ML in PL
CONFERENCE 2024
7 - 10 NOVEMBER / WARSAW, POLAND

Klaudia Bałazy

NVIDIA | Jagiellonian University

Contributed Talk 11:

Efficient Fine-Tuning of LLMs: Exploring PEFT Methods and LoRA-XS Insights

Friday / 8 November   15:00 - 15:25   Lecture Hall A

About me & about the talk

LoRA-XS: Low-Rank Adaptation with Extremely Small Number of Parameters

Klaudia Bałazy[*1] **Mohammadreza Banaei**[*2] **Karl Aberer**[2] **Jacek Tabor**[1]

[1]Jagiellonian University, [2]EPFL

[*]Equal contribution.

Klaudia Bałazy
Deep Learning Engineer & AI Researcher

JAGIELLONIAN UNIVERSITY IN KRAKÓW

group of machine learning research gmum

Agenda

1. What is PEFT?
2. Why do we need it?
3. What are the PEFT approaches?
4. Our PEFT proposal: LoRA-XS

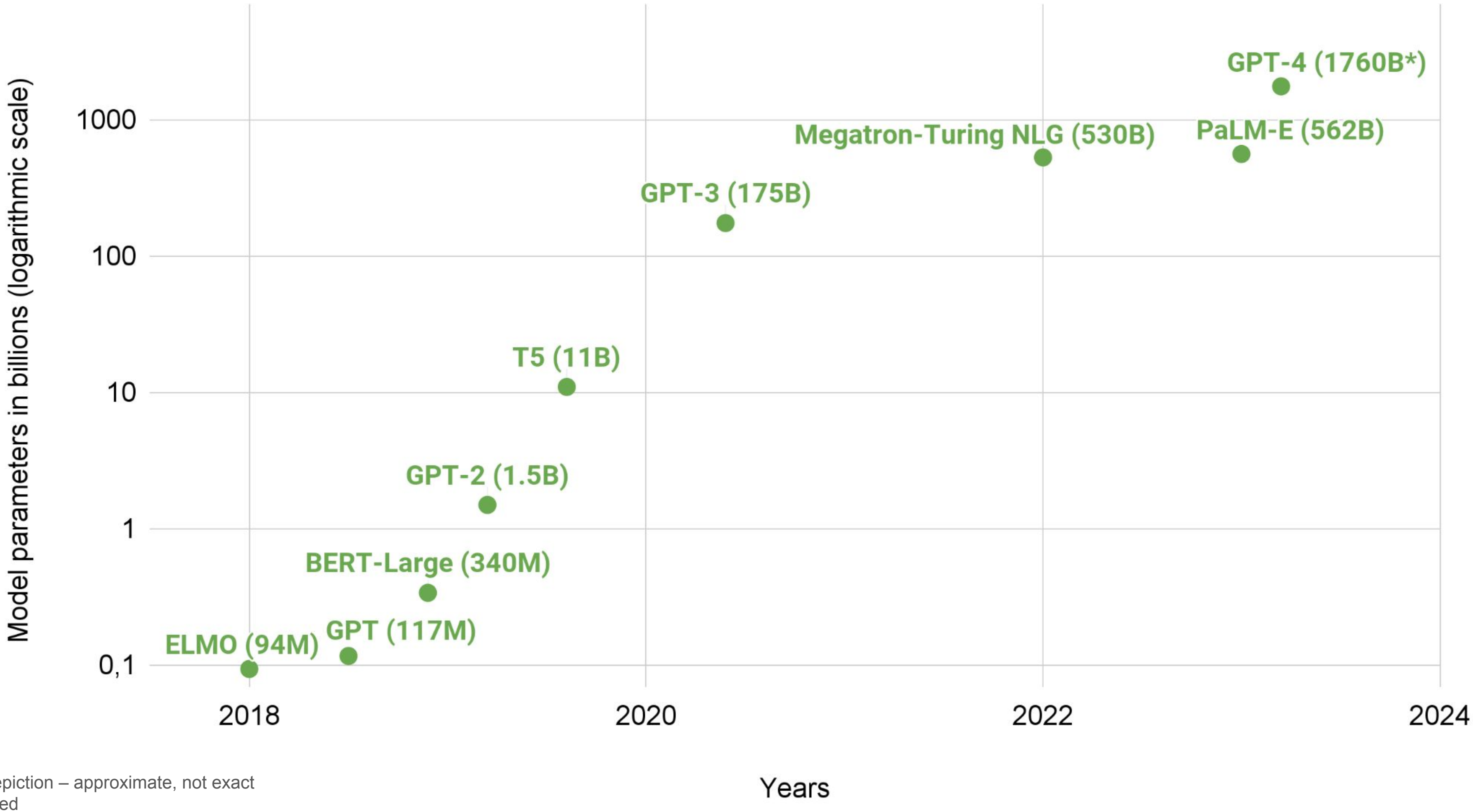
Agenda

1. **What** is PEFT? **Parameter-Efficient Fine-Tuning**
2. Why do we need it?
3. What are the PEFT approaches?
4. Our PEFT proposal: LoRA-XS

Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18],[19],[20],[21],[23],[24]

Agenda

1. What is PEFT? Parameter-Efficient Fine-Tuning
2. **Why** do we need it?
3. What are the PEFT approaches?
4. Our PEFT proposal: LoRA-XS



Trend depiction – approximate, not exact
*Unverified
Sources: [1],[2],[3],[4],[5],[6],[7],[8],[9]

Total Training Memory ≈

Model Weights

+ Activations

+ (Optimizer States + Gradients) * Number of Trainable Parameters

Method	Bits	7B	13B	30B	70B	110B	8x7B
Full	AMP	120GB	240GB	600GB	1200GB	2000GB	900GB
Full	16	60GB	120GB	300GB	600GB	900GB	400GB
Freeze	16	20GB	40GB	80GB	200GB	360GB	160GB
LoRA/GaLore/BAdam	16	16GB	32GB	64GB	160GB	240GB	120GB
QLoRA	8	10GB	20GB	40GB	80GB	140GB	60GB
QLoRA	4	6GB	12GB	24GB	48GB	72GB	30GB
QLoRA	2	4GB	8GB	16GB	24GB	48GB	18GB

* estimated
Source: https://github.com/hiyouga/LLaMA-Factory#hardware-requirement
References: [17],[22],[25],[26],[27]

Agenda

1. What is PEFT? Parameter-Efficient Fine-Tuning
2. Why do we need it?
3. What are the **PEFT approaches**?
4. Our PEFT proposal: LoRA-XS

PEFT methods

ULMFiT

2015

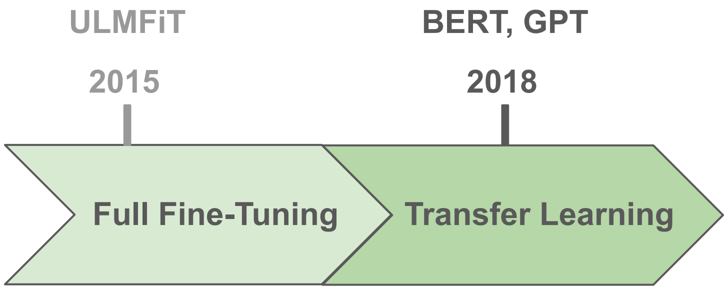
Full Fine-Tuning

➕ Comprehensive Learning

➖ High Computational Cost

Sources: [10]

PEFT methods



- \+ Reduced Training Cost
- − Inflexibility in Frozen Layers

Sources: [3],[4],[10]

PEFT methods

+ Efficient Parameter Use
− Additional Inference Overhead

Sources: [3],[4],[10],[11],[18]

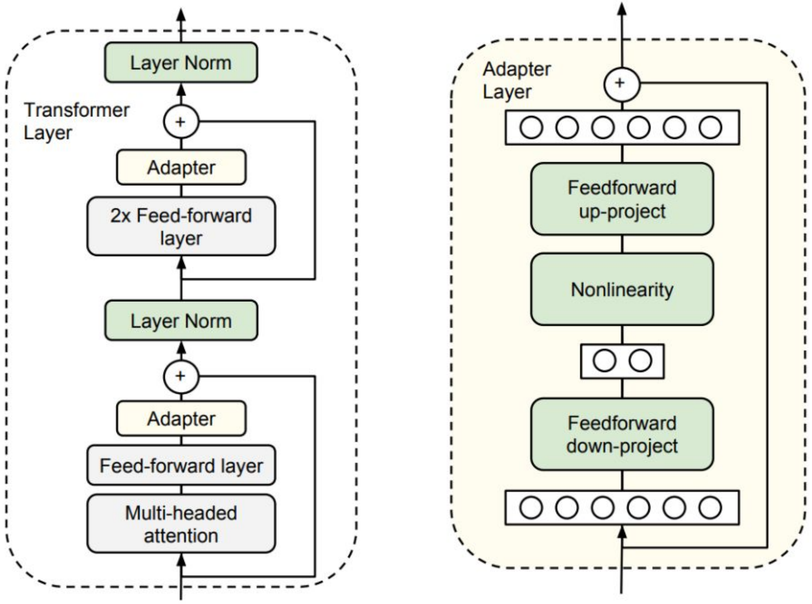
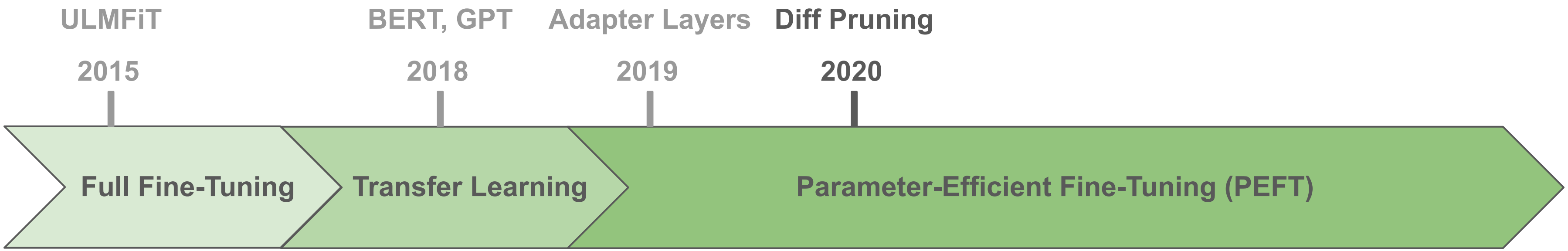


Figure 2. Architecture of the adapter module and its integration with the Transformer. **Left:** We add the adapter module twice to each Transformer layer: after the projection following multi-headed attention and after the two feed-forward layers. **Right:** The adapter consists of a bottleneck which contains few parameters relative to the attention and feedforward layers in the original model. The adapter also contains a skip-connection. During adapter tuning, the green layers are trained on the downstream data, this includes the adapter, the layer normalization parameters, and the final classification layer (not shown in the figure).

Sources: Houlsby, Neil, et al. "Parameter-efficient transfer learning for NLP." International conference on machine learning. PMLR, 2019.

PEFT methods

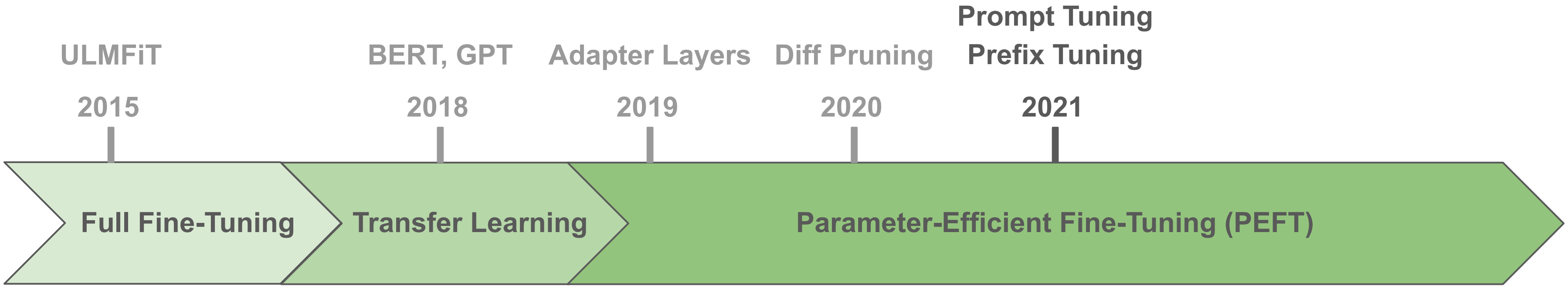


\+ Efficient Multi Task Deployment (mobile applications)

− Higher memory consumption compared to traditional fine-tuning

Sources: [3],[4],[10],[11],[12],[18]

PEFT methods



\+ Zero Training of Core Model

\- Performance Variability

Sources: [3],[4],[10],[11],[12],[13],[14],[18]

Hard Prompt Tuning

```
1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"

2) "English: '{english_sentence}' | German: {german_translation}"

3) "From English to German: '{english_sentence}' -> {german_translation}"
```

Sources:
Raschka, S. (2023, April 30). Understanding Parameter-Efficient LLM Finetuning: Prompt Tuning and Prefix Tuning. The Machine Learning Magazine. https://magazine.sebastianraschka.com/p/understanding-parameter-efficient
Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).
Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).

```
1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"

2) "English: '{english_sentence}' | German: {german_translation}"

3) "From English to German: '{english_sentence}' -> {german_translation}"
```

Hard Prompt Tuning

```python
soft_prompt = torch.nn.Parameter( # Make tensor trainable
    torch.rand(num_tokens, embed_dim)) # Initialize soft prompt tensor

def input_with_soft_prompt(x, soft_prompt) :
    x = concatenate([soft_prompt, x], # Prepend soft prompt to input
                    dim=seq_len)
    return x

# train soft prompt tensor via gradient descent
train(model(input_with_soft_prompt(x)))

# use model with soft prompts
model(input_with_soft_prompt(x))
```

Soft Prompt Tuning

Sources:
Raschka, S. (2023, April 30). Understanding Parameter-Efficient LLM Finetuning: Prompt Tuning and Prefix Tuning. The Machine Learning Magazine. https://magazine.sebastianraschka.com/p/understanding-parameter-efficient
Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).
Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).

Hard Prompt Tuning

```
1) "Translate the English sentence '{english_sentence}' into German: {german_translation}"

2) "English: '{english_sentence}' | German: {german_translation}"

3) "From English to German: '{english_sentence}' -> {german_translation}"
```

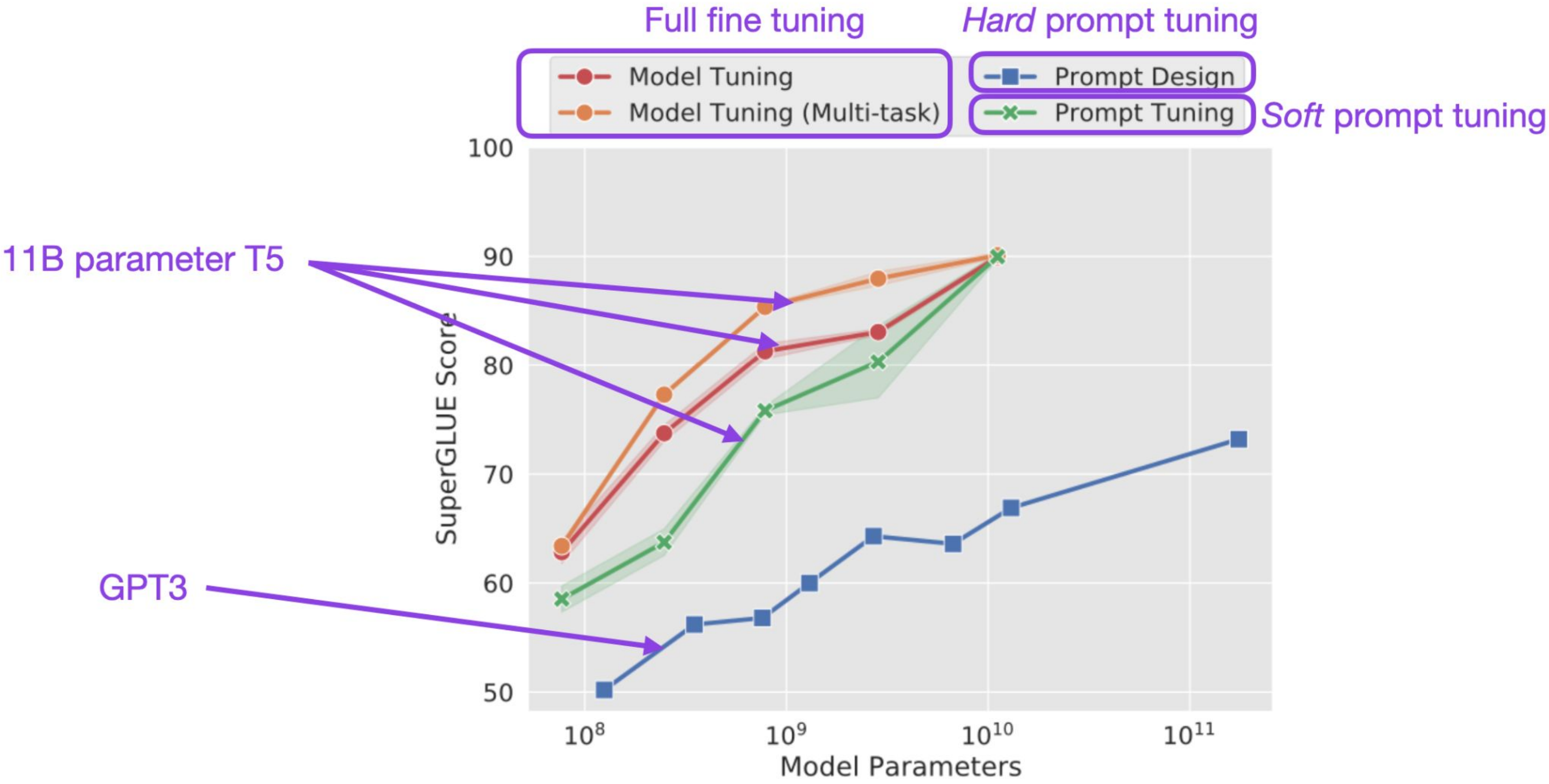
Soft Prompt Tuning

```python
soft_prompt = torch.nn.Parameter( # Make tensor trainable
    torch.rand(num_tokens, embed_dim)) # Initialize soft prompt tensor

def input_with_soft_prompt(x, soft_prompt) :
    x = concatenate([soft_prompt, x], # Prepend soft prompt to input
                    dim=seq_len)
    return x

# train soft prompt tensor via gradient descent
train(model(input_with_soft_prompt(x)))

# use model with soft prompts
model(input_with_soft_prompt(x))
```

Prefix Tuning

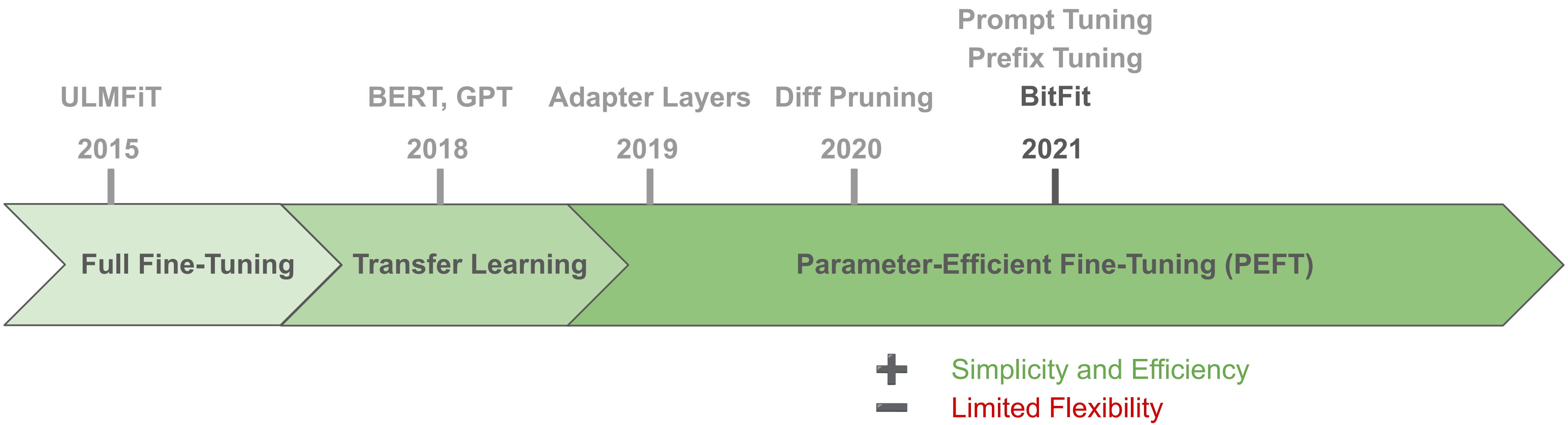
```python
def transformer_block_with_prefix(x, soft_prompt):
    soft_prompt = FullyConnectedLayers(soft_prompt)  # Prefix
    x = concatenate([soft_prompt, x],                # Prefix
                    dim=seq_len)                     # Prefix
    residual = x
    x = self_attention(x)
    x = LayerNorm(x + residual)
    residual = x
    x = FullyConnectedLayers(x)
    x = LayerNorm(x + residual)
    return x
```

Sources:
Raschka, S. (2023, April 30). Understanding Parameter-Efficient LLM Finetuning: Prompt Tuning and Prefix Tuning. The Machine Learning Magazine. https://magazine.sebastianraschka.com/p/understanding-parameter-efficient
Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).
Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).



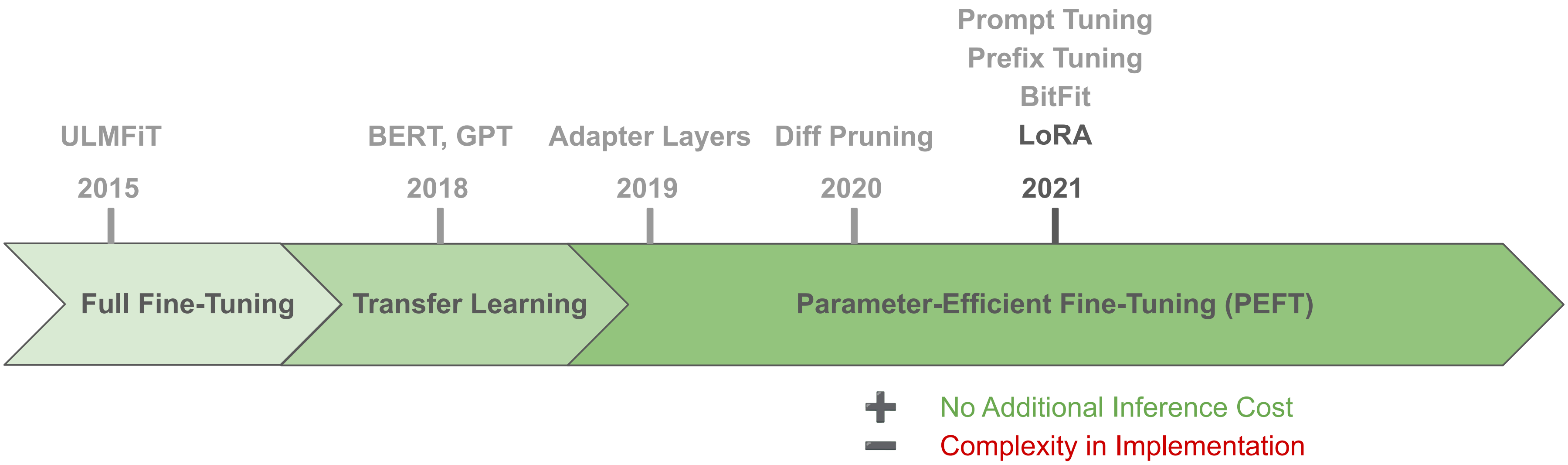
Sources:
Raschka, S. (2023, April 30). Understanding Parameter-Efficient LLM Finetuning: Prompt Tuning and Prefix Tuning. The Machine Learning Magazine. https://magazine.sebastianraschka.com/p/understanding-parameter-efficient
Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).
Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).

PEFT methods



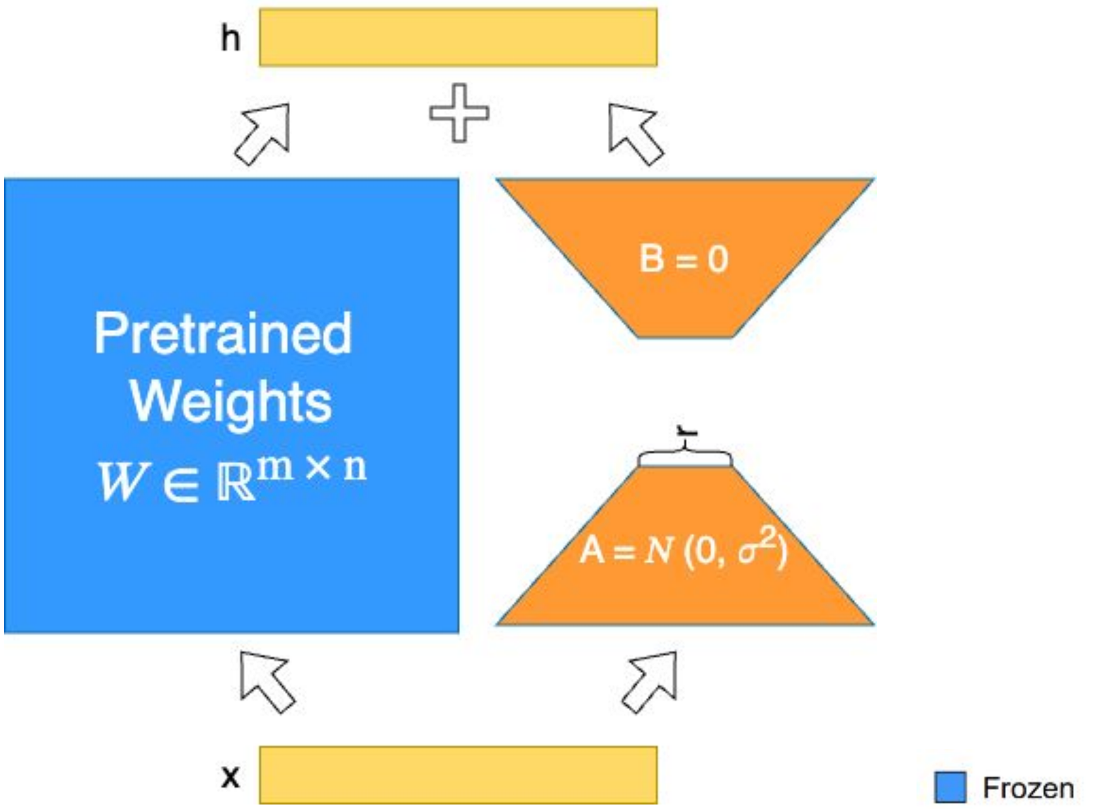
Sources: [3],[4],[10],[11],[12],[13],[14],[16],[18]

PEFT methods



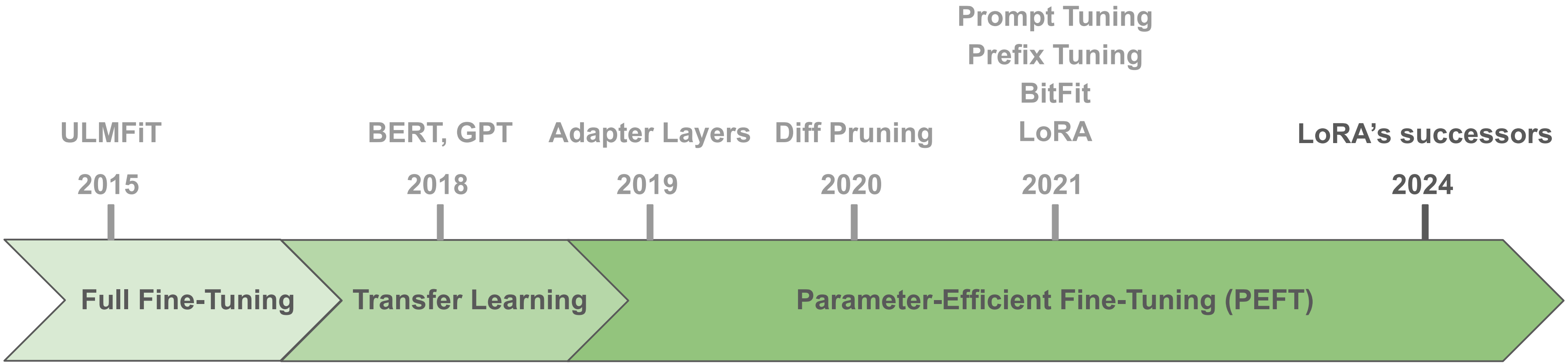
Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18]

LoRA



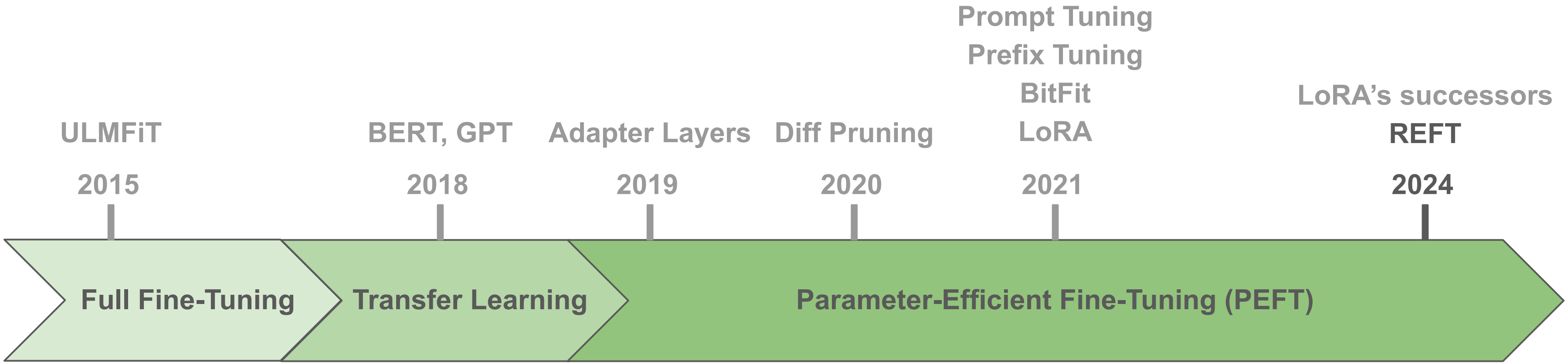
Sources: Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).

PEFT methods



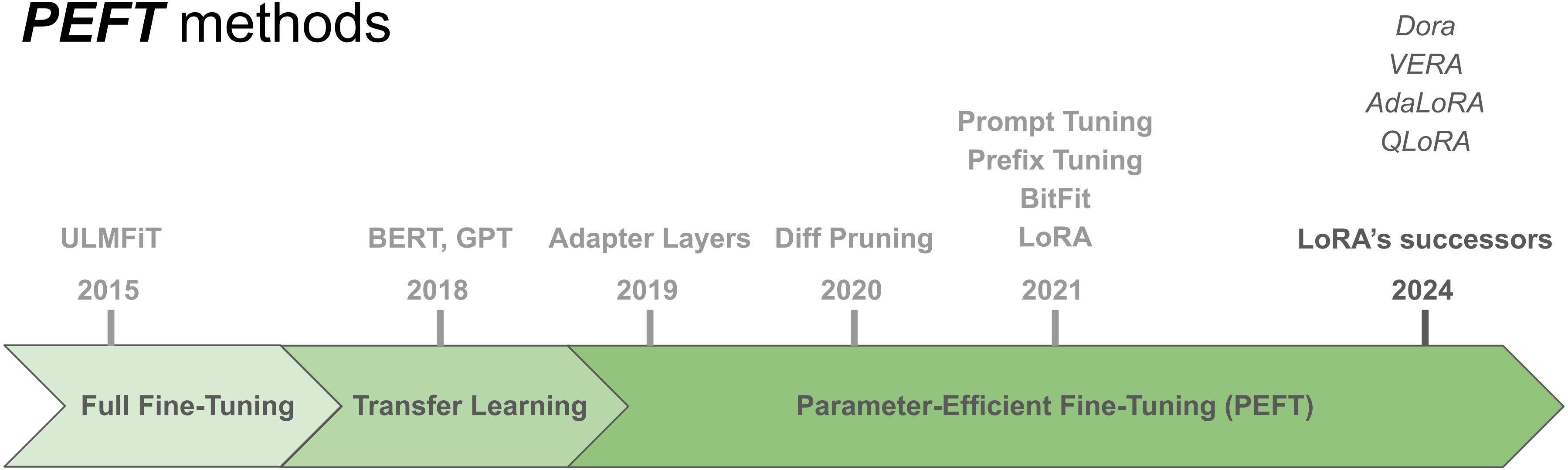
Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18],[19],[20],[21]

PEFT methods



Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18],[23]

PEFT methods

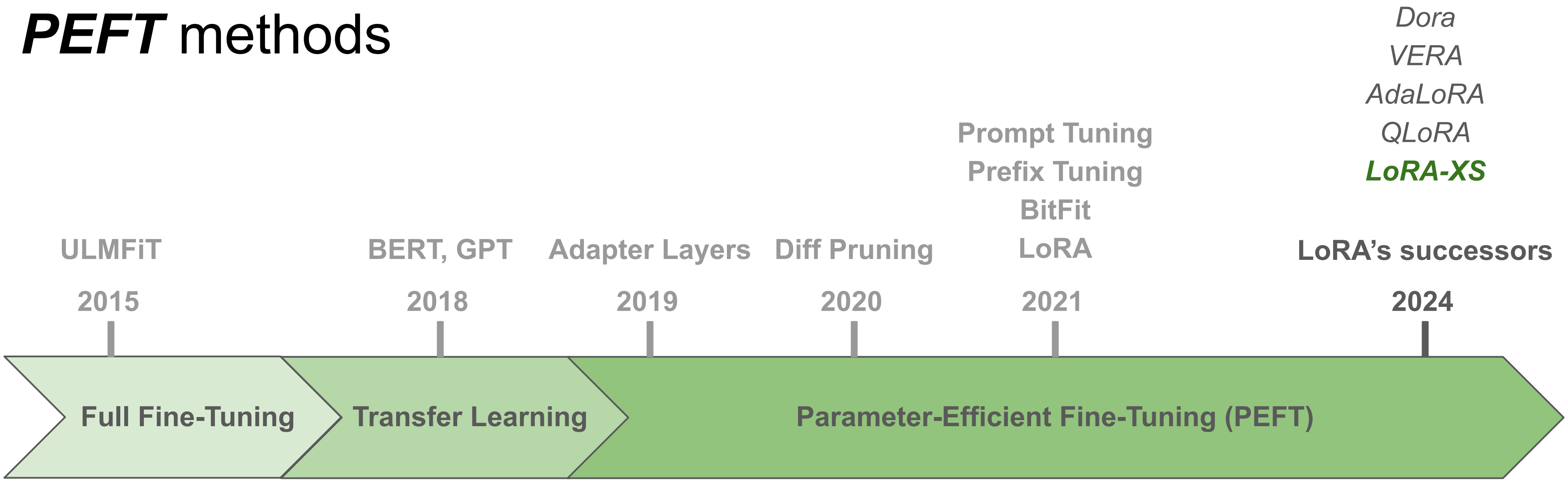


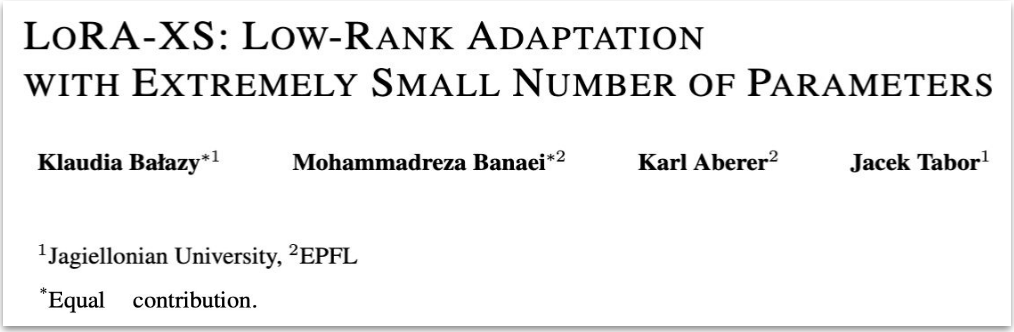
Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18],[19],[20],[21],[23]

Agenda

1. What is PEFT? Parameter-Efficient Fine-Tuning
2. Why do we need it?
3. What are the PEFT approaches?
4. Our PEFT proposal: **LoRA-XS**

PEFT methods





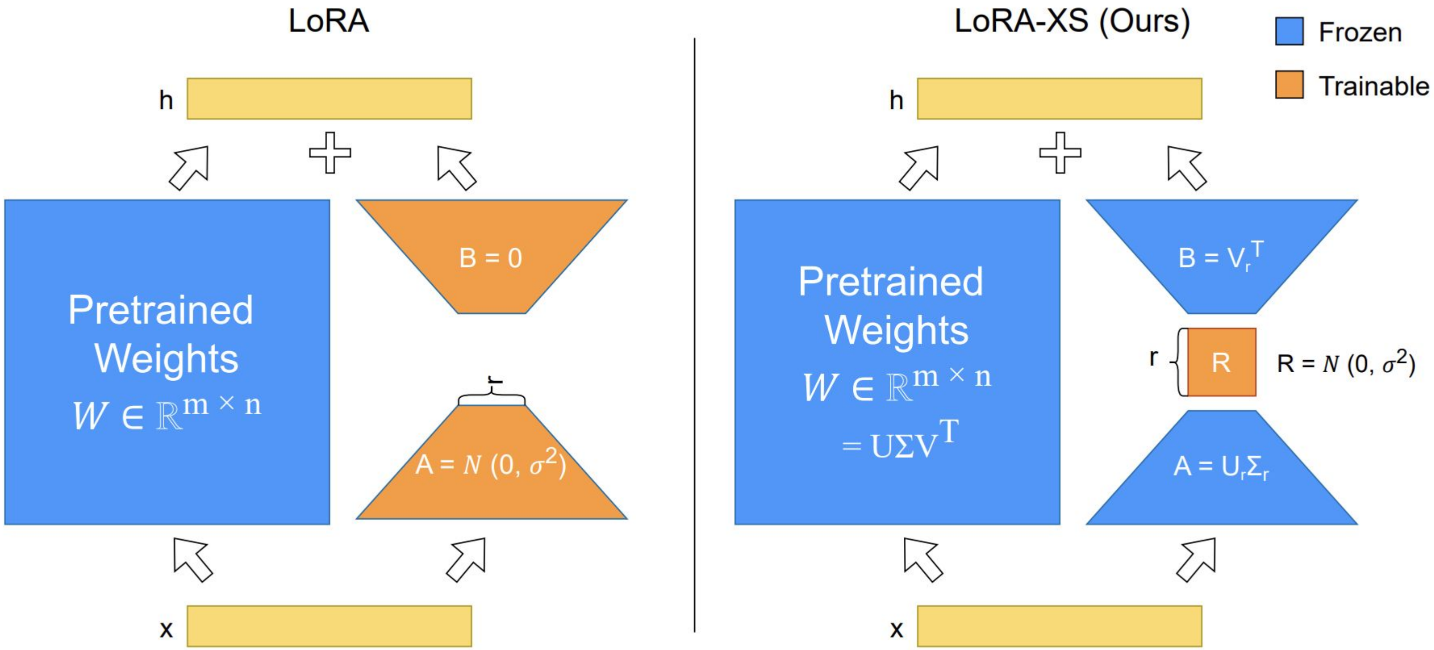
LoRA-XS: Low-Rank Adaptation with Extremely Small Number of Parameters

Klaudia Bałazy[*1] **Mohammadreza Banaei**[*2] **Karl Aberer**[2] **Jacek Tabor**[1]

[1]Jagiellonian University, [2]EPFL

[*]Equal contribution.

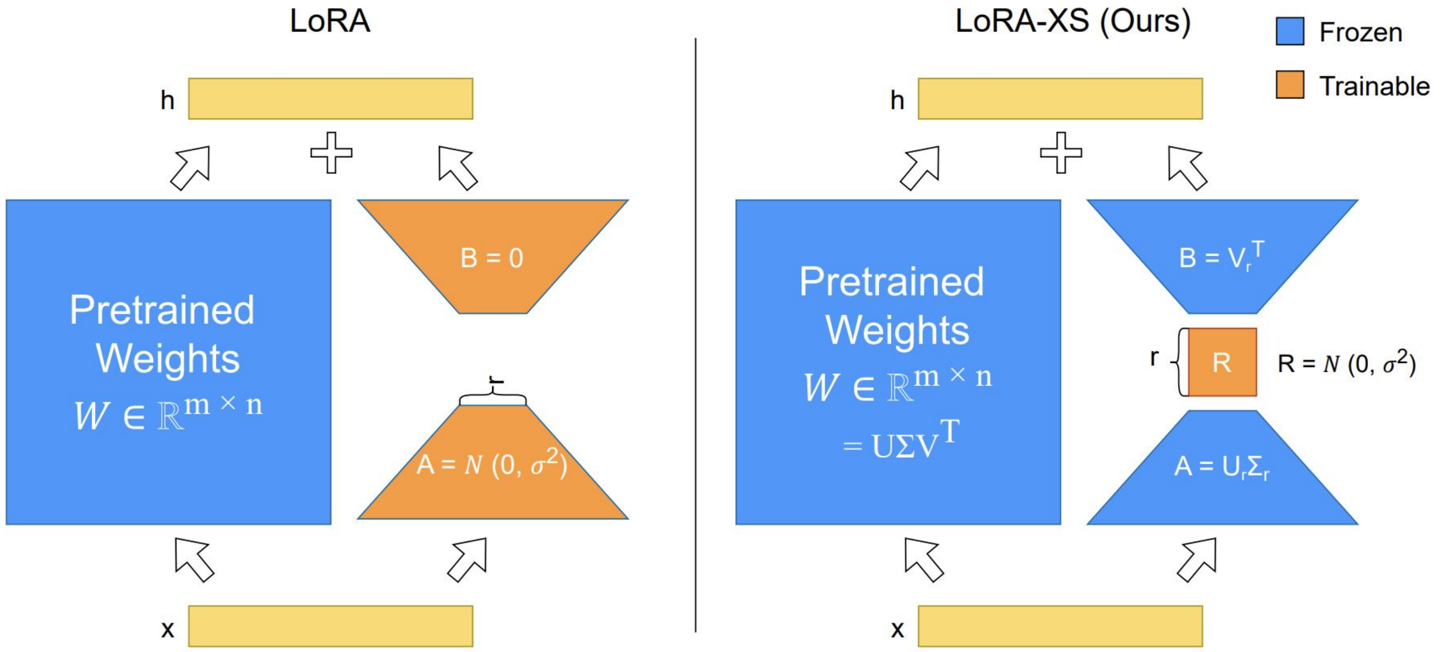
Sources: [3],[4],[10],[11],[12],[13],[14],[16],[17],[18],[19],[20],[21],[23],[24]



Traditional **LoRA** forward path for $x \in \mathbb{R}^n$:

$h = xW + x\Delta W = xW + xAB$, where:

$W \in \mathbb{R}^{m \times n}$, $\Delta W \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$ and $r << \min(m,n)$.



LoRA-XS forward path:

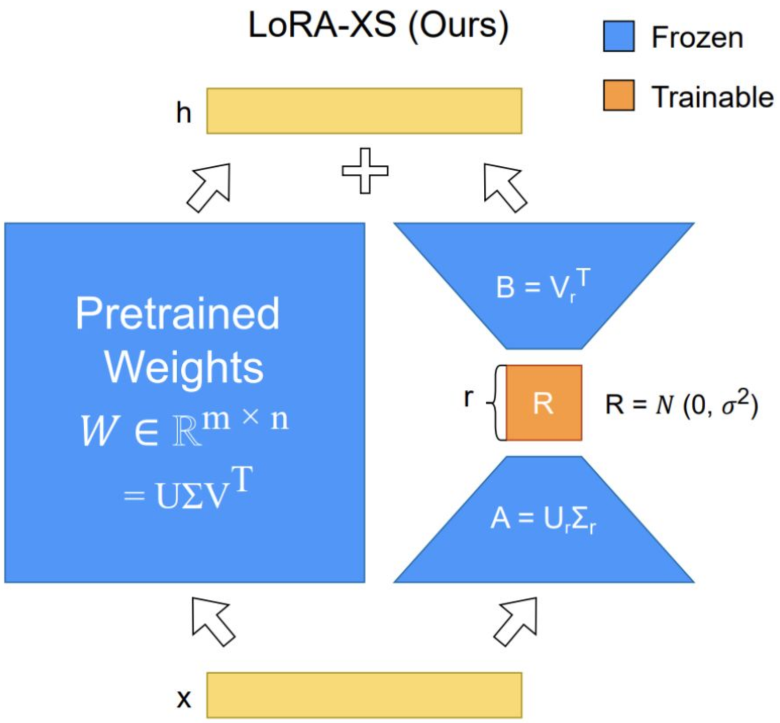
$h = xW + x\Delta W = xW + xARB$, where:

$W \in \mathbb{R}^{m \times n}$, $\Delta W \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{r \times r}$, $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$ and $r \ll \min(m,n)$.

$SVD(W) = U\Sigma V^T$ and $A = U_r\Sigma_r$ and $B = V_r^T$.

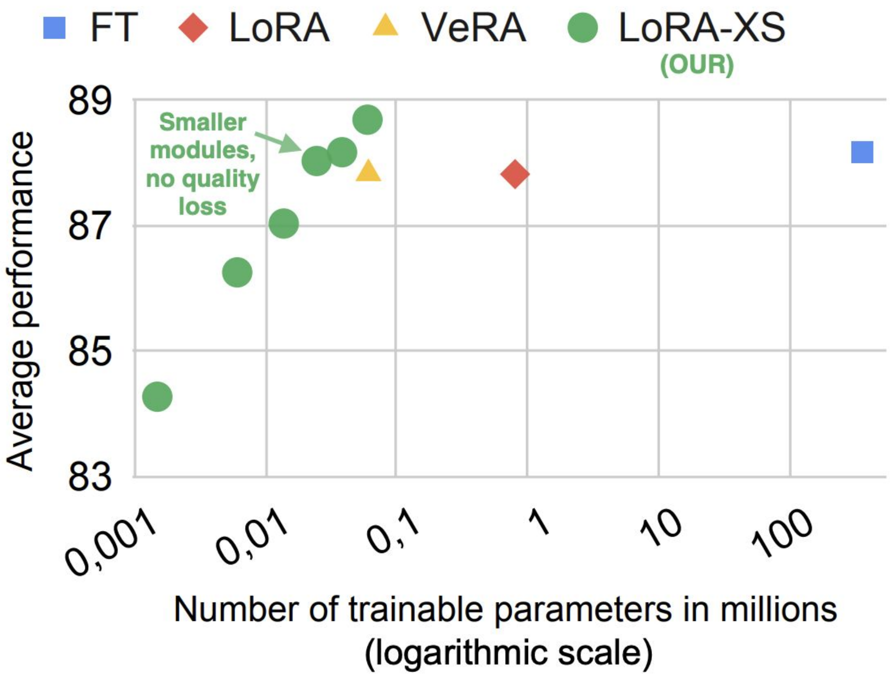
LoRA-XS

1. **Fewer trainable parameters** + decoupling from the model dimension



LoRA-XS

1. Fewer trainable parameters + decoupling from the model dimension
2. **Strong results** on GLUE, GSM8k, MATH and eight commonsense reasoning benchmarks for RoBERTa-large, LLaMA2-7B, LLaMA3-8B, Mistral 7B and Gemma 7B.



Average performance of RoBERTa-large on a subset of GLUE tasks as a function of the number of trainable parameters (in millions) for different adaptation methods: Full Fine-Tuning (FT), LoRA, VERA, and LoRA-XS.

LoRA-XS insights

1. Fewer trainable parameters + decoupling from the model dimension
2. Strong results on GLUE, GSM8k, MATH and eight commonsense reasoning benchmarks for RoBERTa-large, LLaMA2-7B, LLaMA3-8B, Mistral 7B and Gemma 7B.
3. Theoretical derivation backed up by experimental results: **SVD-initialized LoRA-XS modules enhance convergence and performance,** especially when tasks align with pre-training objectives.

Init. Type	SST-2	COLA	MRPC	QNLI
random	94.72	58.53	85.78	88.80
SVD of random	**94.84**	55.27	84.31	88.34
SVD of W	94.72	**60.11**	**87.50**	**90.94**

Performance of LoRA-XS with various initialization schemes. We present the best median scores across different learning rates, averaged over 5 seeds for rank 4. We report Matthew's correlation for CoLA and accuracy for the other tasks.

LoRA-XS insights

1. Fewer trainable parameters + decoupling from the model dimension
2. Strong results on GLUE, GSM8k, MATH and eight commonsense reasoning benchmarks for RoBERTa-large, LLaMA2-7B, LLaMA3-8B, Mistral 7B and Gemma 7B.
3. Theoretical derivation backed up by experimental results: SVD-initialized LoRA-XS modules enhance convergence and performance, especially when tasks align with pre-training objectives.
4. **Top singular vectors** in transformer weights **retain the most task-relevant knowledge.**

LoRA-XS insights

1. Fewer trainable parameters + decoupling from the model dimension
2. Strong results on GLUE, GSM8k, MATH and eight commonsense reasoning benchmarks for RoBERTa-large, LLaMA2-7B, LLaMA3-8B, Mistral 7B and Gemma 7B.
3. Theoretical derivation backed up by experimental results: SVD-initialized LoRA-XS modules enhance convergence and performance, especially when tasks align with pre-training objectives.
4. Top singular vectors in transformer weights retain the most task-relevant knowledge.
5. **Retaining the top singular vectors** consistently yields **better performance for LoRA-XS** across various tasks.

LoRA-XS insights

1. Fewer trainable parameters + decoupling from the model dimension
2. Strong results on GLUE, GSM8k, MATH and eight commonsense reasoning benchmarks for RoBERTa-large, LLaMA2-7B, LLaMA3-8B, Mistral 7B and Gemma 7B.
3. Theoretical derivation backed up by experimental results: SVD-initialized LoRA-XS modules enhance convergence and performance, especially when tasks align with pre-training objectives.
4. Top singular vectors in transformer weights retain the most task-relevant knowledge.
5. Retaining the top singular vectors consistently yields better performance for LoRA-XS across various tasks.
6. The results indicate improved performance when **top singular values Σ are included** in most cases.

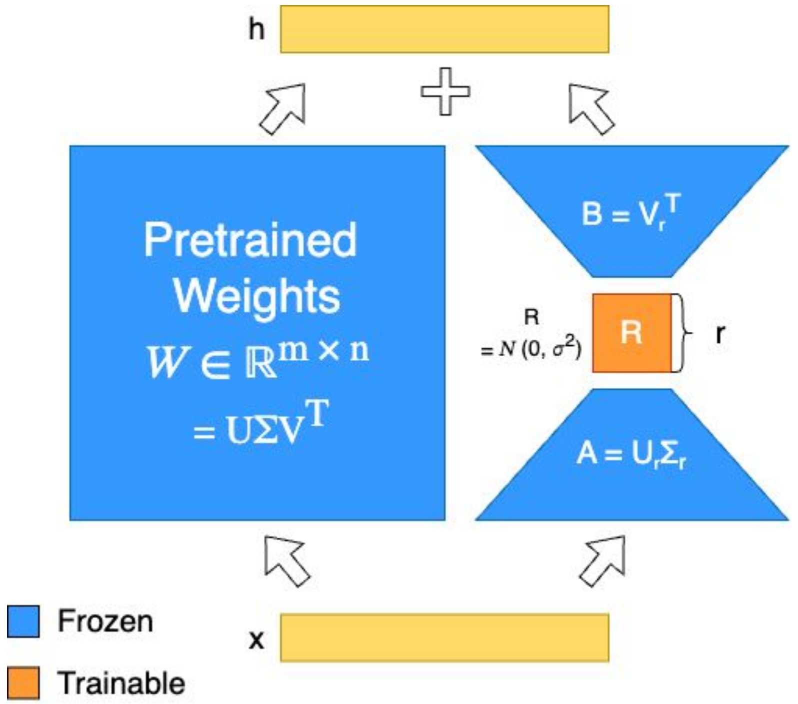
$$h = xW + x\Delta W = xW + xARB$$

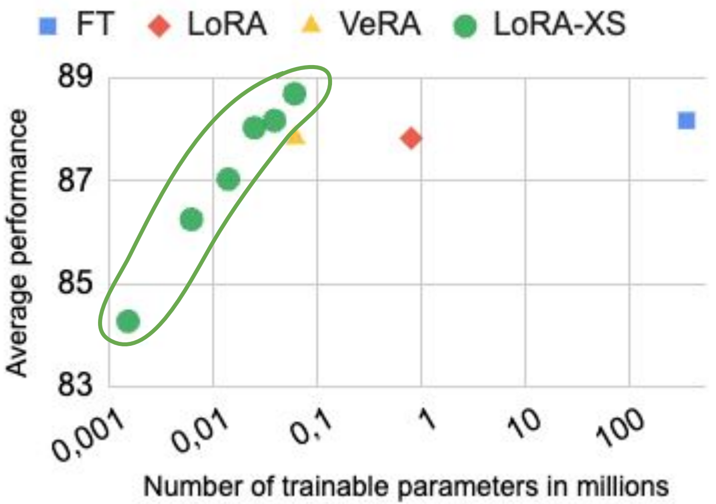
$$SVD(W) = U\Sigma V^T$$

$$\boldsymbol{A=U_r\Sigma_r} \text{ and } B=V_r^T \text{ vs } \boldsymbol{A=U_r} \text{ and } B=V_r^T$$

When to use LoRA-XS?

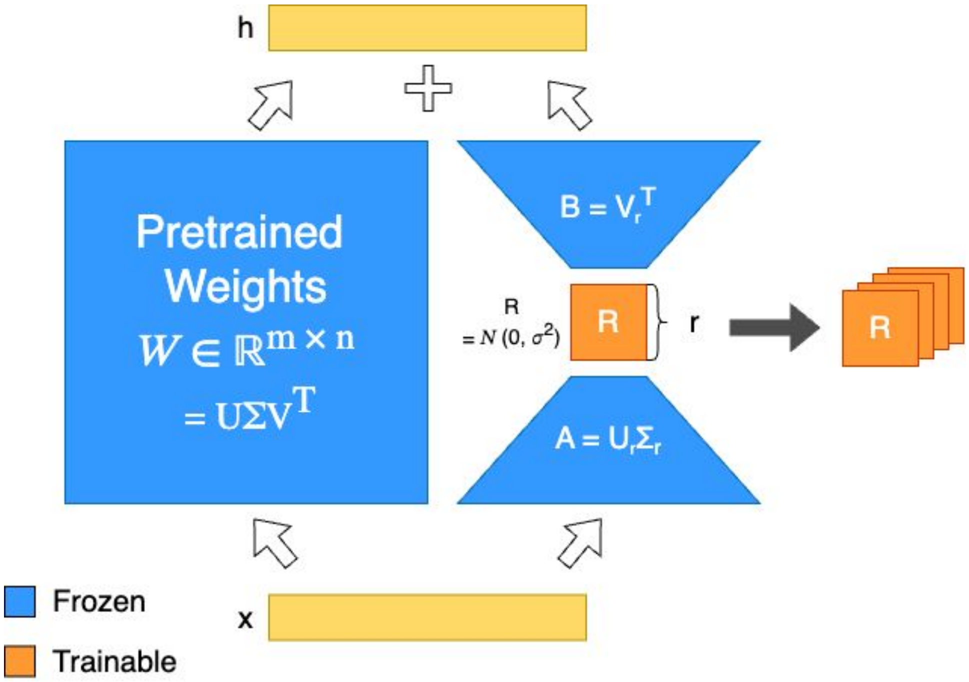
✅ Extreme memory constraints (decoupling from the model dimension)





When to use LoRA-XS?

✅ Need to store a huge number of personalized models



Agenda

1. What is PEFT? Parameter-Efficient Fine-Tuning
2. Why do we need it?
3. What are the PEFT approaches?
4. Our PEFT proposal: LoRA-XS

Thank you! 😊

Bibliography

[1] Peters, Matthew E. "Deep contextualized word representations." arXiv preprint arXiv:1802.05365 (2018).
[2] Fawad Ali (2023, April 11). GPT-1 to GPT-4: Each of OpenAI's GPT Models Explained and Compared. MakeUseOf. https://www.makeuseof.com/gpt-models-explained-and-compared/ (Access: 22 Oct 2024)
[3] Radford, Alec. "Improving language understanding by generative pre-training." (2018).
[4] Kenton, Jacob Devlin Ming-Wei Chang, and Lee Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." Proceedings of naacL-HLT. Vol. 1. 2019.
[5] Radford, Alec, et al. "Language models are unsupervised multitask learners." OpenAI blog 1.8 (2019): 9.
[6] Raffel, Colin, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." Journal of machine learning research 21.140 (2020): 1-67.
[7] Brown, Tom B. "Language models are few-shot learners." arXiv preprint arXiv:2005.14165 (2020).
[8] Smith, Shaden, et al. "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model." arXiv preprint arXiv:2201.11990 (2022).
[9] Driess, Danny, et al. "Palm-e: An embodied multimodal language model." arXiv preprint arXiv:2303.03378 (2023).
[10] Howard, Jeremy, and Sebastian Ruder. "Universal language model fine-tuning for text classification." arXiv preprint arXiv:1801.06146 (2018).
[11] Houlsby, Neil, et al. "Parameter-efficient transfer learning for NLP." International conference on machine learning. PMLR, 2019.
[12] Guo, Demi, Alexander M. Rush, and Yoon Kim. "Parameter-efficient transfer learning with diff pruning." arXiv preprint arXiv:2012.07463 (2020).
[13] Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." arXiv preprint arXiv:2104.08691 (2021).
[14] Li, Xiang Lisa, and Percy Liang. "Prefix-tuning: Optimizing continuous prompts for generation." arXiv preprint arXiv:2101.00190 (2021).
[15] Raschka, S. (2023, April 30). Understanding Parameter-Efficient LLM Finetuning: Prompt Tuning and Prefix Tuning. The Machine Learning Magazine. https://magazine.sebastianraschka.com/p/understanding-parameter-efficient (Access: 22 Oct 2024).

Bibliography

[16] Zaken, Elad Ben, Shauli Ravfogel, and Yoav Goldberg. "Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models." arXiv preprint arXiv:2106.10199 (2021).
[17] Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." arXiv preprint arXiv:2106.09685 (2021).
[18] HuggingFace, PEFT, https://huggingface.co/docs/peft/
[19] Kopiczko, Dawid J., Tijmen Blankevoort, and Yuki M. Asano. "Vera: Vector-based random matrix adaptation." arXiv preprint arXiv:2310.11454 (2023).
[20] Liu, Shih-Yang, et al. "Dora: Weight-decomposed low-rank adaptation." arXiv preprint arXiv:2402.09353 (2024).
[21] Zhang, Qingru, et al. "AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning." arXiv preprint arXiv:2303.10512 (2023).
[22] Dettmers, Tim, et al. "Qlora: Efficient finetuning of quantized llms." Advances in Neural Information Processing Systems 36 (2024).
[23] Wu, Zhengxuan, et al. "Reft: Representation finetuning for language models." arXiv preprint arXiv:2404.03592 (2024).
[24] Bałazy, Klaudia, et al. "LoRA-XS: Low-Rank Adaptation with Extremely Small Number of Parameters." arXiv preprint arXiv:2405.17604 (2024)
[25] https://github.com/hiyouga/LLaMA-Factory#hardware-requirement (Access: 22 Oct 2024)
[26] Zhao, Jiawei, et al. "Galore: Memory-efficient llm training by gradient low-rank projection." arXiv preprint arXiv:2403.03507 (2024).
[27] Luo, Qijun, Hengxu Yu, and Xiao Li. "BAdam: A Memory Efficient Full Parameter Training Method for Large Language Models." arXiv preprint arXiv:2404.02827 (2024).

Check out our other talks during ML in PL!

Friday:

Session 2 / Lecture Hall B / 10:35

Deep learning for effective analysis of high content screening
Adriana Borowa

Session 4 / Lecture Hall A / 14:30

Efficient fine-tuning of LLMs: exploring PEFT methods and LORA-XS insights
Klaudia Bałazy

Session 5 / Lecture Hall B / 14:30

Current trends in intrinsically interpretable Deep Learning
Dawid Rymarczyk

Neural rendering: the future of 3D modeling
Przemysław Spurek

JAGIELLONIAN UNIVERSITY IN KRAKÓW

group of machine learning research
gmum

Saturday:

Session 7 / Lecture Hall A / 12:00

AdaGlimpse: Active Visual Exploration with Arbitrary Glimpse Position and Scale
Adam Pardyl

Session 8 / Lecture Hall B / 12:00

Augmentation-aware Self-supervised Learning with Conditioned Projector
Marcin Przewięźlikowski

gmum.net