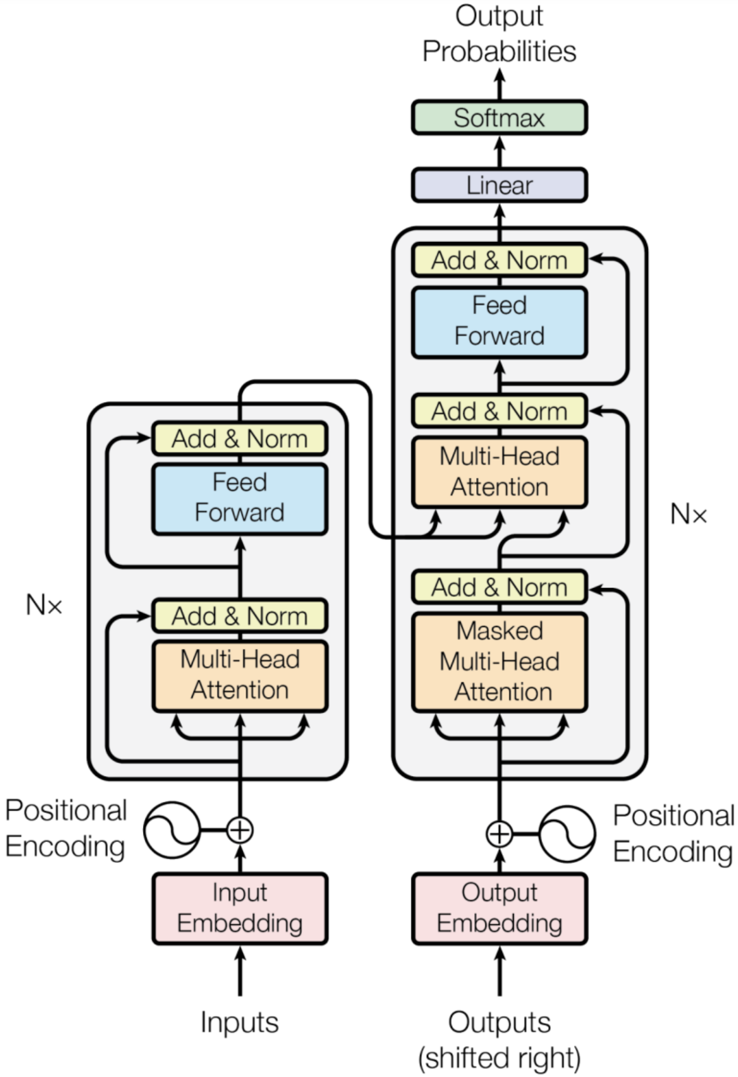
3 Shades of Attention

Encoder-Decoder Attention

Encoder Self-Attention



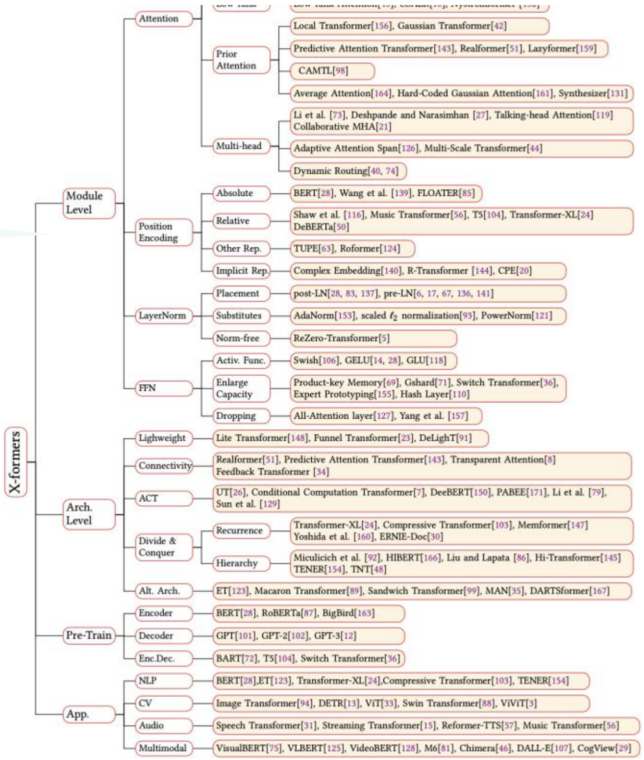
MaskedDecoder Self-Attention



Variants of positional embeddings

Architectural choices

Multi-modal models



Outline

Transformer LM (cont.)

Adaptation

Lecture Evaluation, Quiz

Language Models are not trained to do what you want

PROMPT	*Explain the moon landing to a 6 year old in a few sentences.*
COMPLETION	GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

There is a mismatch between LLM pre-training and user intents.

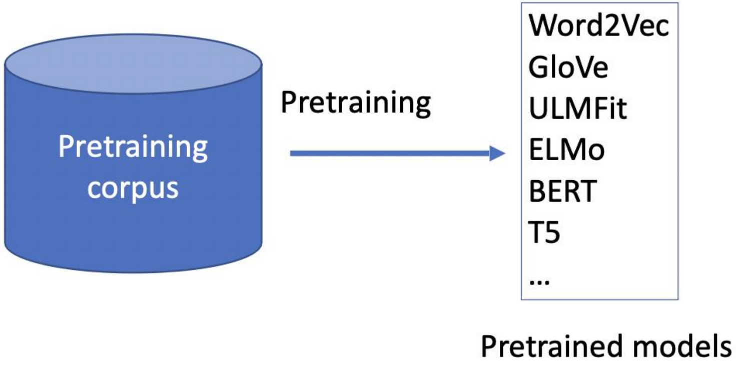
Adapting Language Models

You have a pre-trained language model that is pre-trained on massive amounts of data.
They do not necessarily do useful things—they only complete sentences.

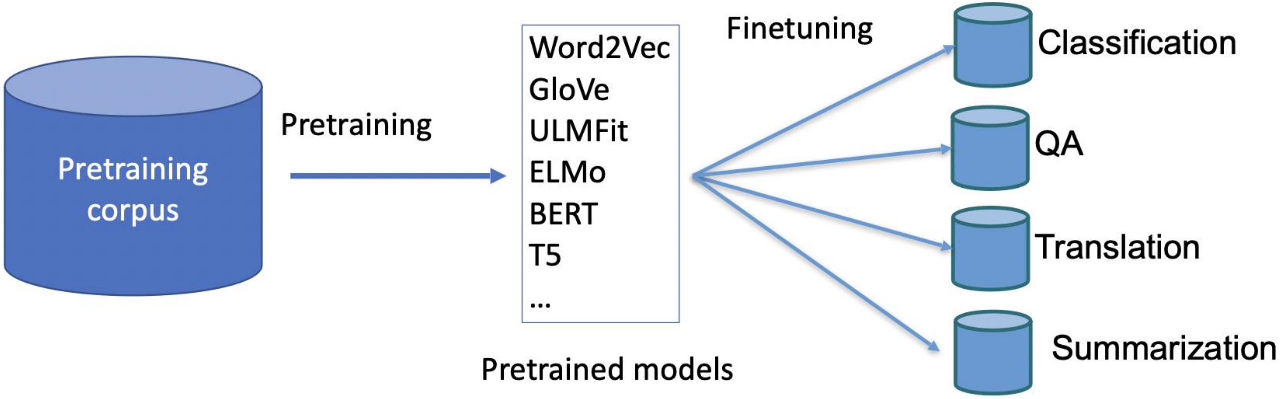
Now how to you ”adapt” them for your use-case?

- **Tuning:** adapting (modifying) model parameters
- **Prompting:** adapting model inputs (language statements)

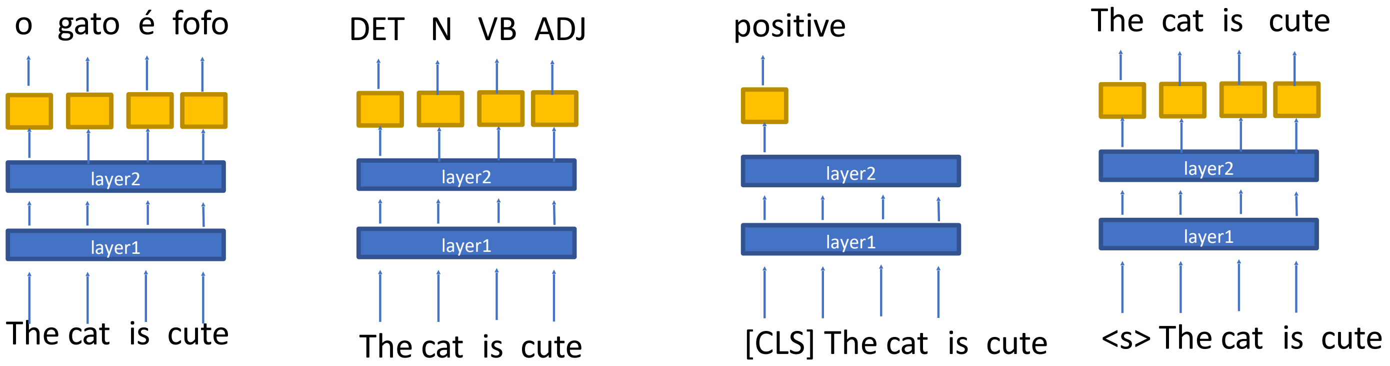
Fine-Tuning for Tasks



Fine-Tuning for Tasks



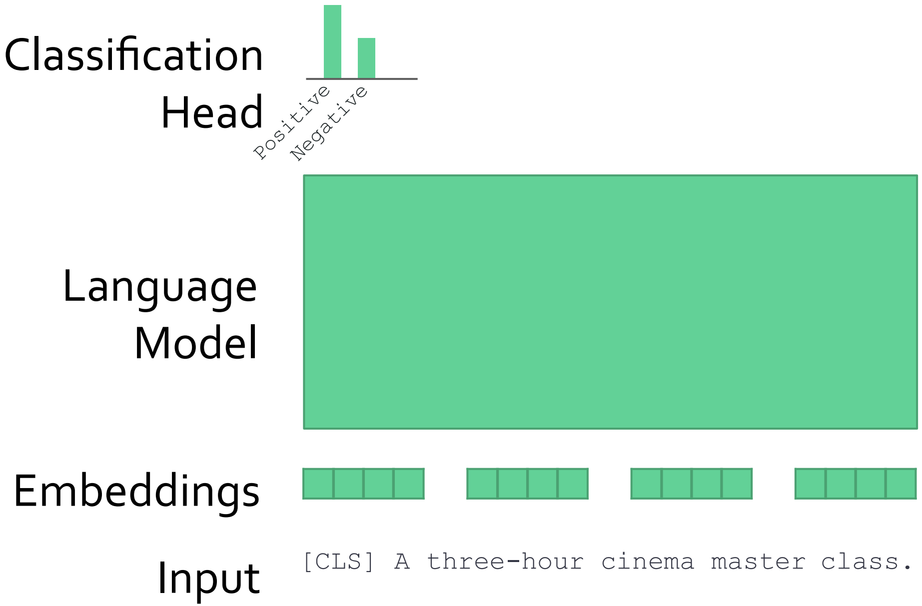
Fine-Tuning for Tasks



Translation | POS Tagging | Text classification | Language modeling

Fine-tuning Pre-trained Models

- Whole model tuning:
  - Run an optimization defined on your task data that updates **all** model parameters
- Head-tuning:
  - Run an optimization defined on your task data that updates the parameters of the model "head"



[ACL 2022 Tutorial Beltagy, Cohan, Logan IV, Min and Singh]

Parameter-efficient Fine-tuning

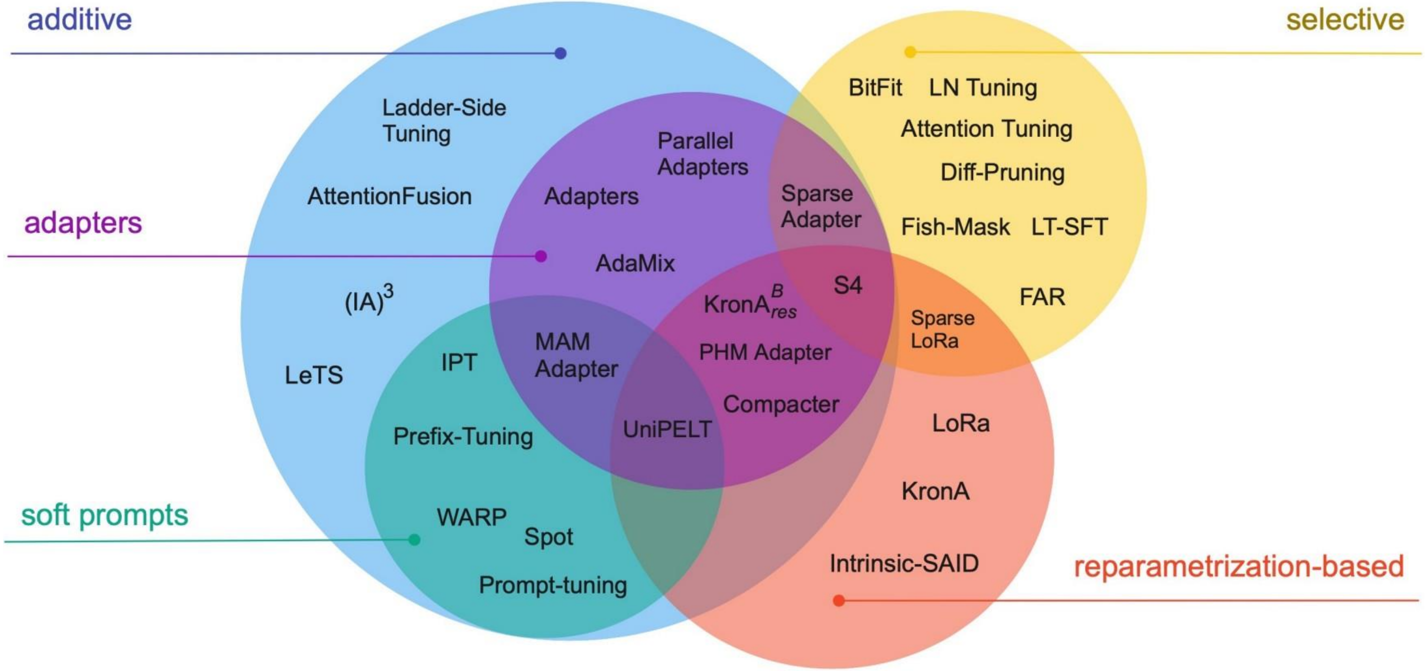
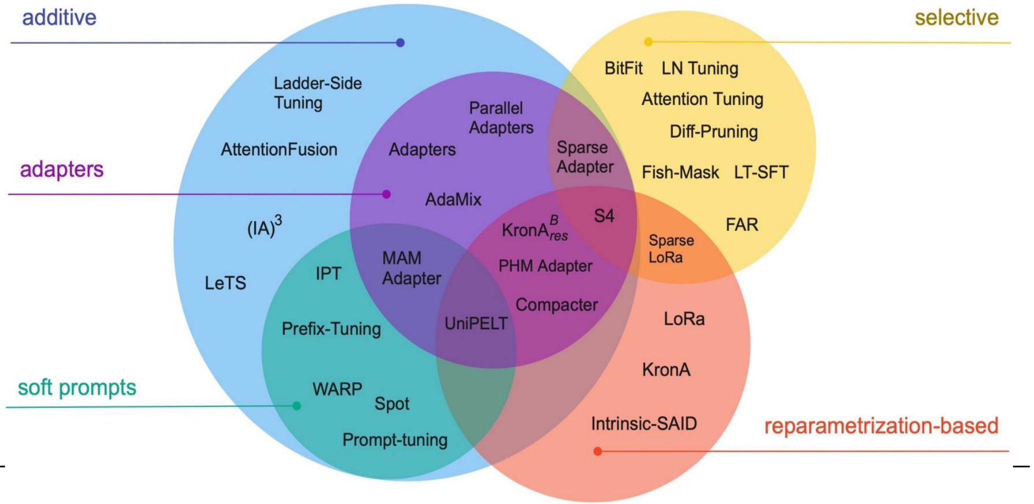


fig source https://arxiv.org/pdf/2303.15647.pdf

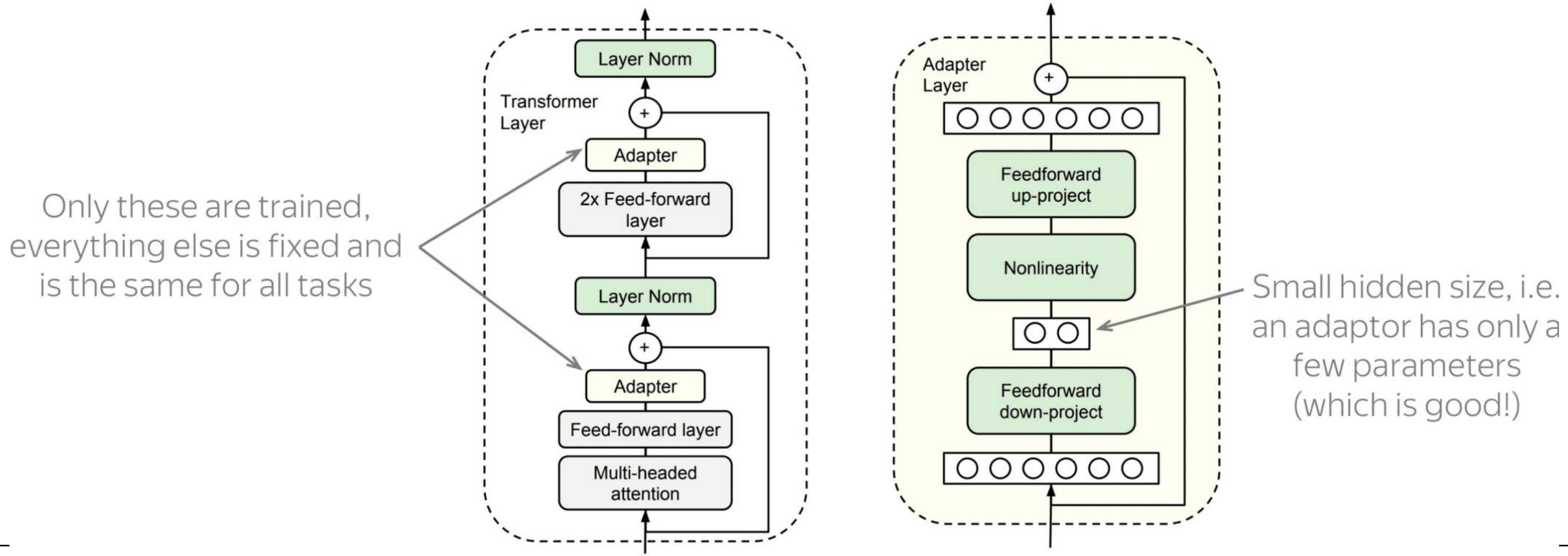
Parameter-efficient Fine-tuning: Adding Models

- Augmenting the existing pre-trained model with extra parameters or layers and training only the new parameters
- One commonly used method:
  - **Adapters**



Adapters

- **Idea:** train small sub-networks and only tune those.
  - Adapter layer projects to a low dimensional space to reduce parameters.
- No need to store a full model for each task, **only the adapter params**.



["Parameter-Efficient Transfer Learning for NLP", Houlsby et al., 2019.]

Question

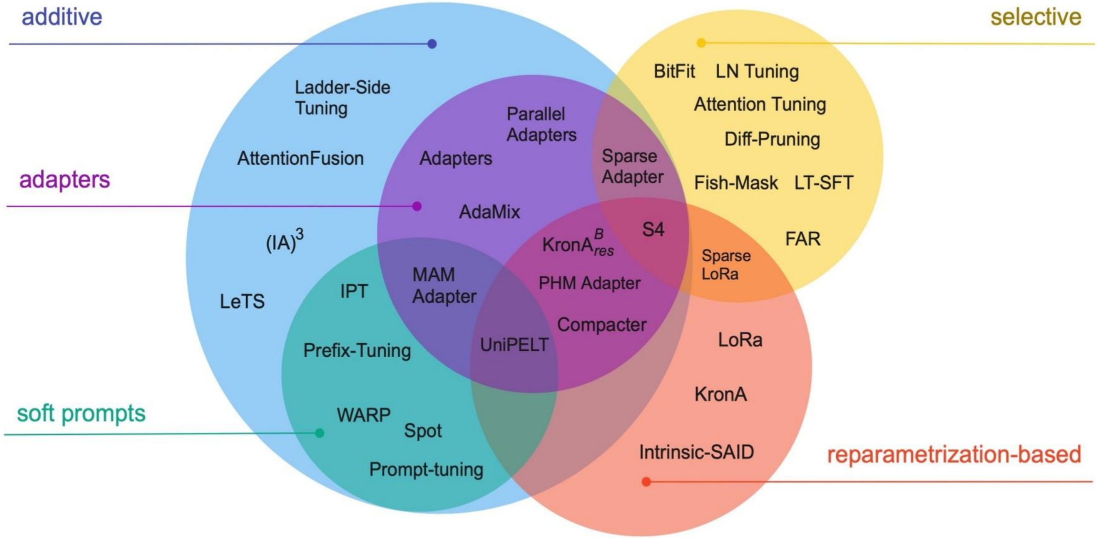
- Is parameter-efficient tuning more
  (1) computationally efficient
  (2) memory-efficient

than whole-model tuning?

Answer to (1) It is not faster!
You still need to do the entire forward and backward pass.
Answer to (2) It is more memory efficient.
You only need to keep the optimizer state for parameters that you are fine-tuning and not all the parameters.

Selective methods

- Selective methods fine-tune a subset of the existing parameters of the model.
- It could be a layer depth-based selection, layer type-based selection, or even individual parameter selection.



Ben Zaken et al., 2021: "BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models"

BitFit

- BitFit only tunes the bias terms in self-attention and MLP layers
- only updates about 0.05% of the model parameters

$$\mathbf{Q}^{m,\ell}(\mathbf{x}) = \mathbf{W}_q^{m,\ell}\mathbf{x} + \mathbf{b}_q^{m,\ell}$$
$$\mathbf{K}^{m,\ell}(\mathbf{x}) = \mathbf{W}_k^{m,\ell}\mathbf{x} + \mathbf{b}_k^{m,\ell}$$
$$\mathbf{V}^{m,\ell}(\mathbf{x}) = \mathbf{W}_v^{m,\ell}\mathbf{x} + \mathbf{b}_v^{m,\ell}$$

$$\mathbf{h}_2^\ell = \text{Dropout}\big(\mathbf{W}_{m_1}^\ell \cdot \mathbf{h}_1^\ell + \mathbf{b}_{m_1}^\ell\big) \quad (1)$$
$$\mathbf{h}_3^\ell = \mathbf{g}_{LN_1}^\ell \odot \frac{(\mathbf{h}_2^\ell + \mathbf{x}) - \mu}{\sigma} + \mathbf{b}_{LN_1}^\ell \quad (2)$$
$$\mathbf{h}_4^\ell = \text{GELU}\big(\mathbf{W}_{m_2}^\ell \cdot \mathbf{h}_3^\ell + \mathbf{b}_{m_2}^\ell\big) \quad (3)$$
$$\mathbf{h}_5^\ell = \text{Dropout}\big(\mathbf{W}_{m_3}^\ell \cdot \mathbf{h}_4^\ell + \mathbf{b}_{m_3}^\ell\big) \quad (4)$$
$$\text{out}^\ell = \mathbf{g}_{LN_2}^\ell \odot \frac{(\mathbf{h}_5^\ell + \mathbf{h}_3^\ell) - \mu}{\sigma} + \mathbf{b}_{LN_2}^\ell \quad (5)$$

Ben Zaken et al., 2021. “BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models”

BitFit

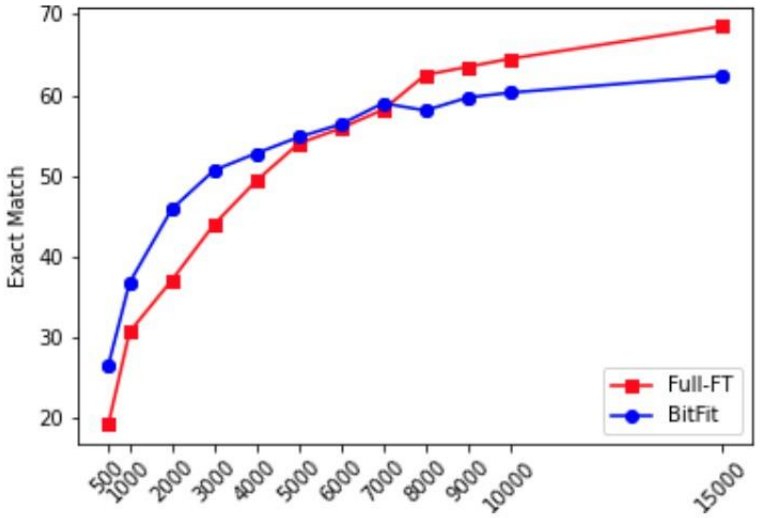


Figure 2: Comparison of BitFit and Full-FT with $\text{BERT}_{\text{BASE}}$ exact match score on SQuAD validation set.

Ben Zaken et al., 2021. “BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models”

Limitations of Pre-training, then Fine-tuning

- Often you need a large labeled data
  - Though more pre-training can reduce the need for labeled data

“I have an extremely large collection of clean labeled data”

“I have an extremely large collection of clean labeled data”

-- No one

Outline

Transformer LM (cont.)

Adaptation

Lecture Evaluation, Quiz

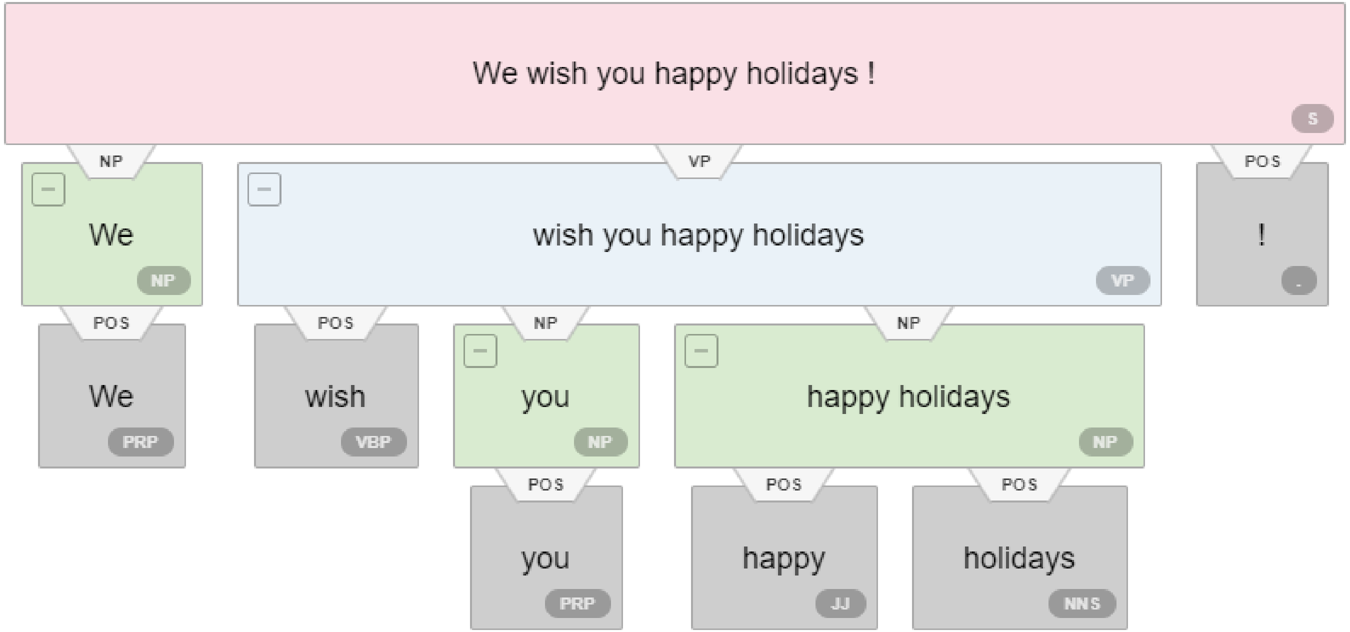
Lecture and Exercise Evaluation

TECHNISCHE UNIVERSITÄT DARMSTADT

Lecture

Exercise

Menti time


We wish you happy holidays !
S
NP
VP
POS
We
NP
wish you happy holidays
VP
!
.
POS
We
PRP
POS
wish
VBP
NP
you
NP
NP
happy holidays
NP
POS
you
PRP
POS
happy
JJ
POS
holidays
NNS