AI for the open-world:

The Learning Principles

Speaker: Jianyu Zhang
Date: June, 2025

- Jianyu Zhang, **AI for the Open-World: the Learning Principles**. arXiv preprint 2025. (Jianyu's PhD thesis. advisors: Léon Bottou & Yann Lecun).
  - *Jianyu Zhang, Léon Bottou.* ***Learning useful representations for shifting tasks and distributions.*** *ICML 2023.*
  - *Jianyu Zhang, Niklas Nolte, Ranajoy Sadhukhan, Beidi Chen, and Léon Bottou.* ***Memory mosaics.*** *ICLR 2024.*
  - *Jianyu Zhang, Léon Bottou.* ***Memory Mosaics at scale.*** *Under review 2025.*

What AI did we achieve?

- AI for artificial environments. e.g., *Chess*, *Go*, *Atari games*.
- AI for specific real-world tasks. e.g., handwritten *recognition*, *face recognition*.

What AI did we achieve?

AI for the closed world

- Competence on **specific tasks**
  - One can collect **a lot of data**
  - One can define a precise **criteria of success.** e.g. game scores, precision
    - This criteria not only reveals whether a machine has achieved a goal, but also reveals how the machine falls short of the goal.
    - Human designers can fix the problems one after the other until the machine is deemed good enough for the task.
    - At the end, instead of proving that our machine is intelligent, we often find satisfaction in proving that we are intelligent.[1]

solve a narrow range of tasks

require a lot of examples

rely on priori knowledge of human designers

This is **not** what we want!

1. Leon Bottou, https://leon.bottou.org/feuilleton/turing

What AI do we need?

AI for the open-world

- Solves any task that a human could possibly undertake
  - With fewer examples,
    - We cannot wait too long for a AI system to learn a new task.
  - And less priori knowledge from designers.
    - We cannot assign one human designer to each possible task.

AI for the Open-world

AI for open-world requires a machine to learn on a **wide range** of new tasks/domains (versatility) quickly using **fewer examples** and **less task-specific priori knowledge** (from human designers).

Learning Principles to build AI for open-world

AI for the open-world
wide range of tasks
few examples
less priori knowledges

Rich Features,
a richer set of features beyond the need of i.i.d. generalization, helps the learning of a **broader** range of unseen tasks;

disentangled representation,
driven by a cheap yet reliable pressure predictive disentanglement, **reduces** the number of examples required on unseen tasks

inference-time learning,
a memory-based method at inference-time, leveraging rich features and disentangled representation, reduces the relying of designers' priori knowledge.

Can we translate the success in Closed-world to Open-world?

AI for the closed-world	AI for the open-world
narrow range of tasks	wide range of tasks
Lots of examples	few examples
Priori knowledges from designers	less priori knowledges

- 💡**Train on everything!** *I.e. The mainstream belief of foundational models.*
  - **Computationally infeasible:**
    - Combining two pieces of knowledge results in a new knowledge, leading to an exponentially large number of possible combinations
  - **Practically implausible:**
    - Pieces of knowledge change over time.

No! We need other learning principles to build AI for the open-world!

This talk explores such **learning principles** and corresponding **techniques** to build AI for the **open-world.**

*Note: this is **not** a talk about language models or vision-models or applications of deep learning, even though it contains many large-scale experiments of such kind.*

1. rich features

*a richer set of features beyond the need of i.i.d. generalization, helps the learning of a **broader** range of unseen tasks;*

Rich Features

- Benefits of rich features in learning unseen tasks.
- Can we construct rich features by ERM training on big data & large model *(i.e., foundational model)?*
- Approaches to construct rich features

Warm-up case

- Background
  - L2 weight decay in deep neural network stochastic optimization has a sparsity bias (sparse features) [Blanc et al. 2020]
- Transfer learning experiment
  - Pretrain resnet on cifar10 (1st row), then linear-probe on cifar100 (2nd row).
  - As expected, L2 weight decay helps IID performance on Cifar10,
  - but hurt OOD performance on Cifar100 (linear-probe).

L2 weight decay	0	$5e-4$
CIFAR10	91.41±0.81	**94.89±0.23**
CIFAR10→CIFAR100	**49.68±0.72**	29.17±0.50

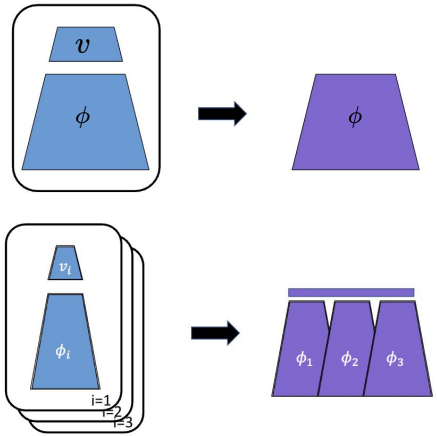
[zhang et al. 2023]

Rich features help learning unseen tasks.

Blanc, Guy, et al. Implicit regularization for deep neural networks driven by an Ornstein-Uhlenbeck-like process. PMLR, 2020.
Zhang, J., & Bottou, L. Learning useful representations for shifting tasks and distributions. ICML 2023.

Can we find rich features by scaling-up ERM training?

- Let's compare two representations *(same number of parameter)*:
  - 1) Train a **large model** with supervised or self-supervised objective. Then take the penultimate layer as representation.
  - 2) Train **multiple small models** (on the same data) with different **random seed.**[1] Then concatenate the penultimate layer representations.
- We know rich features help learning unseen tasks.
- 🤔 Which representation is better for learning unseen tasks (via linear-probe or fine-tune)?

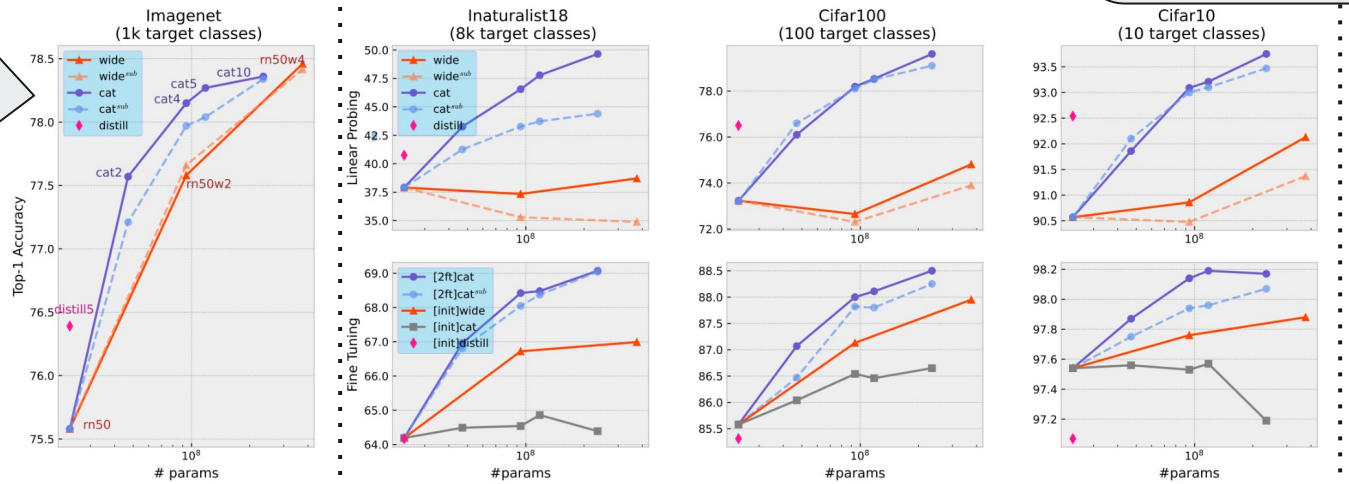


1. Random seed controls network initialization and examples order in SGD optimization.

- Supervised transfer learning case.
  - Pretraining ResNets on Imagenet.
  - linear-probing / find-tuning on broad unseen tasks
  - Orange solid line: 1st representation, big models
  - Blue solid line: 2nd representation, concatenation of small models

For IID, big model (resnet50w4) outperforms the concatenation of small models. *(nothing to supervise)*

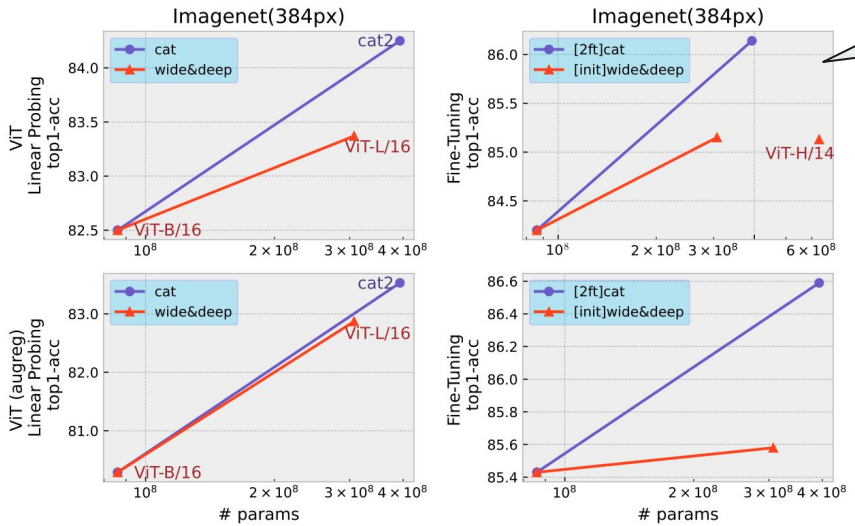
For the learning on unseen tasks, the concatenation of small models is far better!



(Zhang, et. al, 2023)

Zhang, J., & Bottou, L. Learning useful representations for shifting tasks and distributions. ICML 2023.

- How about using Transformer architecture?



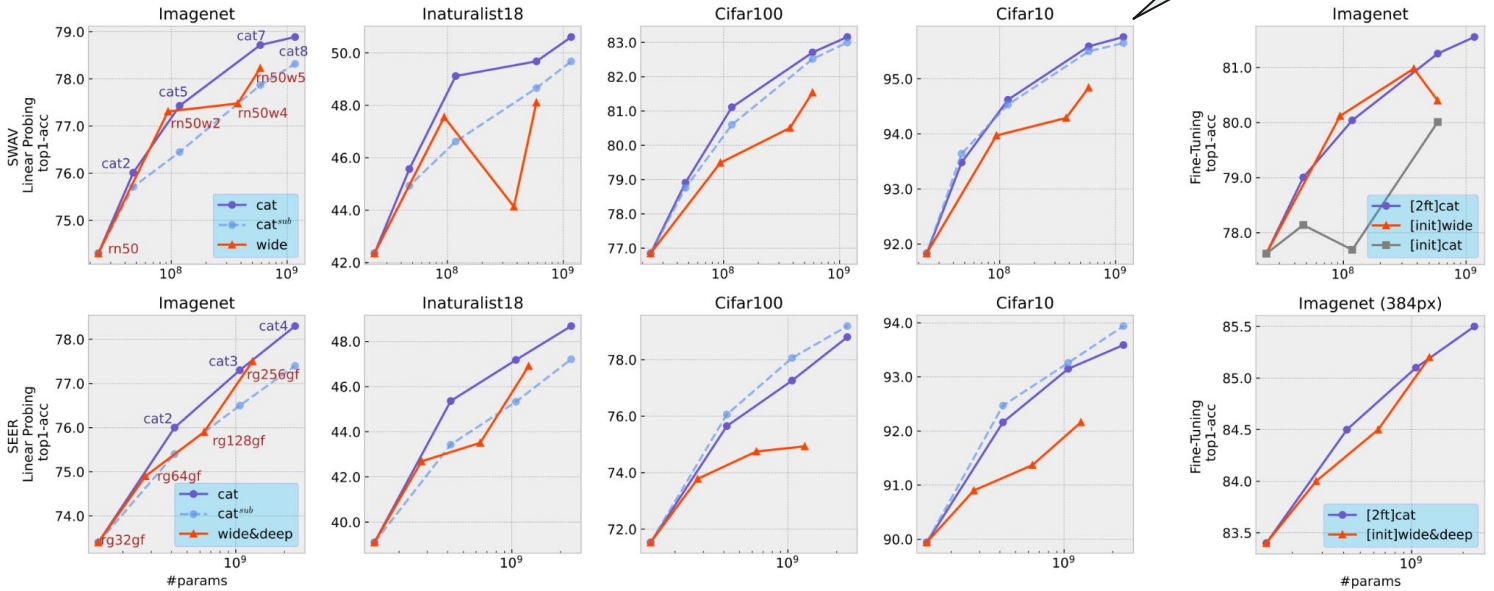
Same result.

Figure 3. Supervised transfer learning from IMAGENET21K to IMAGENET on vision transformers. (Zhang, et. al, 2023)

Zhang, J., & Bottou, L. Learning useful representations for shifting tasks and distributions. ICML 2023.

- How about using self-supervised learning, large data, and large model?
  - SWAV is trained on ImageNet.
  - SEER is trained on 1B images with 10B dense parameters.

Same result.



(Zhang, et. al, 2023)

Zhang, J., & Bottou, L. Learning useful representations for shifting tasks and distributions. ICML 2023.

Can we find rich features by scaling-up ERM training?

- 😢 NO.

Once the optimization process discovered a group of features that are helpful to the IID training, there is no motivation to find other features that not are ***incremental helpful*** *to training, even though these features could be substantially help to other distributions (OOD).*

This confliction between rich features and optimization motivates us to move from "**optimization**" to "**memory**", and create Memory Mosaics (to be introduced later).

Approaches to construct rich features

- Adversarial discovery
  - Bonsai [Zhang, et al., 2022]
- Randomness
  - feature ensemble [Zhang, et al., 2023]
  - very-large dropout (in fine-tuning) [Zhang, et al., 2024]
  - weight averaging (in fine-tuning) [Ramé, et al., 2023]

Zhang, J., Lopez-Paz, D., & Bottou, L. (2022, June). Rich feature construction for the optimization-generalization dilemma. ICML
Zhang, J., & Bottou, L. (2023, July). Learning useful representations for shifting tasks and distributions. ICML
Zhang, J., & Bottou, L. (2024). Fine-tuning with Very Large Dropout. *arXiv preprint arXiv:2403.00946*.
Ramé, et al. (2023). Model ratatouille: Recycling diverse models for out-of-distribution generalization. ICML

Rich Features

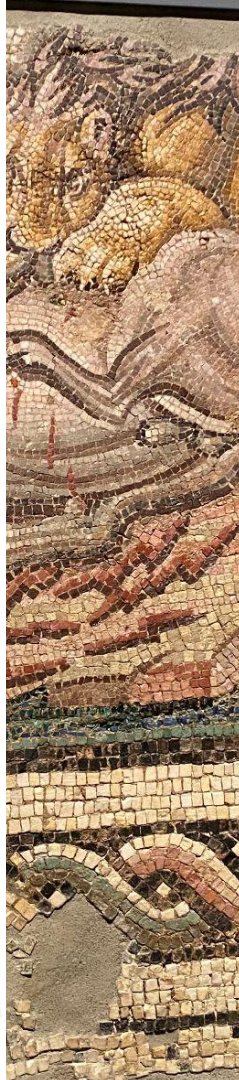
- Benefits of rich features
  - Help learning broader range of unseen tasks
- Can we find rich features by scaling-up ERM training?
  - **No!**
- Approaches to find rich features
  - Adversarial discovery — Bonsai
  - Randomness — feature ensemble, weight averaging, very-large dropout

2. Disentangled representation

driven by a cheap yet reliable pressure predictive disentanglement,
***reduces** the number of examples required on unseen tasks*

Disentangled representation

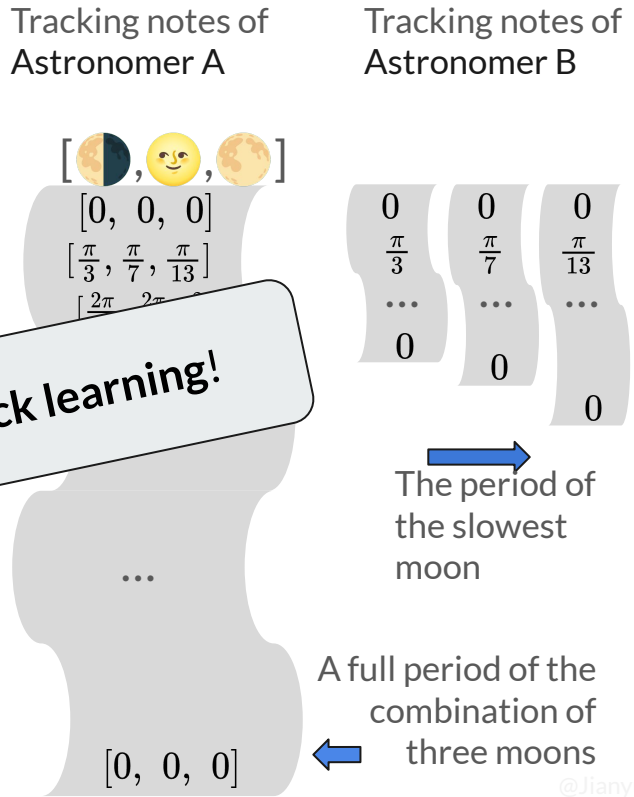
- **Benefits** of disentanglement
- The **pressure** that motivates a model to learn disentanglement
- An **architecture** to learn disentanglement in practice (memory mosaics)



Benefits of disentanglement

- **Example:** Tracking Three Moons
  - Astronomer **A** tracks the **joint position** of three moons
    - **Least Common Multiple** (periods of mo
  - Astro
    each r **eparately**
    - **Max** (periods of moons)

Disentanglement leads to **quick learning!**



The pressure that motivates a model to learn disentanglement

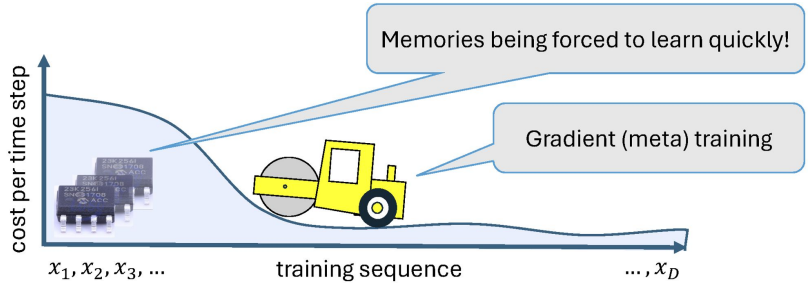
- **Statistical view** [Roth 2022]: disentanglement is defined on "independence".
  - Lack robustness with respect to changing data distributions.
- **Causal view** [Bengio 2013, Bengio 2019]: disentanglement is defined on active environments.
  - Cannot be tested without active experiments

Not reliable

expensive

Roth, Karsten, et al. "Disentanglement of correlated factors via hausdorff factorized support." *arXiv preprint arXiv:2210.07347* (2022).
Bengio, Yoshua, et al. "A meta-transfer objective for learning to disentangle causal mechanisms." *arXiv preprint arXiv:1901.10912* (2019).
Bengio, Yoshua. Deep learning of representations: Looking forward. In International conference on statistical language and speech processing, pages 1–37. Springer, 2013b.

Predictive disentanglement: a quick learning pressure



- View one natural sequence (e.g. an article) as one environment/distribution. (**cheap environments)**
- **Training objective** minimizes average loss at different sequence length. (the area under the curve).
- **The tail of sequence** is easy to model/predict anyway.
- Then training objective instead minimizes the **sequence length** needed to produce good predictions. (**The "quick learning" pressure)**
- **Disentanglement** occur when it is more efficiently to predict in isolation than together**.** (i.e. quick learning → disentanglement)

What we need to do is to encourage this "prediction in isolation" behavior.

Memory Mosaics:
architecture that encourages disentanglement

- Memory Mosaics is a network of multiple separated Associative Memories.
- Associative memory contains key-value pairs.
  - **Key** represents the recent past
  - **Value** represents the near future
  - Key-value pairs are treated as permutation invariant

Encourage "prediction in isolation"

Key is query.
Query is key.

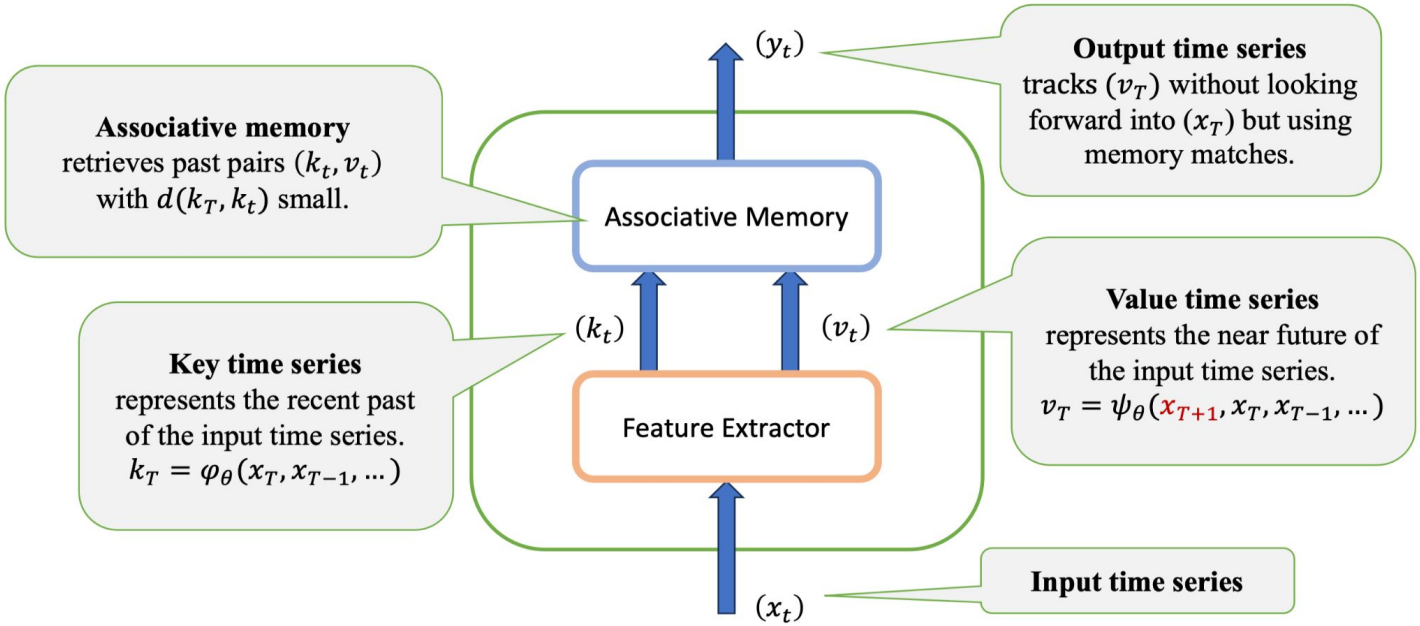
No position encoding.

Zhang, J., Nolte, N., Sadhukhan, R., Chen, B., & Bottou, L. (2024). Memory mosaics. *ICLR*.

Associative Memory *(no query!)*

- Store key-value pairs, retrieve values given a key. *(Invariant to permutation of stored pairs.)*
  - $$k \quad \mapsto \quad f\left(k; \{(k_1, v_1) \dots (k_n, v_n)\}\right)$$
- The retrieval function is a **conditional expectation**.
  - $$= \mathbb{E}(V \mid K = k)$$
- Estimate the key-value distribution by **Kernel Density Estimation** (e.g. Gaussian kernel smoothing)
  - $$= \sum_{i=1}^{n} \frac{1}{Z} e^{-\beta \|k - k_i\|^2} v_i \quad \text{with} \quad Z = \sum_{i=1}^{n} e^{-\beta \|k - k_i\|^2}$$
- Connect to attention by **fixing key vectors squared norm.**
  - $$= \sum_{i=1}^{n} \frac{e^{\beta\, k^\top k_i}}{\sum_{j=1}^{n} e^{\beta\, k^\top k_j}} v_i$$

Construct key and value (on sequence data)**?**



How to construct key and value?

Simple Case

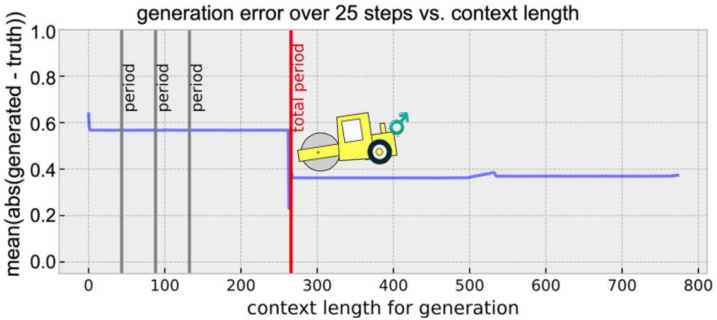
$$k_T = W_\varphi \, x_T$$
$$v_T = W_\psi \, x_{T+1}$$

expressive case

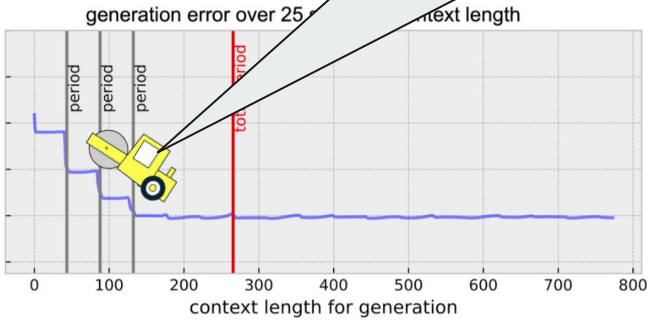
$$k_T = \alpha_\varphi \mathrm{Norm}\left(\bar{k}_T\right) \quad \text{with} \quad \overbrace{\bar{k}_T = \tilde{k}_T + \lambda_\varphi \bar{k}_{T-1}}^{\text{leaky average over t = T, T-1..., 1}} \qquad \tilde{k}_T = W_\varphi \, x_T$$
$$v_T = \alpha_\psi \mathrm{Norm}\left(\bar{v}_T\right) \quad \text{with} \quad \underbrace{\bar{v}_T = \tilde{v}_T + \lambda_\psi \tilde{v}_{T+1}}_{\text{convolution over t=T and T+1}} \qquad \tilde{v}_T = W_\psi \, x_T$$

Evaluate the “prediction in isolation” bias in Memory Mosaics

- Experiments on *Tracking Three Moons* problem
- **Left:** one big associative memory
- **Right:** many small associative memories



One big associative memory: Good predictions after the least common multiple of all moons periods (red vertical line).



Many small associative memories: Improving predictions after each moon periods (black vertical lines).

👷 I am good!
Multiple small associative memories make things easy.

Memory Mosaics on real-world tasks

- A fair comparison with transformer architectures.
- **C-mems (contextual memory)** are associative memory units of context.
- **P-mems (persistent memory)** are associative memory units of gradient-updatable key-value pairs. (fixed after training).
- NO position encoding
- key=query.

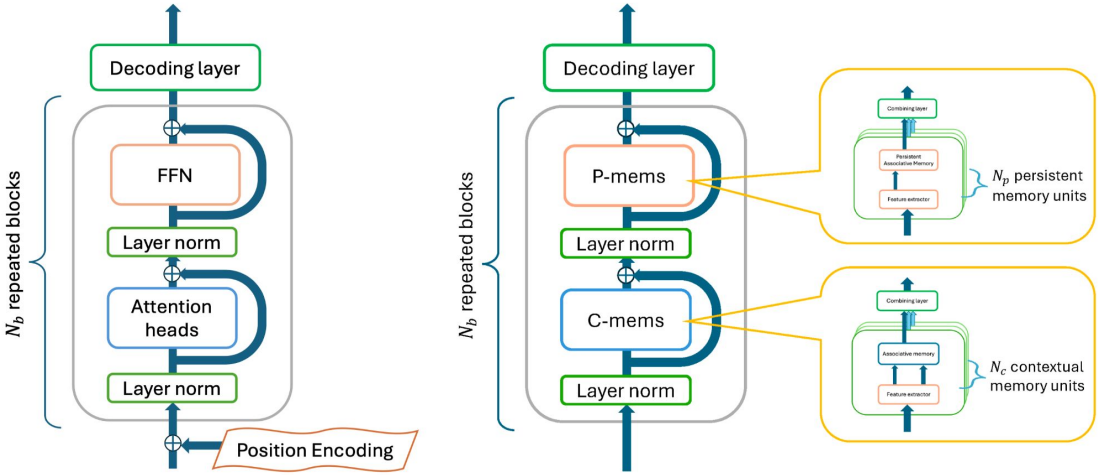
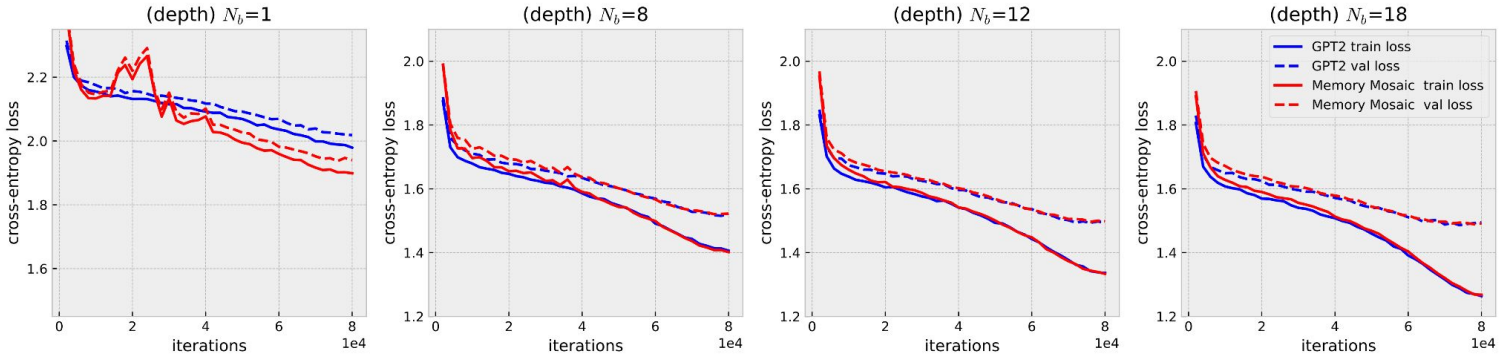


Figure 8: *Left*: Classic GPT2-small transformer. *Right*: GPT2-like Memory Mosaic

In-distribution performance

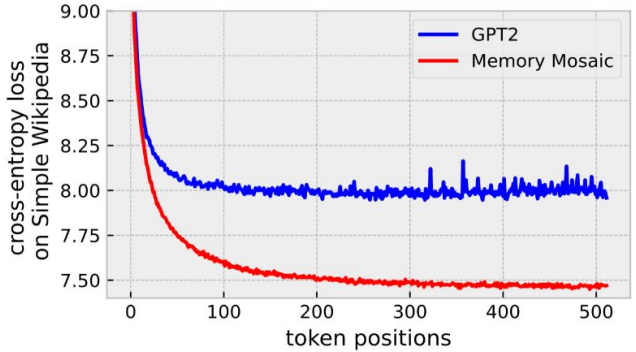
- Choose Transformer as a baseline, because it performs well on languages.
- Train on 3-4 years old tiny stories.
- Memory Mosaics and Transformer performs closely.



We focus on open-world and out-of-distribution performance.

Out-of-distribution performance

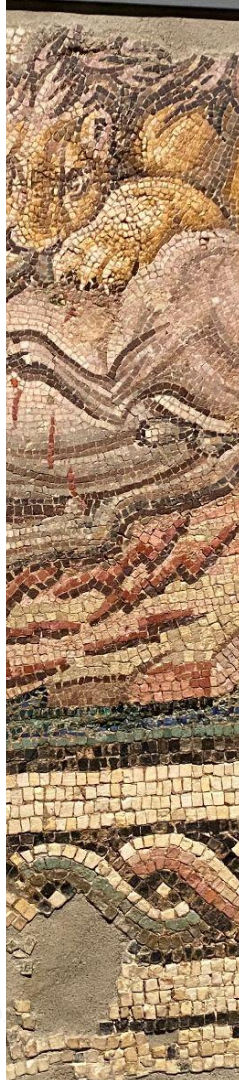
- Train on 3-4 years old tiny stories.
- Inference on Simple English Wikipedia (hard for 3-4 years old children)
- Similar to the *three-moons* problem, Memory Mosaics is faster and better in adjusting Wikipedia (a new data).



The plot shows the **per-token average loss** as a function of the generated token positions.

Disentangled representation

- Benefits of disentanglement
  - reducing the need of examples
- The pressure of learning disentanglement
  - *Predictive Disentanglement – a cheap yet reliable pressure.*
- An architecture to learn disentanglement in practice
  - *Memory Mosaics*



3. Inference-time learning

a memory-based method at inference-time, leveraging rich features and disentangled representation, reduces the relying of designers' priori knowledge.

Inference-time learning

- Why study inference-time learning?
- Build model to perform inference-time learning
- Gap between: “training on everything” & inference-time learning approaches.

Why study inference-time learning?

- Given rich & disentangled features, how to learn new tasks (*with few data*)?
  - 💡 Transfer learning (fine-tuning): data preprocessing, choose architectures, tune hyper-parameters, …
- What if I have another new task to learn?
  - Try the above process again.

The learning process on new tasks involves a lot of **priori knowledge from human designers**.
This is of course expensive. More importantly, it doesn't reveal the Intelligence of Machine, but the Intelligence of us.

Inference-time learning aims at reducing the need of priori knowledge from human designers while learning new tasks.

model-based vs memory-based approach

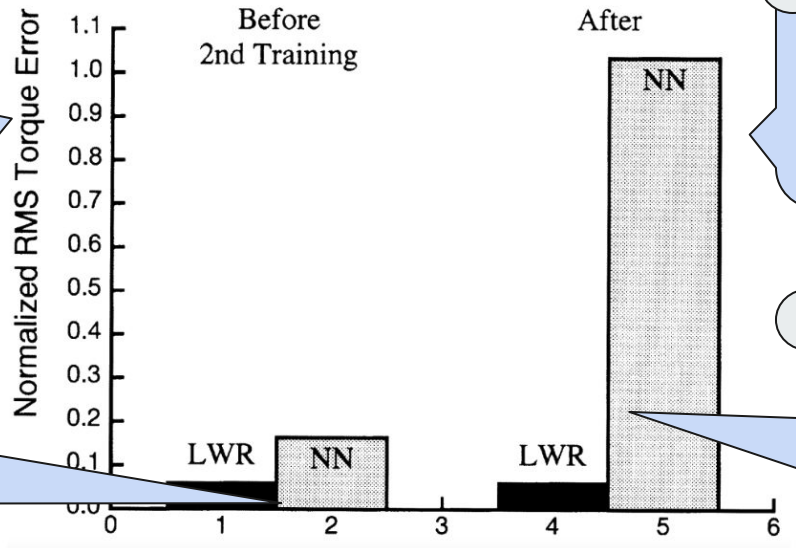
- To learn new tasks one after other with less priori knowledge and fewer examples, a model needs to **shape hypothesis space** and avoid **negative interference** (catastrophic forgetting).
- Shaping hypothesis space
  - Model-based: preprocessing, architectures, learning mechanisms *(e.g. regularization)* 😵
  - Memory-based: one smoothing parameter *(e.g. bandwidth in kernel smoothing)* 😄
- Negative interference
  - Model-based: hard to avoid.
  - Memory-based method were initially motivated to solve the negative interference problem. [Atkeson et al. 1997]

Christopher G Atkeson, Andrew W Moore, and Stefan Schaal. Locally weighted learning. Lazy learning, pages 11–73, 1997.

- Negative interference in memory-based and model-based methods.

1 ***L**ocal **W**eight **R**egression* (LWR) and ***N**eural **N**etwork* *(NN)* are used to predict the torques of two jointed arm dynamics.

2 Train and evaluate on the same distribution **A**: Both models work well!



[Atkeson et al. 1997]

3 Then both model are trained on ten attempts to make a **particular desired movement**. (distribution **B**)

4 After that, evaluate both models on distribution **A**. **NN fail, LWR success.**

Christopher G Atkeson, Andrew W Moore, and Stefan Schaal. Locally weighted learning. Lazy learning, pages 11–73, 1997.

Build model to perform inference-time learning

- We need **memory-based** methods!
- Potential problem of memory-based method: Curse of dimensionality
- Rich feature and disentangled representation could reduce this difficulty[1]
  - Rich feature: reduce the need for feature learning in inference time
  - Disentangled representation: organize each feature nicely in a low-dimensional space

Model for inference-time learning: memory-based methods at inference-time
\+ rich features and disentangled representations obtained from pre-training stage.

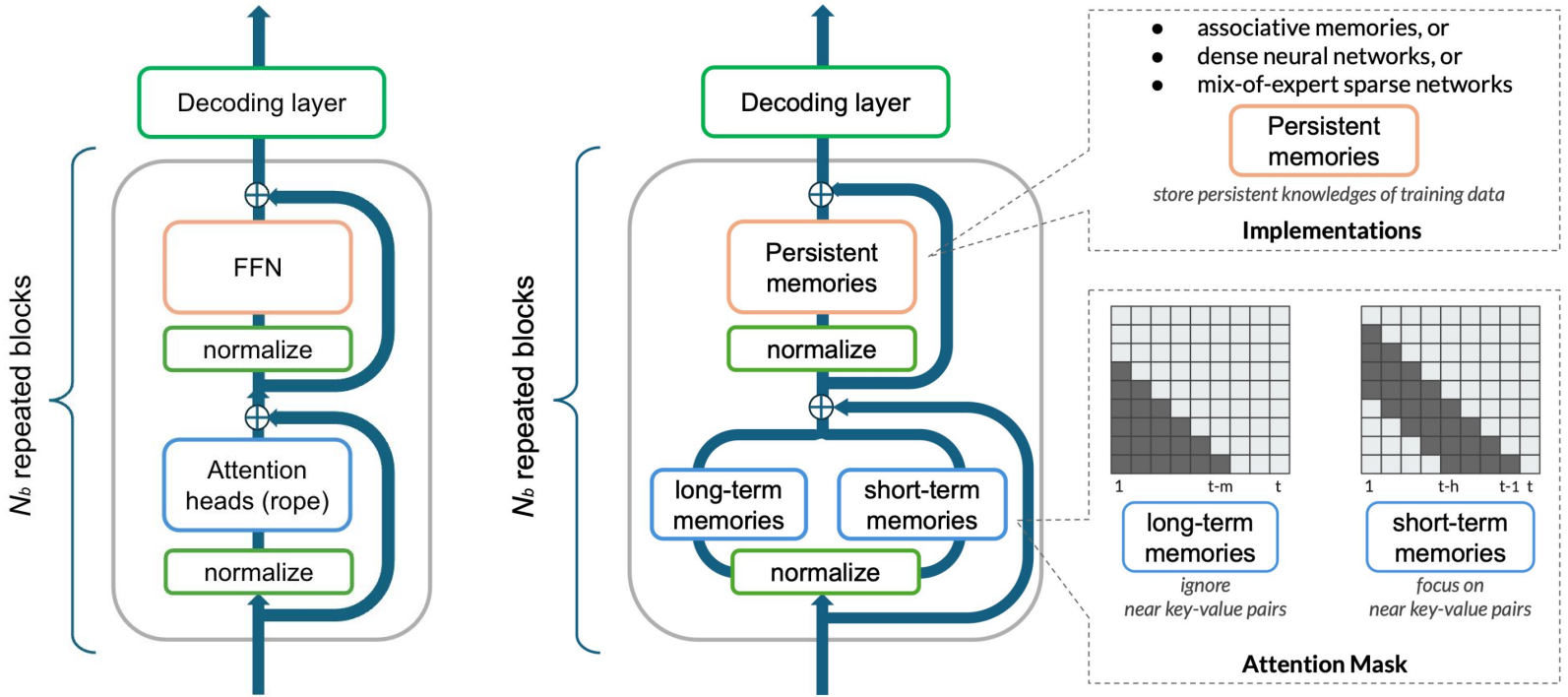
1. This argument was first suggested by Pascal Vincent during my Memory Mosaic talk at the FAIR Lab Offsite in June 2024. Thus, the speaker thanks Pascal Vincent for his insightful comments.

Now, I am going to introduce an attempt to build such inference-time learning machine in practice, named Memory Mosaics v2.

Memory Mosaics v2 Architecture

Zhang, J., & Bottou, L. (2025). memory mosaics at scale. *under review*.
Zhang, J. (2025). AI for the Open-World: the Learning Principles. arXiv preprint arXiv:2504.14751.

3-level Memories

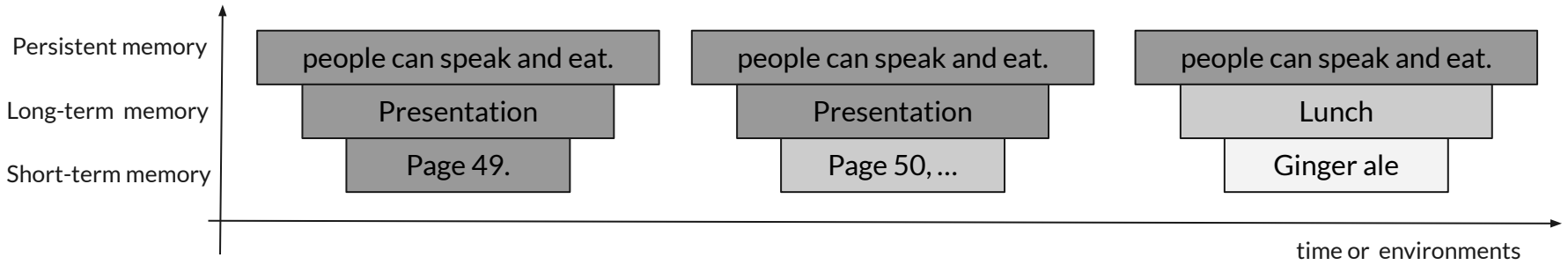


Transformer

Memory Mosaics v2

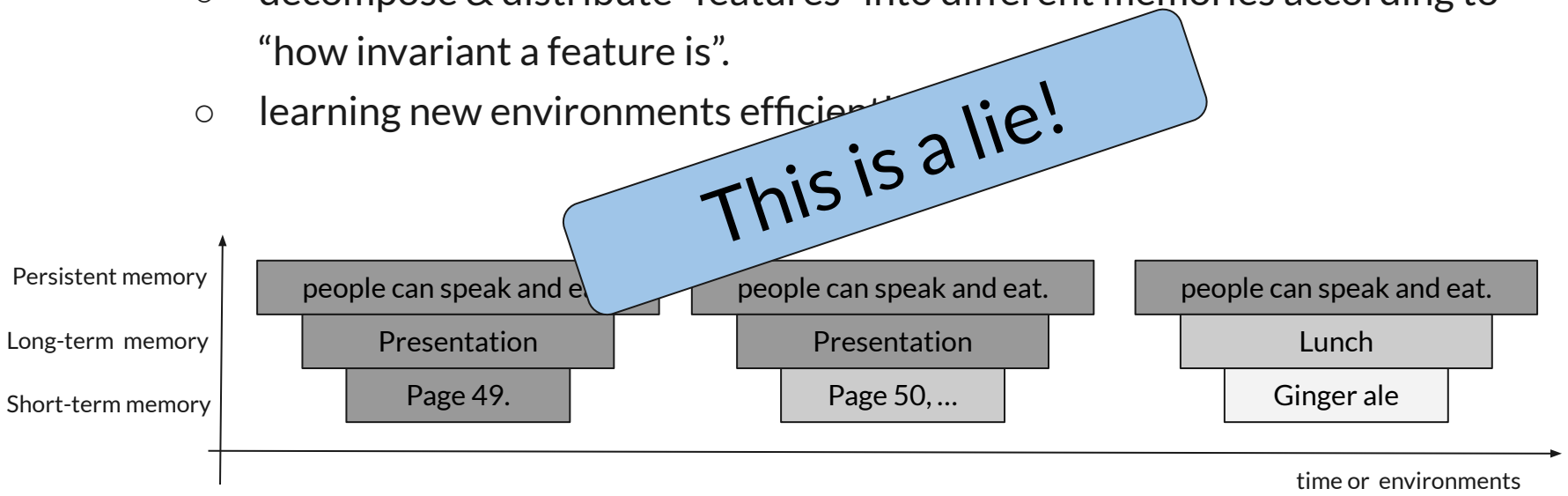
Why 3-level Memories?

- Goal of design:
  - decompose & distribute “features” into different memories according to “how invariant a feature is”.
  - learning new environments efficiently.



Why 3-level Memories?

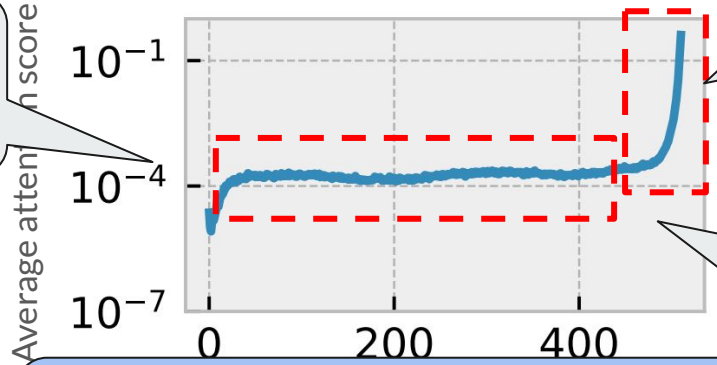
- Goal of design:
  - decompose & distribute “features” into different memories according to “how invariant a feature is”.
  - learning new environments efficien



Why 3-level Memories?

- The story is actually experiments driven.
- We tried to merge long-term and short-term memories as one memory.
- And got the following attention distribution of the last token:

Avg attention scores at far distance are “flat”. (Independent of position)



Avg attention scores at close are “peaky” (depend on position)

Two different regimes. Separating them helps a lot when learning new tasks.

Thus we distribute the “flat” and “peaky” parts into long-term & short-term memories, create 3-level memories.

Adaptive bandwidth (in gaussian kernel smoothing)

- Memory stores key-value pairs, retrieves values given a key.

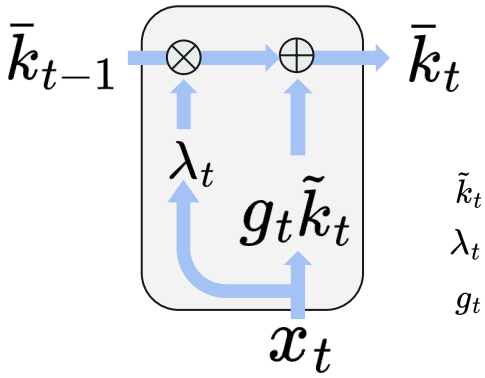
$$f(k, \{(k_1, v_1), \ldots, (k_n, v_n)\}) = \sum_{i=1}^{n} \frac{e^{\beta k^\top k_i}}{\sum_{j=1}^{n} e^{\beta k^\top k_j}} v_i$$

- As the number of examples in memory increases (n), the inverse bandwidth parameter β of kernel smoothing must be increased to sharpen the response.

$$\beta = \beta_1 n^\alpha + \beta_0$$

Feature extractor of key and value

- Key represents the recent past.
  - Memory mosaics compute key by normalizing $\bar{k}_t, k_t = norm(\bar{k}_t)$, where $\bar{k}_t$ comes from the right RNN process.
- Value represents the near future.
  - Memory Mosaics compute value by normalizing $\bar{v}_t, v_t = norm(\bar{v}_t)$, where $\bar{v}_t = W_v x_t + \lambda W_v x_{t+1}$



$$\tilde{k}_t = W_k x_t$$
$$\lambda_t = exp(-|W_\lambda x_t|)$$
$$g_t = exp(W_g x_t)$$

rwkv [1] kinds of process

In a preliminary small scale (1.5B) experiment, this key feature extractor provide ~0.5% on common tasks, 8% on ruler long-context tasks (train on 4k, evaluate on 32k).

[1] Peng, Bo, et al. "Rwkv: Reinventing rnns for the transformer era." *arXiv preprint arXiv:2305.13048* (2023).

Evaluation dimensions

In-distribution

out-of-distribution

- **Persistent**-knowledge storing and retrieval
  - Knowledge learned from training data
  - Benchmarks: MMLU, arc_challenge, gsm8k, etc
- **New**-knowledge storing and retrieval
  - Knowledge learned from inference data
  - Benchmarks: RULER question-answering (“needle in a haystack”).
- **In-context** learning
  - The ability to learn a new distribution/task, rather than simply store and retrieve new-knowledge.
  - Tasks: multi-class classification with semantic labels (e.g. ‘dog’, ‘cat’) or anonymous labels (e.g. ‘class-1’, ‘class-2’)

√

IID regime.
More data + larger model = better performance
(not our focus)

The prerequisite of effective in-context learning.
Imaging a poor goldfish with only 7 seconds memory, how can it learn a 90-mins movie?

OOD regime.
*The new task could be **never seen** in the training data, or even **conflict with** training data.*
(interesting)

Models to compare

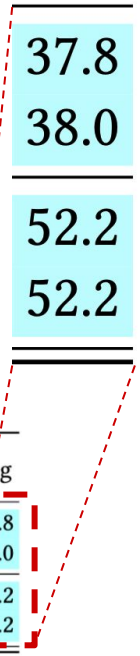
- **Memory Mosaics v2** small & **Transformer** small^ (~Llama-1.5B size)
  - 24 layers, hidden dimension 2048, trained on 200B tokens
- Memory Mosaics v2 large & Transformer large (~Llama-8B size)
  - 32 layers, hidden dimension 4096, trained on 1T tokens
- All models are pretrained on 4K context length. (marked as **“4k”**)
- Then fine-tuned on 32k context length. (marked as **“32k”**).
- We choose Transformer as the only baseline, because other approaches (*RNNs*, *LSTM*, *state-space model*) do not work at all.

^. Memory Mosaics contains some additional parameters than transformer due to the explicit working memory & short-term memory design. For the easy of explanation, I simply names model size as “small” and “large”. I will show later that the additional parameters are worthful.

Persistent-knowledge storing and retrieval

- **Task Description:** qa, arch_easy, mmlu, math, etc (19 common tasks)
- **Results:**
  - Memory Mosaics and Transformer **share the same "persistent memory"** architecture (i.e. FFN with SwiGLU),
  - **perform closely** in persistent-knowledge storing and retrieval (after extending to 32k context length).

model	context length	obqa	arc easy	wino-grande	arc challenge	siqa	piqa	boolq	hell-aswag	nq	tqa	squad	mbpp	math	gsm8k	mmlu alt	human eval+	race high	race middle	bbh	avg
transformer small	32k	35.2	61.0	60.1	31.4	44.5	73.6	63.0	59.3	11.7	26.7	54.7	9.2	1.2	3.0	35.2	32.4	37.4	52.2	26.0	37.8
memory mosaics small	32k	35.0	60.1	59.0	33.0	46.6	73.0	62.8	57.9	11.7	29.5	59.2	9.3	1.1	2.4	34.7	31.2	38.3	49.4	27.1	38.0
transformer large	32k	45.8	77.3	72.3	52.6	49.3	80.8	72.6	79.2	31.9	61.5	76.3	9.8	8.7	32.4	49.0	38.3	45.6	62.6	45.6	52.2
memory mosaics large	32k	45.4	78.0	71.2	51.8	48.6	80.4	73.1	78.6	30.9	62.0	78.2	9.6	8.8	27.4	48.2	43.0	46.5	61.6	47.8	52.2



- **Results:**
  - For many common tasks, the long-term memory in Memory Mosaics v2 **does not even fire!**
  - We can safely **remove the long-term memory** in Memory Mosaics v2 after training without hurting the performance. (but reduce parameters and computation)

	params	flops/token	obqa	arc easy	wino-grande	arc challenge	siqa	piqa	boolq	hell-aswag	nq	tqa	avg
Transformer Large	8.8B	16.7B	45.8	77.3	72.3	52.6	49.3	80.8	72.6	79.2	31.9	61.5	62.3
Memory Mosaics v2 Large Without long-term memory	8.3B	15.6B	45.6	77.9	72.0	52.1	49.4	80.5	73.1	78.8	30.3	61.6	62.1
Memory Mosaics v2 Large	9.9B	18.9B	45.6	77.9	72.0	52.1	49.4	80.5	73.4	78.7	29.6	61.7	62.1

flops/token is estimated at context length 256.

New-knowledge storing and retrieval

- **Task description** (A question-answering task from RULER [Hsieh et al, 2024])**:**
  - **Multiple articles** are concatenated, followed by a question conspondings to **one random article.**
  - **Example:** *"Answer the question based on the given documents. The following are given documents. Document 1: [...] Document2: [...][...] Document 20: [...]Question: What religion were the Normans? Answer:"*
  - Memory compression algorithms, *e.g. RNNs, LSTM, state-space models*, fail on this task by construction.
  - 'Local window' memory, *e.g. Alibi position-encoding, local-window attention*, fails on this task by construction.

Hsieh, Cheng-Ping, et al. "RULER: What's the Real Context Size of Your Long-Context Language Models?." *arXiv preprint arXiv:2404.06654* (2024).

New-knowledge storage and retrieval

- **Results:**

Trained on 4k context length, **No fine-tuning.**

model	context length	4k	8k	16k	32k
transformer small	4k	39.4	×	×	×
memory mosaics small	4k	45.0	35.0	34.1	31.7
transformer large	4k	57.7	×	×	×
memory mosaics large	4k	59.3	48.8	46.4	26.5

Memory Mosaics can easily **extrapolate context length** x4 ~ x8 times.

Memory Mosaics v2 can effectively store and retrieve new-knowledge!

Trained on 4k context length, **Fine-tuning on 32k.**

model	context length	4k	8k	16k	32k	64k
transformer small	32k	37.0	29.3	29.0	22.1	×
memory mosaics small	32k	44.9	39.2	39.8	37.3	25.3
transformer large	32k	51.2	48.8	44.7	41.1	×
memory mosaics large	32k	58.9	55.5	54.9	53.4	46.4

Memory Mosaics outperforms Transformer >10% (53.4% vs 41.1%)

In-context learning

- **Task description:** Multi-class classification with **semantic** or **anonymous labels**
  - Input **X** example:
    - *"My bank transfer is still not showing up in my account."*
  - Target label **Y** example:
    - **Semantic** label : *"balance_not_updated_after_bank_transfer"*
    - **Anonymous** label: *"class_71"*
  - **Anonymous label** tasks heavily rely on the learning of "inference data".
  - **Prompt example**: "*Given a customer service query, please predict the intent of the query. [...] The examples are as follows: query: [X1], instant: [Y1], [...], query:[X], instant:*"
  - To reduce the influence of **prompt strategies**, we sweep various prompts strategies and choose the best for each model.

- **2) Results:**

Banking Intent Classification
https://paperswithcode.com/dataset/banking77

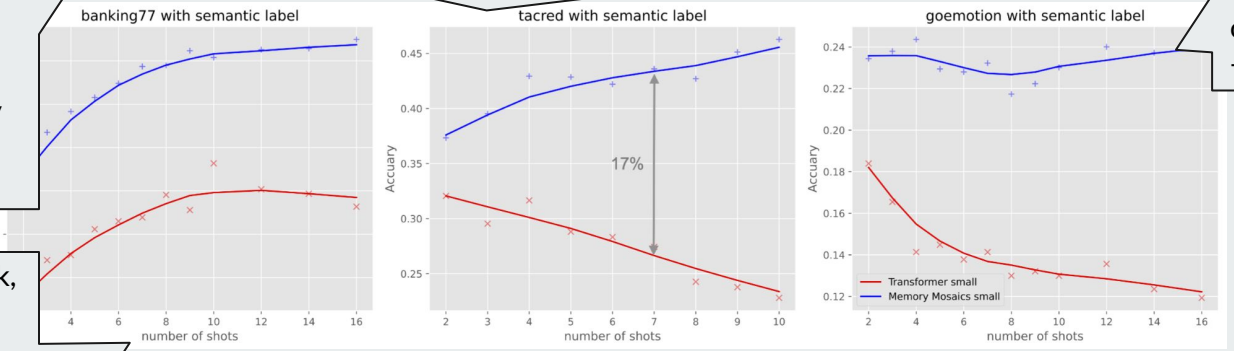
- X: is a bank-domain customer service query.
- Y: indicate the category of the query.
- 77 categories in total.

Relation Classification
https://paperswithcode.com/dataset/tacred

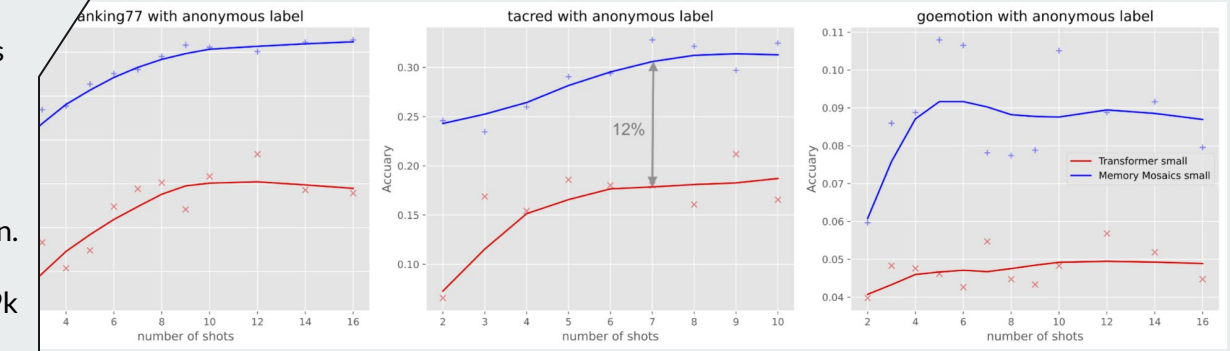
- X: is a sentence extracted from newswire or web text.
- Y: indicates the relationship of two objects in the sentence.
- 41 relation categories

Emotion Classification
https://paperswithcode.com/dataset/goemotions

- X: is a Reddit comment
- Y: indicates the emotion category.
- 28 categories in total.



With **semantic label.** E.g. Y= "*balance_not_updated_after_bank_transfer*"



With **anonymous label.** E.g. Y= *"class_71"*

In the plot of banking task,

- **horizontal-axis:** indicates **the number of demonstration shots**, where each shot contains 77 [X, Y] examples with different categories.

- **vertical-axis:** indicates the accuracy of prediction.

The context length is ~29k when demonstration shots = 16.

- **2) Results:**

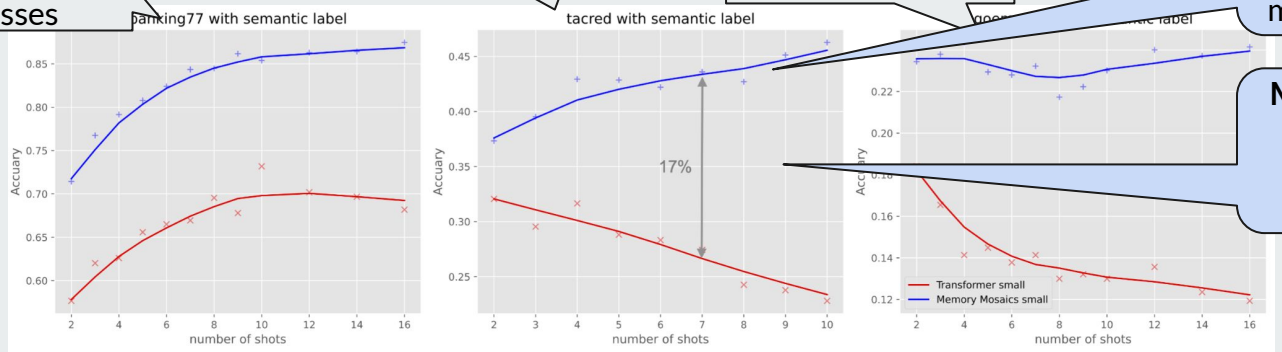
Banking Intent Classification 77 classes

Relation Classification 41 classes

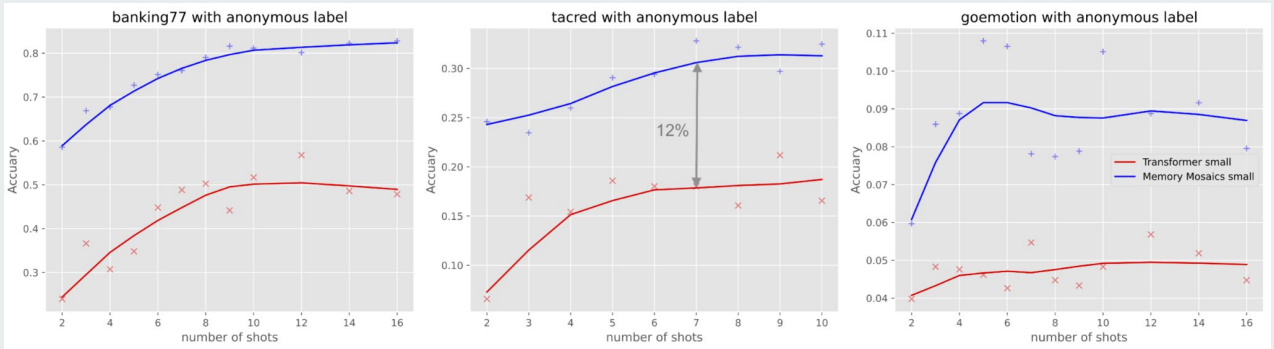
Emotion Classification 28 classes

Memory Mosaics: Increased performance as more data coming

Memory Mosaics outperforms **Transformer** by 10%~20%

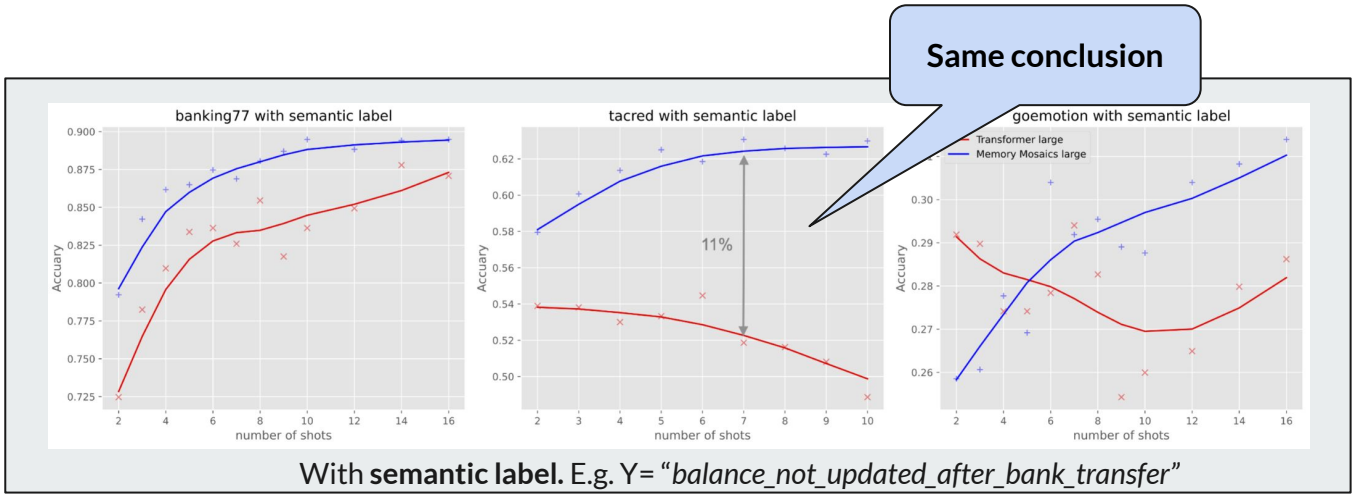


With **semantic label.** E.g. Y= "*balance_not_updated_after_bank_transfer*"

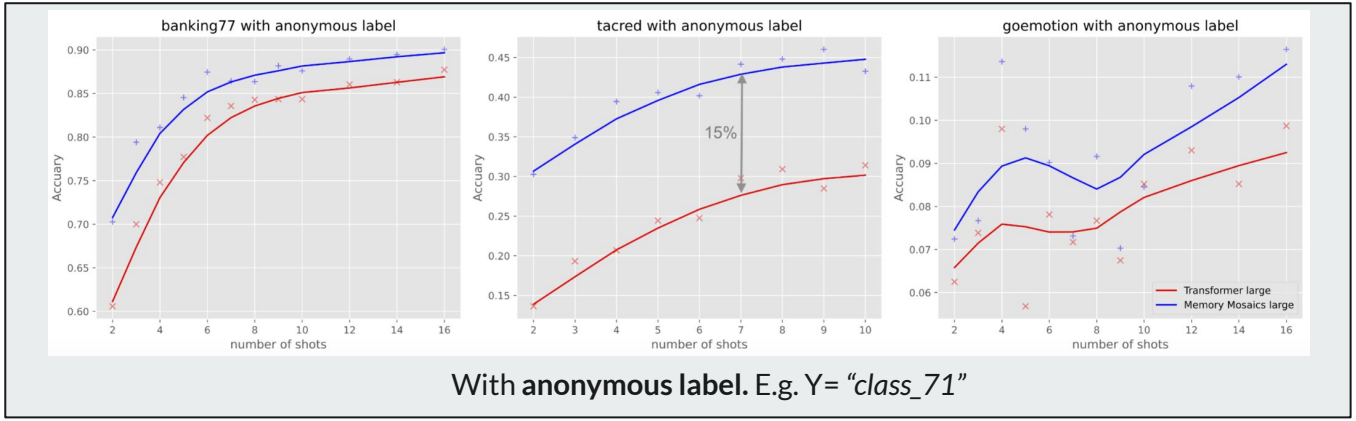


With **anonymous label.** E.g. Y= *"class_71"*

- **Results:** Same comparison on large networks (~8B parameters)



With **semantic label.** E.g. Y= "*balance_not_updated_after_bank_transfer*"



With **anonymous label.** E.g. Y= *"class_71"*

“Training on everything” or inference-time learning?

- To train a huge foundational model, one can either:
  - brutally invest **more money** (GPUs and data), and simply **reuse old recipe** (architecture, etc). *E.g. the mainstream training on everything*
  - or try smart new techniques. *E.g. Memory Mosaics v2, …*
- It might be hard to make a unique decision for all scenarios.
- Let’s have a simple & brutal comparison to help this decision making:

How much additional data does the **transformer** recipe approach need to match the performance of **memory mosaics v2**?

How much does **transformer** cost to match **memory mosaics** ?

- Train **Memory Mosaics large** with 1T tokens,
- Train **Transformer large** with 200B, 1T, 8T tokens.
- Evaluate on both the **new-knowledge storing & retrieval** and **in-context learning** tasks.

How much does **transformer** cost to match **memory mosaics** ?

- New-knowledge storing and retrieval.

Trained on **4k** context length,
Fine-tuning on **32k.**
Evaluate on **4k~64k.**

model	context length	train tokens	4k	8k	16k	32k	64k
transformer large	32k	200B	48.6	42.9	40.7	33.8	×
transformer large	32k	1T	51.2	48.8	44.7	41.1	×
transformer large*	32k	8T	59.2	54.5	50.9	46.9	×
memory mosaics v2 large	32k	1T	58.9	55.5	54.9	53.4	46.4

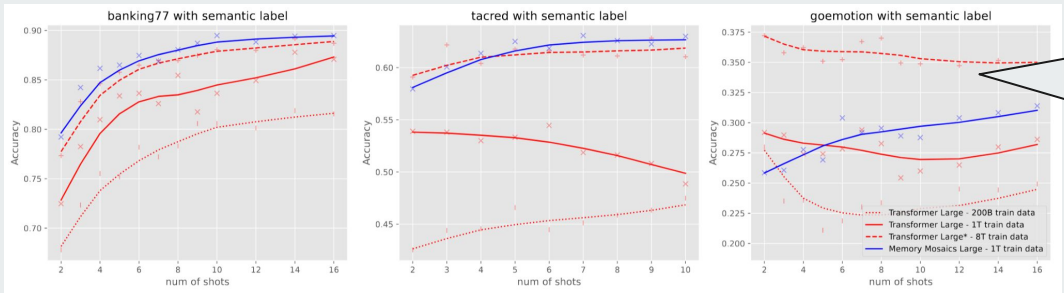
Transformer lags behind Memory Mosaics by 12.3% (41.1% vs 53.4%)

Transformer with **x8 times training data** still lags behind Memory Mosaics **by 6.5%**

How much data does transformer need to match Memory Mosaics? 16T? 32T?
We don't have enough data. We have to do it smartly.

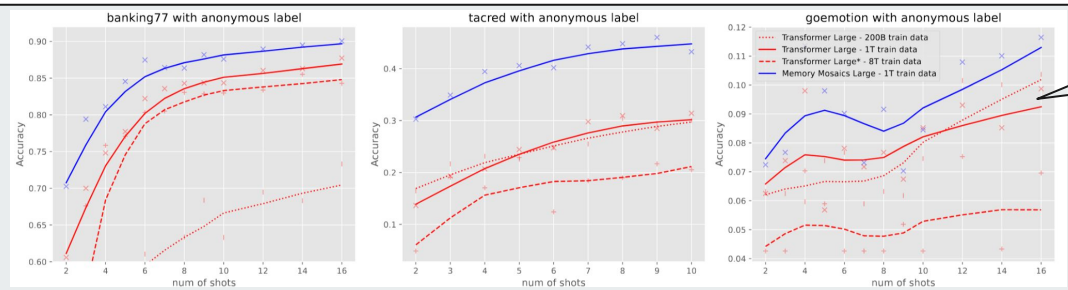
How much does **transformer** cost to match **memory mosaics** ?

- In-context learning



With **semantic label.** E.g. Y= "*balance_not_updated_after_bank_transfer*"

Semantic label tasks: Transformer with **x8 times data** starts to **match** the performance of Memory Mosaics



With **anonymous label.** E.g. Y= *"class_71"*

Anonymous label tasks: **No way!**

Inference-time learning

- Why study inference-time learning?
  - Reduce the relying of priori knowledge from human designers
- Build model to perform inference-time learning
  - Memory Mosaics v2
- Gap between the “training on everything” and inference-time learning.
  - > 8 times training data
- Opportunities
  - Replace long-term memory with hash-table or hierarchical memory to cut inference cost.

Conclusion

- AI for the open-world is the Intelligence of machine.
- AI for the open-world requires unique **principles** and **techniques**.
  - **Rich Features,** helps the learning of a **broader range** of unseen tasks;
  - **Disentangled representation,** reduces the **number of examples** required on unseen tasks
  - **inference-time learning**, reduces the relying of designers' **priori knowledge**.
- This is still the early research stage of AI for the open-world.
  - It costs 5 years to explore, probably another 5-30 years to complete.
- A huge research space in AI for the open-world is opening for us!