# Fine-tuning Image Transformers using Learnable Memory

Mark Sandler, Andrey Zhmoginov, Max Vladymyrov, Andrew Jackson

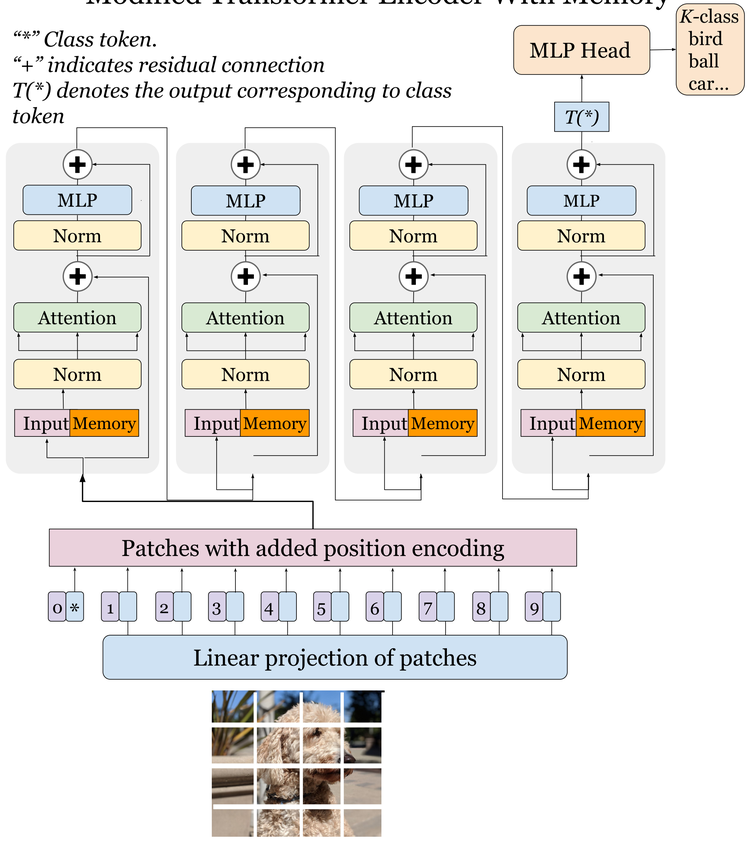
CVPR JUNE 19-24 2022 NEW ORLEANS • LOUISIANA

## Core idea

- Use learnable memory tokens passed alongside input tokens to **every layer** to fine-tune the transformer on new tasks, while keeping the rest of transformer fixed

### Modified Transformer Encoder With Memory

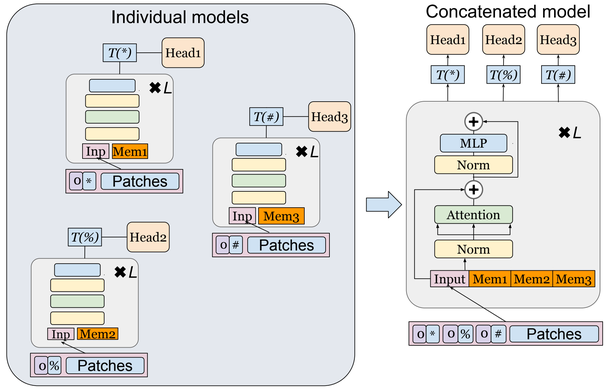
*"*" Class token.*
*"+" indicates residual connection*
*T(*) denotes the output corresponding to class token*




## Use standard self attention at each layer

- The input tokens can attend to memory and class token
- Class token attends to memory and input tokens
- Memory is "learned", but it is passive (doesn't attend to input tokens)

## Extension: Computation reuse in multi tasks




- Problem: Adding input tokens doesn't allow computation reuse
- Solution: Only new task token can attend to new memory, while input tokens only perform self-attention.
- Improves on MLP-head finetuning.

## Experimental Results

| | Full | Head Only | Head + Class | 1 cells | 5 cells | 10 cells | 20 cells |
|---|---|---|---|---|---|---|---|
| Sun-297 | 76.9 | 75.9 | 76.4 | **77.0** | 76.8 | 76.9 | 76.9 |
| iNaturalist | 54.0 | 46.3 | 48.7 | 49.4 | **50.1** | 50.0 | 50.0 |
| Cifar-100 | 91.8 | 85.8 | 88.6 | 90.9 | 91.0 | **91.1** | 90.9 |
| Places-365 | 55.9 | 50.9 | 52.4 | 53.3 | 53.8 | 53.9 | **54.1** |

Table 2. Accuracy for different datasets using the optimal learning rate for each fine-tuning regime.

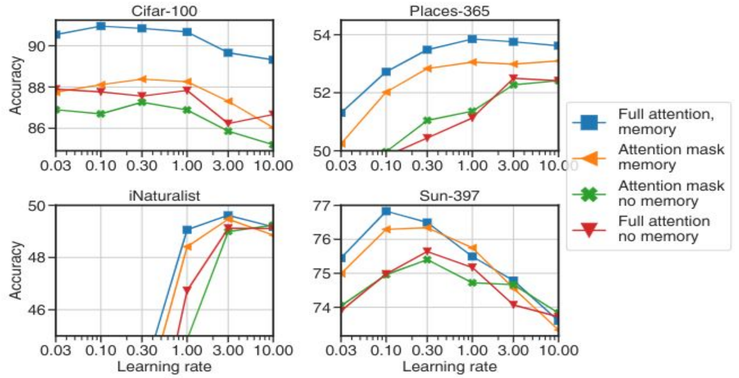
| | Full | Head only | Class | Mem (5) |
|---|---|---|---|---|
| Params | 80M | $768 \times k$ | 768 | $768 \times 12 \times 5$ |
| Flops | ≈1G | 0 | 0 | ≈ 25M |

Table 5. Incremental cost on number of parameters and FLOPS for ViT-32-B for each fine-tuning regime. Class and head fine-tuning do not incur any extra computation cost as the architecture stays unchanged. Here $k$ denotes number of classes, and we assume 5 memory tokens per layer.

| | Accuracy | Params (ViT-B/16) |
|---|---|---|
| Memory | 50.1 | 768 * 12 * m ≈ 46K |
| Adapters [17] | 49.9 | 3078 * 5 * 4 * m ≈ 737K |
| Combined | 50.3 | 46K + 737K ≈ 783K |

Table 4. Performance for i-Naturalist for adapters [17] with bottleneck of size 5, vs learnable memory with 5 cells. (ours). Parameter counts exclude the size of the head which is the same for all methods

## Memory with computation reuse vs full attention




## Summary/Conclusion

- Learnable memory provides parameter efficient fine-tuning method that also allows computation reuse. More parameter efficient/Accurate than competing methods (e.g. adapters, and other types of memory)