DeepLorIA

Mastering Large Language Models: Efficient Techniques for Fine-Tuning

Barbara Gendron-Audebert, PhD student (MosAIk team)

LORIA, Université de Lorraine, CNRS
DeepLorIA Network

January 15, 2025

About Me

2nd-year PhD student - Knowledge-Enhanced Language Models

Research Focus

- Controlled Conversational Models through Conversation-Dedicated Ontology
- *Keywords: Large Language Models (LLMs), Conversational Agents, Ontologies, Fine-Tuning*

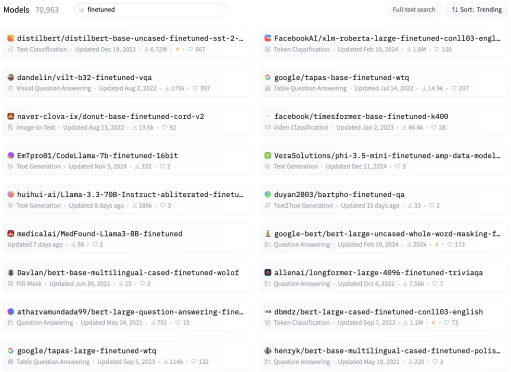
Experience in LLM fine-tuning

- Run pre-defined fine-tuning setups (Causal Language Modeling, Classification,...)
- Develop new fine-tuning pipelines to consider external knowledge
- Focus on textual modality

Experimenting LLM Fine-Tuning

Many successful LLM fine-tunings?

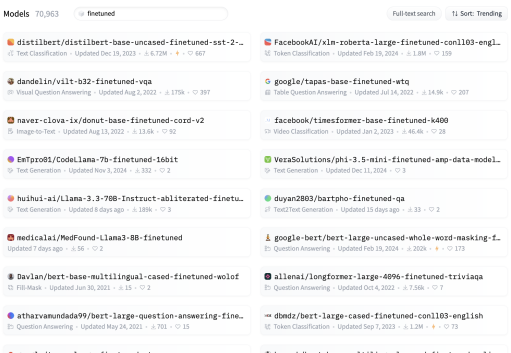
`https://huggingface.co/`



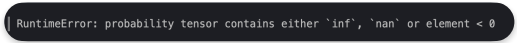
Experimenting LLM Fine-Tuning

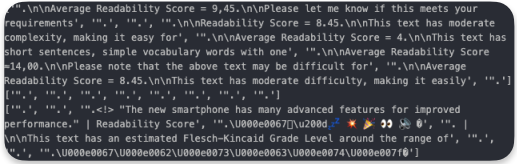
Many successful LLM fine-tunings?

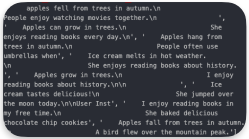
`https://huggingface.co/`

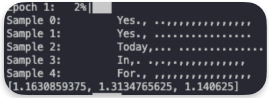


Fine-tuning LLMs in real life?



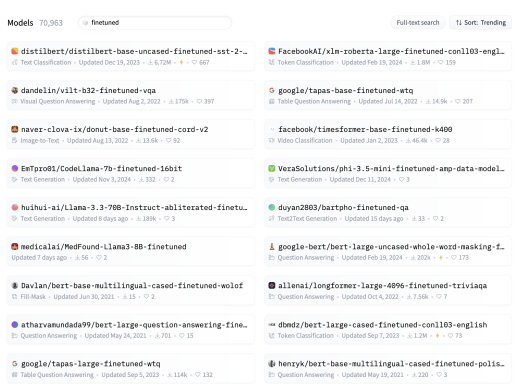






Experimenting LLM Fine-Tuning

Many successful LLM fine-tunings?
`https://huggingface.co/`



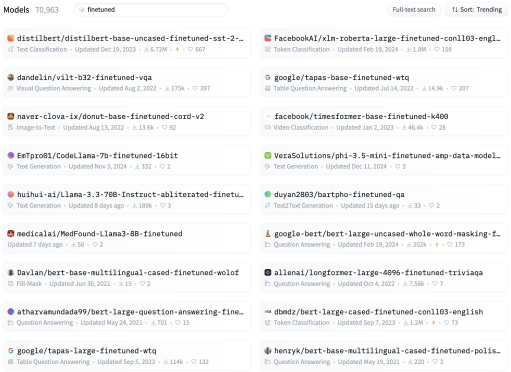
- Fine-tuning LLMs relies on obscure "magic formulas"

Fine-tuning LLMs in real life?

- Fine-tuning LLMs is hard

Experimenting LLM Fine-Tuning

Many successful LLM fine-tunings?
`https://huggingface.co/`



✗ Fine-tuning LLMs relies on obscure "magic formulas"

Fine-tuning LLMs in real life?

✗ Fine-tuning LLMs is hard

Deep-Learning-Based Sequence Modeling: Recurrent Models (1)

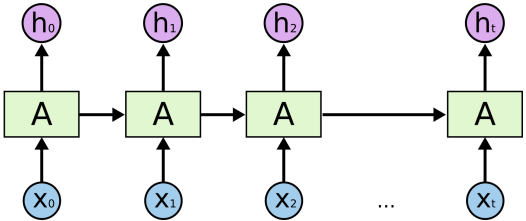


Figure 1 – Illustration of the Recurrent Neural Network (RNN, [18, 10]) architecture[1]

- ✓ Keep token order
- ✓ Handle variable-length sequences
- ✓ Parameter sharing across the sequence

- ✗ Exploding and vanishing gradient
- ✗ Long-term dependencies
- ✗ Slow computing, no parallelization

[1] Olah, C. (2015). Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Deep-Learning-Based Sequence Modeling: Recurrent Models (2)

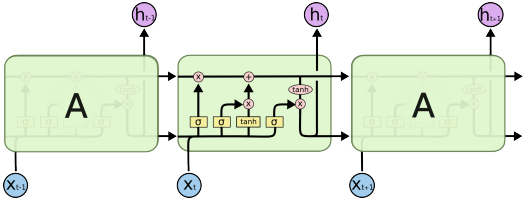


Figure 2 – Illustration of the Long Short Term Memory Neural Network (LSTM, [7]) architecture[2]

- ✓ Keep token order
- ✓ Handle variable-length sequences
- ✓ Parameter sharing across the sequence
- ✓ No exploding/vanishing gradient
- ✓ Long-term dependencies
- ✗ Slow computing, no parallelization

[2] Olah, C. (2015). Understanding LSTM Networks. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Attention Principle [1] and Transformer Model [20]

From Transformer-Based Models to Large Language Models (LLMs)

LLMs scale Transformers by stacking encoders and/or decoders together

- Parallelizable and optimized versions exist (e.g. quantization)
- Enable deeper and broader knowledge representation
- Large context window allows for more accurate generation

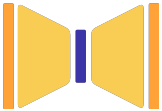


Figure 3 – Full Transformer

Figure 4 – Encoder

Figure 5 – Decoder

From Transformer-Based Models to Large Language Models (LLMs)

LLMs scale Transformers by stacking encoders and/or decoders together

- Parallelizable and optimized versions exist (e.g. quantization)
- Enable deeper and broader knowledge representation
- Large context window allows for more accurate generation

Fine-tuning adapts an LLM to a specific task through further parameter updates

- Can be performed with any LLM structure, but:
  - There are *required structures* for some specific tasks
  - There are *preferred models* for some specific tasks

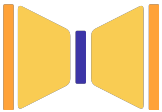
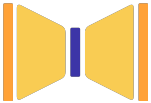


Figure 3 – Full Transformer

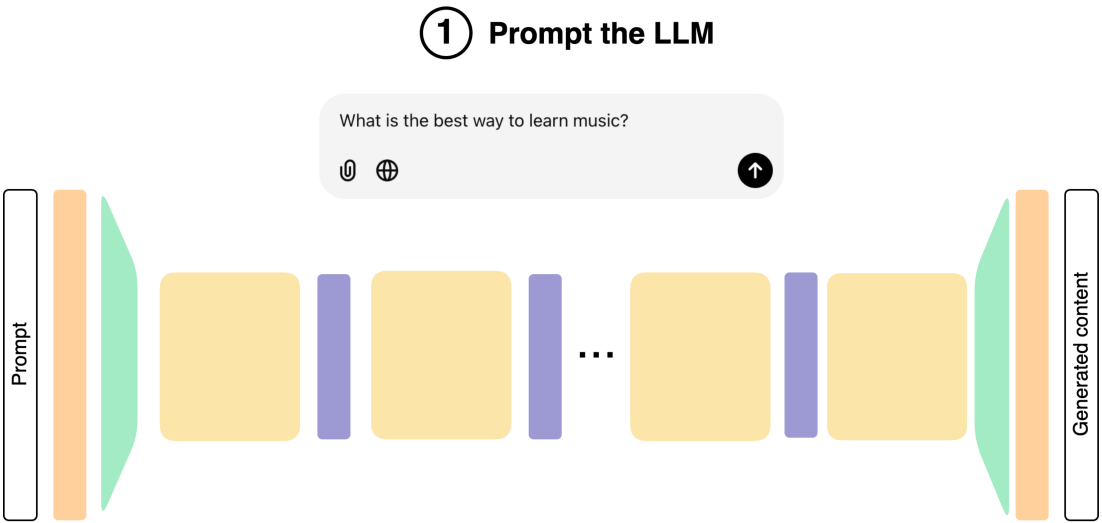
Figure 4 – Encoder

Figure 5 – Decoder

What Use-Cases of LLMs?

Model Structure	Task Examples	Model examples
	• Summarization • Machine Translation • Question Answering	• BART [11], mBART [12] • T5 [17], Flan-T5 [3] • bert2BERT [2]
	• Sequence Embedding • Text Classification • Regression	• BERT [5], mBERT [15] • RoBERTa [13] • DistilBERT [19]
	• Text Completion • Text Generation • Code Generation	• GPT-3.5, GPT-4o • Llama-3 [6] • Qwen2.5 [16]

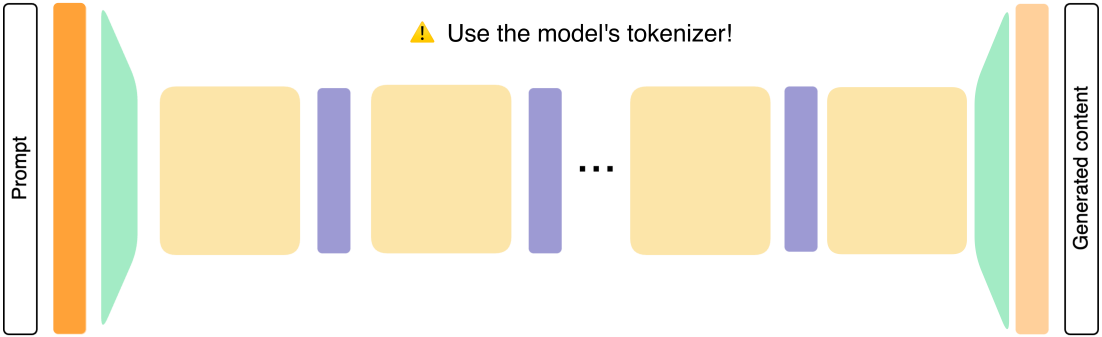
Inside a Decoder-Only LLM



② Tokenize the Prompt Content

"What is the best way to learn music?"
[531, 9, 45, 22, 3316, 2444, 34, 2172, 334]

⚠ Use the model's tokenizer!

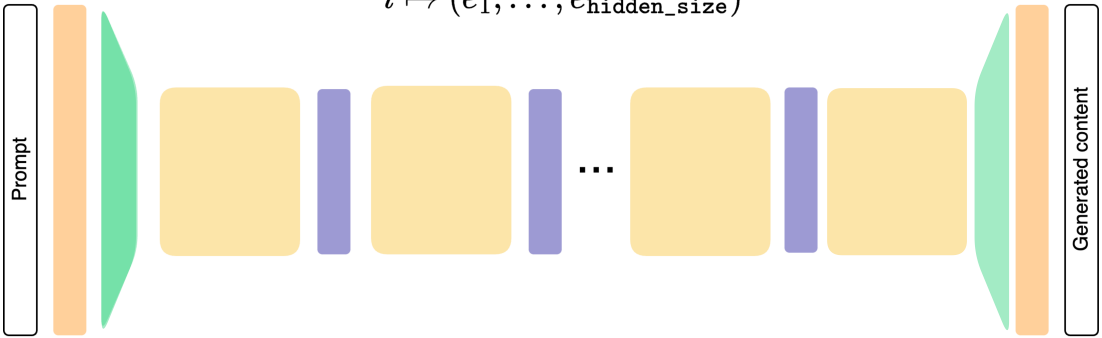


Inside a Decoder-Only LLM

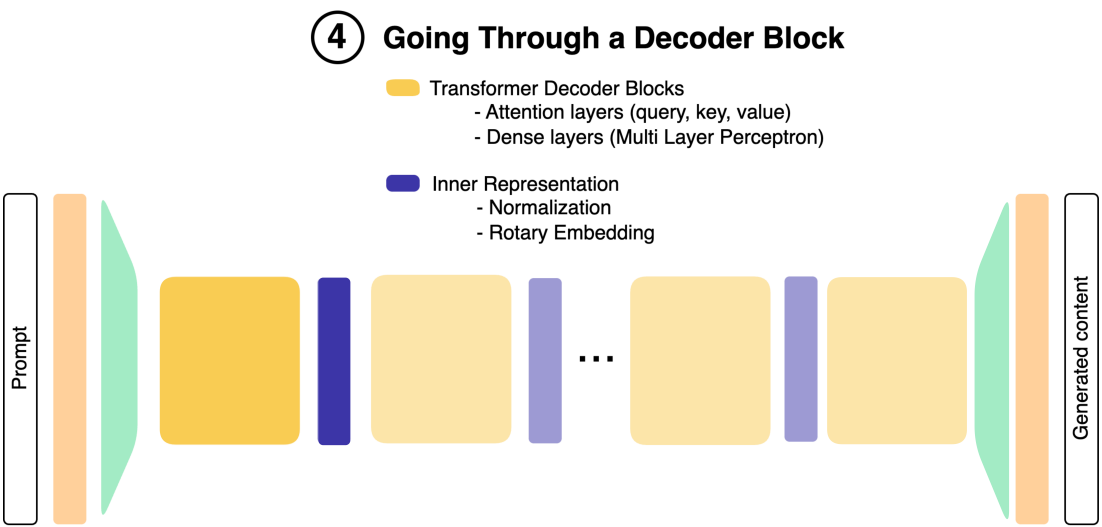
③ Apply an Embedding layer

`Embedding(vocab_size, hidden_size)`

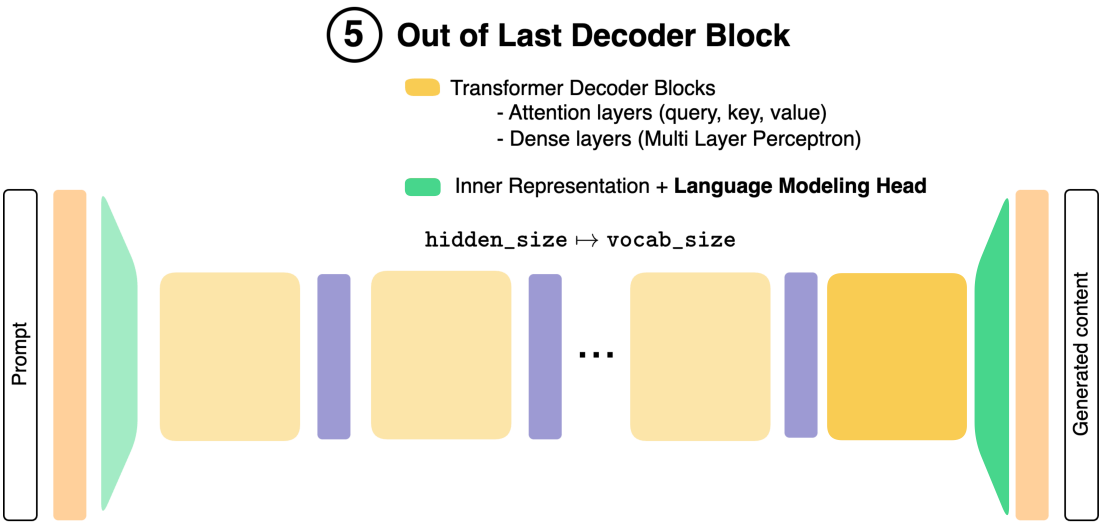
$$i \mapsto (e_1, \ldots, e_{\texttt{hidden\_size}})$$



Inside a Decoder-Only LLM

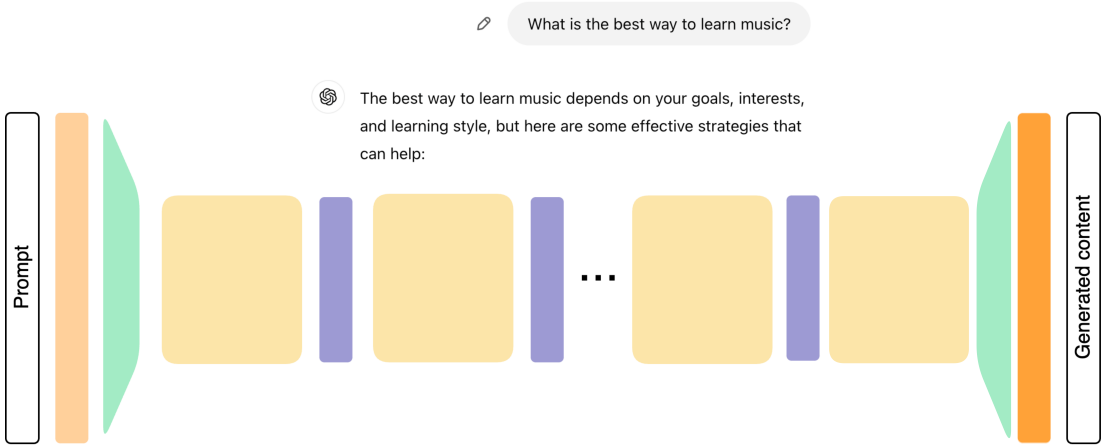


Inside a Decoder-Only LLM

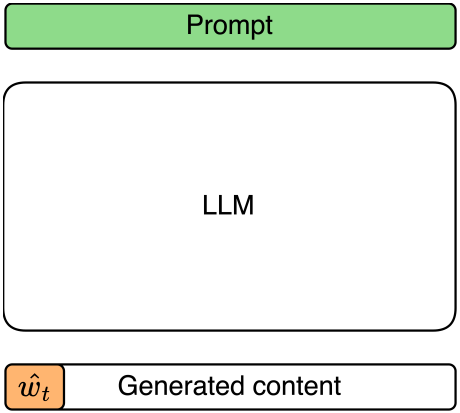


Inside a Decoder-Only LLM

6 Decode Generated Tokens in Natural Language



The Autoregressive Principle

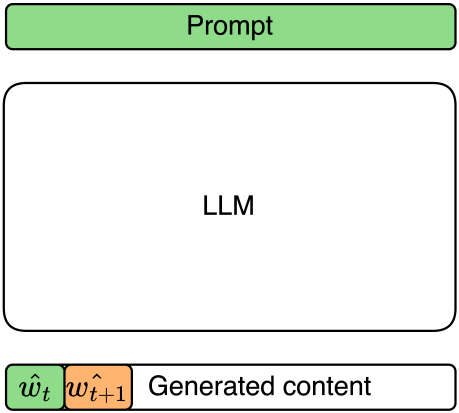


Context

Predicted token

$$\hat{w}_t = \arg\max_{w_i} P(w_i | w_1, w_2, \ldots, w_{t-1})$$

The Autoregressive Principle



Context
Predicted token

$$\hat{w_{t+1}} = \arg\max_{w_i} P(w_i | w_1, w_2, \ldots, w_t)$$

- **Top p**: adjust the range of tokens to consider regarding their probability.
- **Top k**: choose among the k more likely tokens. Default is 1 (most likely token).
- **Temperature:** control "creativity" by adding random noise to select less likely tokens.

How Large are Large Language Models?

Model	Parameters	Layers	Context Size
Gemini 1.5	200B?	-	10M
GPT-4 turbo	1.8T	120	128k
Claude 2.1	12B	-	200k

Table: Some *closed-source* model specifications

Model	Parameters	Layers	Context Size
Llama 3.3	70B	80	128k
Phi 4	14B	40	16k
Qwen 2.5	0.5B-72B	24-80	32k - 128k
Mistral-v0.3	7B	32	32k

Table: Some *open-weights* model specifications

Side remark: most LLMs called "open-source" are actually open-weights!

How Large are Large Language Models?

Model	Parameters	Layers	Context Size
Gemini 1.5	200B?	-	10M
GPT-4 turbo	1.8T	120	128k
Claude 2.1	12B	-	200k

Table: Some *closed-source* model specifications

Model	Parameters	Layers	Context Size
Llama 3.3	70B	80	128k
Phi 4	14B	40	16k
Qwen 2.5	0.5B-72B	24-80	32k - 128k
Mistral-v0.3	7B	32	32k

Table: Some *open-weights* model specifications

Side remark: most LLMs called "open-source" are actually open-weights!

- Most models involve several gigabytes in RAM GPU to perform inference
- Updating each parameter value during fine-tuning would be too costly

$\longrightarrow$ fine-tuning should be **efficient**

Fine-Tuning as an *Affinage*: the Cheese Analogy

Fine-Tuning as an *Affinage*: the Cheese Analogy

Cow / Raw milk

Milk preparation

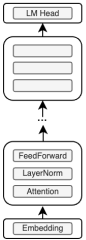
Maturing

Fine-Tuning as an *Affinage*: the Cheese Analogy

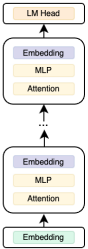
Cow / Raw milk

Milk preparation

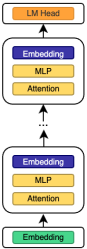
Maturing



Textual data / Raw model

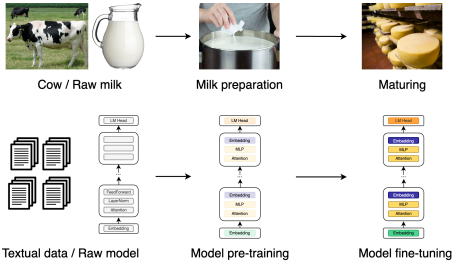


Model pre-training



Model fine-tuning

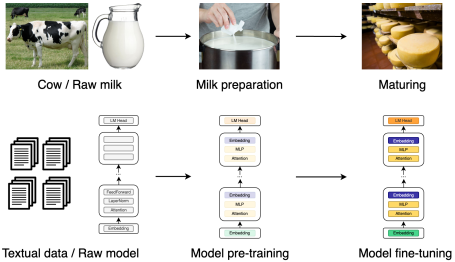
Fine-Tuning as an *Affinage*: the Cheese Analogy



Fine-tuning, as cheese maturing phase, modifies **in place** its given instance.

- Ensure fine-tuning is performed in the right conditions
- Check pre-trained baseline reliability
- Test several (adapted) pre-trained baselines for one fine-tuning experiment

Fine-Tuning as an *Affinage*: the Cheese Analogy



Fine-tuning, as cheese maturing phase, modifies **in place** its given instance.

- Ensure fine-tuning is performed in the right conditions
- Check pre-trained baseline reliability
- Test several (adapted) pre-trained baselines for one fine-tuning experiment

$\longrightarrow$ fine-tuning should be **stable** and **consistent** with pre-trained baseline

Efficient Fine-Tuning With Adapters [8]

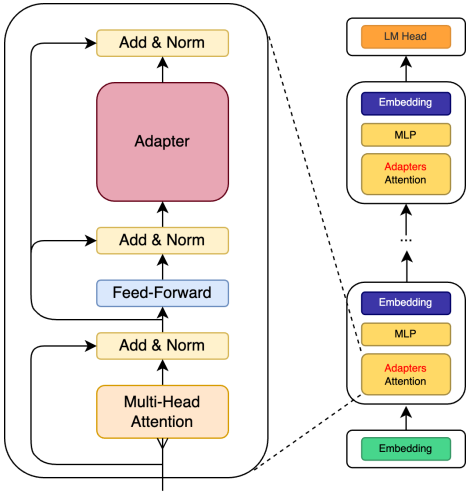


Figure 6 – An attention block with adapters

Adapters layers are inserted to enable parameter-efficient fine-tuning[a]

Common usage:

- Freeze all pre-trained model layers
- Insert trainable MLP layers into attention blocks (query, value) and/or model head

[a] https://huggingface.co/PEFT
[b] https://adapterhub.ml/

Efficient Fine-Tuning With Adapters [8]

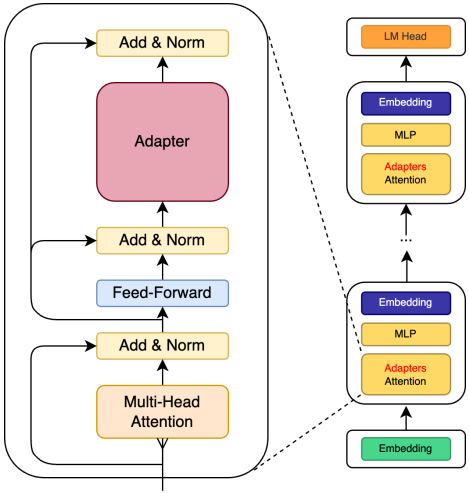


Figure 6 – An attention block with adapters

Adapters layers are inserted to enable parameter-efficient fine-tuning[a]

Common usage:

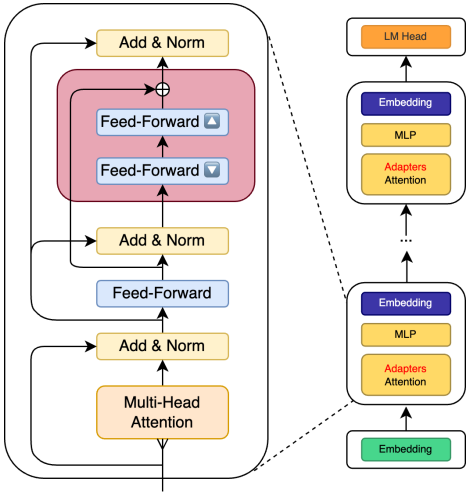
- Freeze all pre-trained model layers
- Insert trainable MLP layers into attention blocks (query, value) and/or model head

Advantages:

- Fewer param. updates than full fine-tuning
- Helps reduce catastrophic forgetting [14]
- Easy to share fine-tuned models[b]

[a] `https://huggingface.co/PEFT`
[b] `https://adapterhub.ml/`

Efficient Fine-Tuning With LoRA Adapters [9]



LoRA: Low-Rank Adaptation
A specific adapter block structure

- Information from the model can be represented (almost) equally well in a lower dimensional space
- The rank r of the lower dim. space should be determined by hyperparameter tuning
- Quantized versions: QLoRA [4]

Figure 7 – An attention block with LoRA adapters

Wrap-Up

Thanks for your attention!

Figure 8 – Practical session: `https://github.com/B-Gendron/tutorial-deeploria/lab/`

References I

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *ICLR* abs/1409.0473 (2014). URL: https://api.semanticscholar.org/CorpusID:11212020.

[2] Cheng Chen et al. *bert2BERT: Towards Reusable Pretrained Language Models*. 2021. arXiv: 2110.07143 [cs.CL]. URL: https://arxiv.org/abs/2110.07143.

[3] Hyung Won Chung et al. *Scaling Instruction-Finetuned Language Models*. 2022. arXiv: 2210.11416 [cs.LG]. URL: https://arxiv.org/abs/2210.11416.

[4] Tim Dettmers et al. *QLoRA: Efficient Finetuning of Quantized LLMs*. 2023. arXiv: 2305.14314 [cs.LG]. URL: https://arxiv.org/abs/2305.14314.

[5] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *NAACL*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, 2019.

References II

[6] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI]. URL: https://arxiv.org/abs/2407.21783.

[7] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-term Memory". In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[8] Neil Houlsby et al. *Parameter-Efficient Transfer Learning for NLP*. 2019. arXiv: 1902.00751 [cs.LG]. URL: https://arxiv.org/abs/1902.00751.

[9] Edward J. Hu et al. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685 [cs.CL]. URL: https://arxiv.org/abs/2106.09685.

[10] M I Jordan. *Serial order: a parallel distributed processing approach. Technical report, June 1985-March 1986*. Tech. rep. California Univ., San Diego, La Jolla (USA). Inst. for Cognitive Science, May 1986. URL: https://www.osti.gov/biblio/6910294.

References III

[11] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *ACL*. Ed. by Dan Jurafsky et al. Association for Computational Linguistics, 2020.

[12] Yinhan Liu et al. *Multilingual Denoising Pre-training for Neural Machine Translation*. 2020. arXiv: 2001.08210 [cs.CL]. URL: https://arxiv.org/abs/2001.08210.

[13] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *ArXiv* (2019).

[14] Jonas Pfeiffer et al. "AdapterFusion: Non-Destructive Task Composition for Transfer Learning". In: *CoRR* abs/2005.00247 (2020). arXiv: 2005.00247. URL: https://arxiv.org/abs/2005.00247.

[15] Telmo Pires, Eva Schlinger, and Dan Garrette. "How multilingual is Multilingual BERT?" In: *ArXiv* (2019).

References IV

[16] Qwen et al. *Qwen2.5 Technical Report*. 2025. arXiv: 2412.15115 [cs.CL]. URL: https://arxiv.org/abs/2412.15115.

[17] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *ArXiv* abs/1910.10683 (2019).

[18] David E. Rumelhart and James L. McClelland. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. 1987, pp. 318–362.

[19] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *CoRR* abs/1910.01108 (2019). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108.

[20] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* (2017).