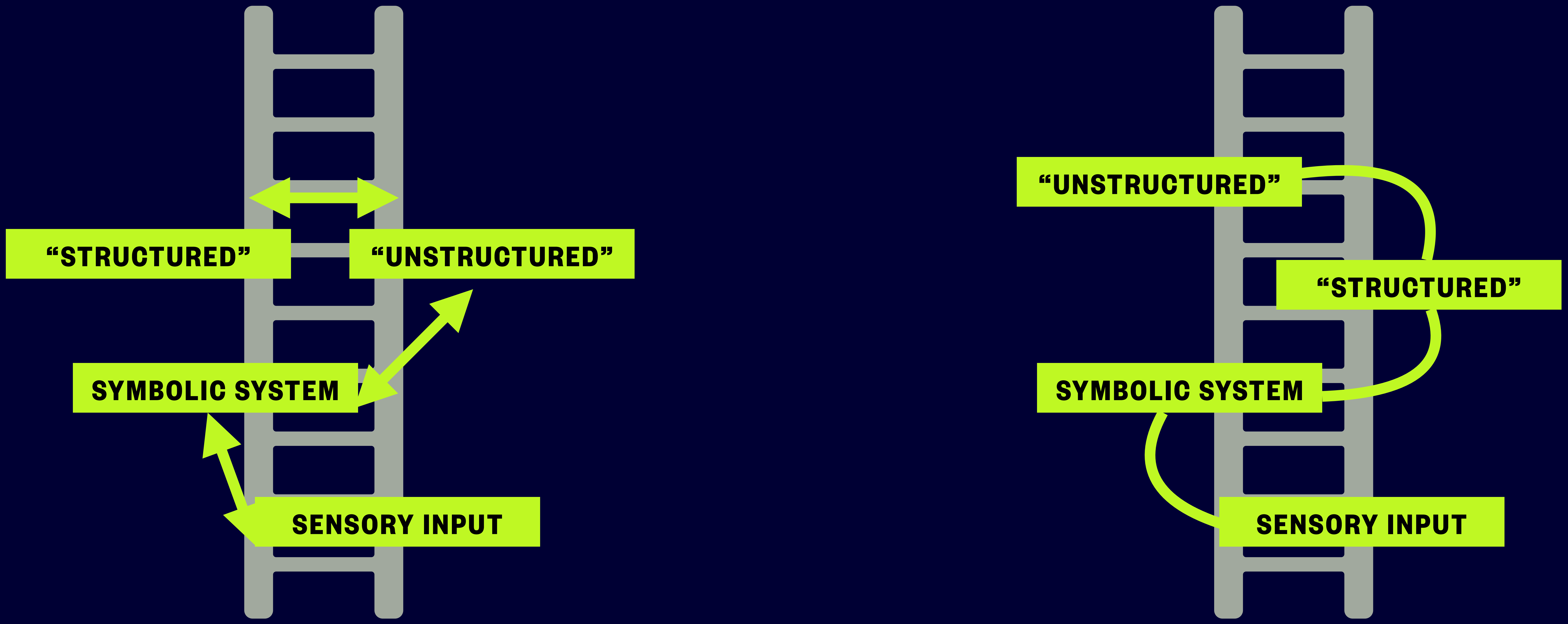


Bridging Structured and Unstructured Learning in Natural Language Processing

Yihong Chen 07/2024 Meta

Personal Briefing
Yihong Chen

- Selected Publication
  - Generalization on unseen and rare XYZ
    - Improving Language Plasticity via Pretraining with Active Forgetting (NeurIPS 2023)
    - ReFactorGNNs: Revisiting Factorisation-based Models from a Message-Passing Perspective (NeurIPS 2022)
    - λopt: Learn to Regularize Recommender Models in Finer Levels (KDD 2019)
  - Self-supervised learning
    - Relation Prediction as an Auxiliary Training Objective for Improving Multi-Relational Graph Representations (AKBC 2021)
  - Efficient model training/adaptation
    - Breaking Physical and Linguistic Borders: Multilingual Federated Prompt Tuning for Low-Resource Languages (ICLR 2024)
    - Mini-Model Adaptation: Efficiently Extending Pretrained Models to New Languages via Aligned Shallow Training (ACL 2023)
    - Learnable Embedding Sizes for Recommender Systems (ICLR 2021)
  - Conversational agents
    - You impress me: Dialogue generation via mutual persona perception (ACL 2020)
    - Learning-to-ask: Knowledge Acquisition via 20 Questions (KDD 2019)

- Research Areas
  - Natural Language Processing
    - knowledge graphs
    - language models
- Education
  - Undergraduate and master's at EE Tsinghua
  - PhD at UCL and Meta
- Collaborators

Research Theme

Towards AI systems with more controllability

- The history of AI has come a long way but are we there yet?
    - From expert systems to deep learning
    - Now it seems that everything converges to language models, LLMs!
        - structured rule-based based AI → unstructured data-reliant AI?
- LLMs are awesome
    - They are trained and inferences in continuous spaces which is good for scaling up and free-form generation!
- However, once you start examining the generation from RAW LLMs,
    - They can be hard to control: hallucination, bias, toxicity, “magic” etc.
    - Moreover, LLM weights are static snapshotting *partial* reality at a certain time point.
    - Our reality/values/needs is always *evolving*.
    - These giant models quickly evolve to our latest reality/values/needs.

Research Theme

Towards AI systems with more controllability

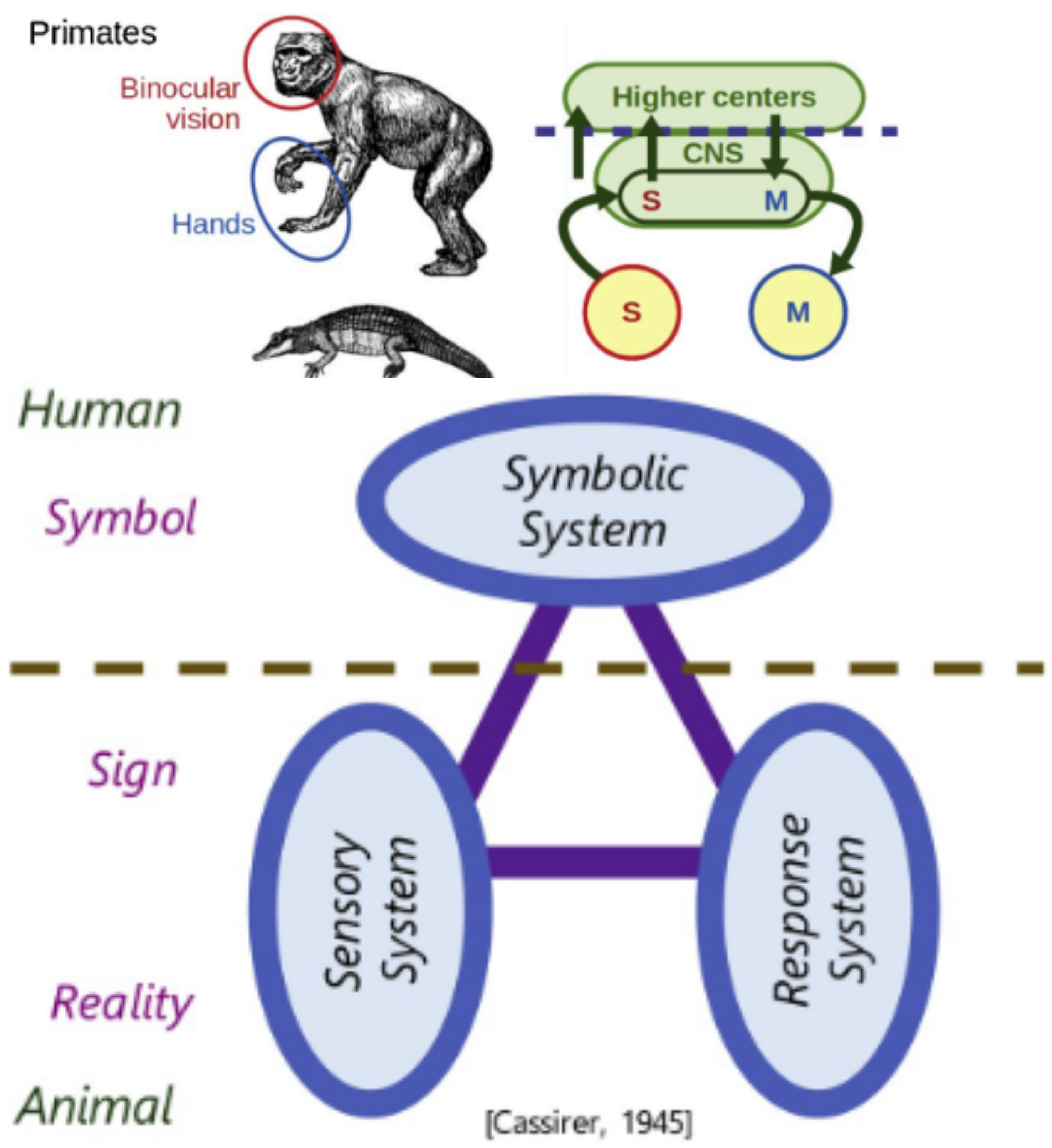
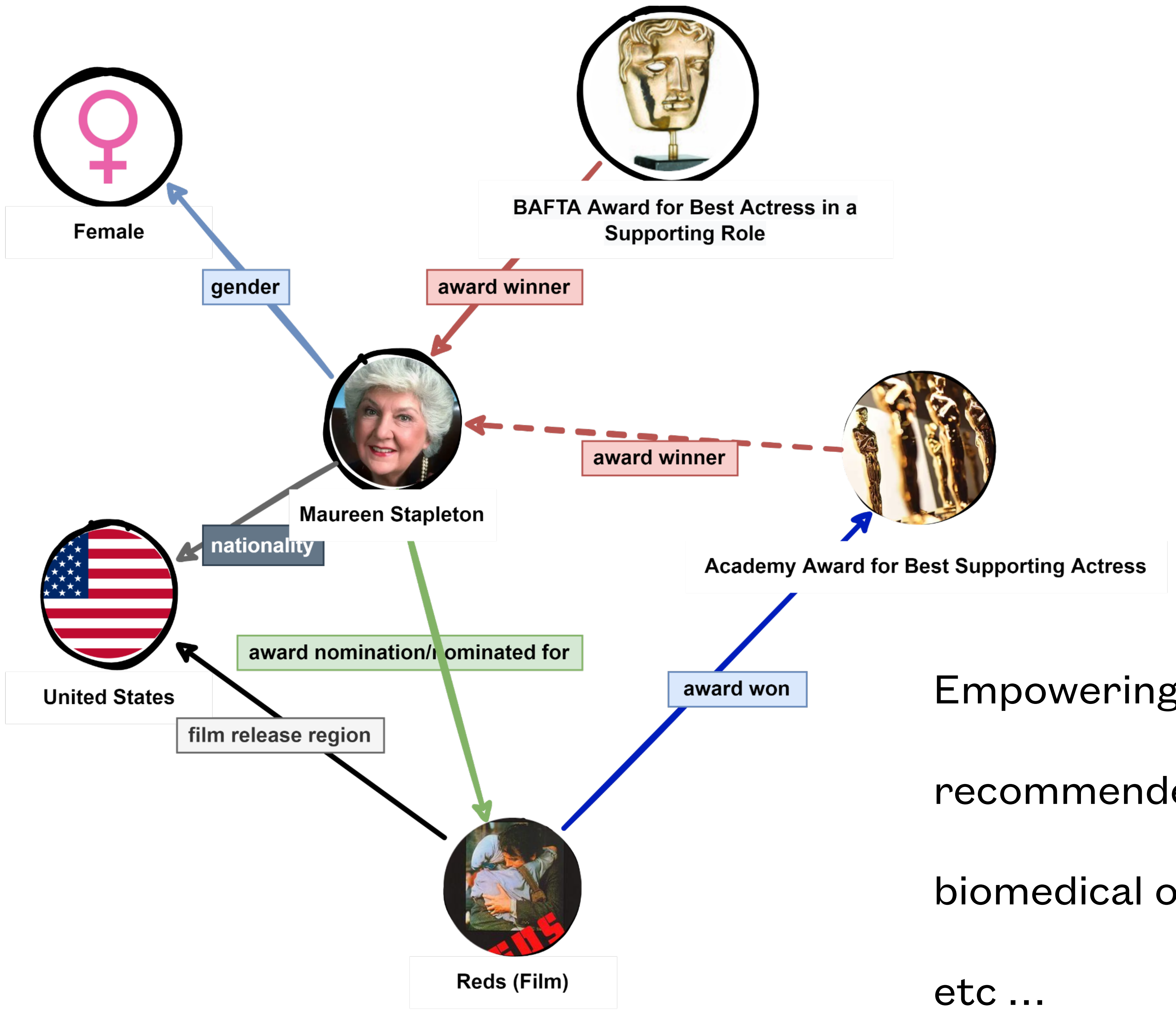


image source: Rafael Vieira Bretas, Yumiko Yamazaki, Atsushi Iriki *Phase transitions of brain evolution that produced human language and beyond*, Neuroscience Research 2020

- In order to progress from such naive "continuous space reasoning"
  - Approach 1: **Scaling**
    - continue retraining/pretraining with more data and more frequently
  - Approach 2: Mimicking "natural" intelligence, which has gone through sensory to *symbolic* evolution
    - allowed planning and reasoning to happen *before* motion
    - and fast adaptation to new environments with *tools* developed in old environments
  - augmenting LLMs with xyz
    - RAG
    - CoT
    - Tools
    - Magic prompts, data mixture, synthetic data prompt …
    - Great, but not that easy to control …

Research Theme

Controlling via a symbolic system to *structure* the reality



How about knowledge graphs ???

Empowering Google search,

recommender systems,

biomedical ontological reasoning

etc ...

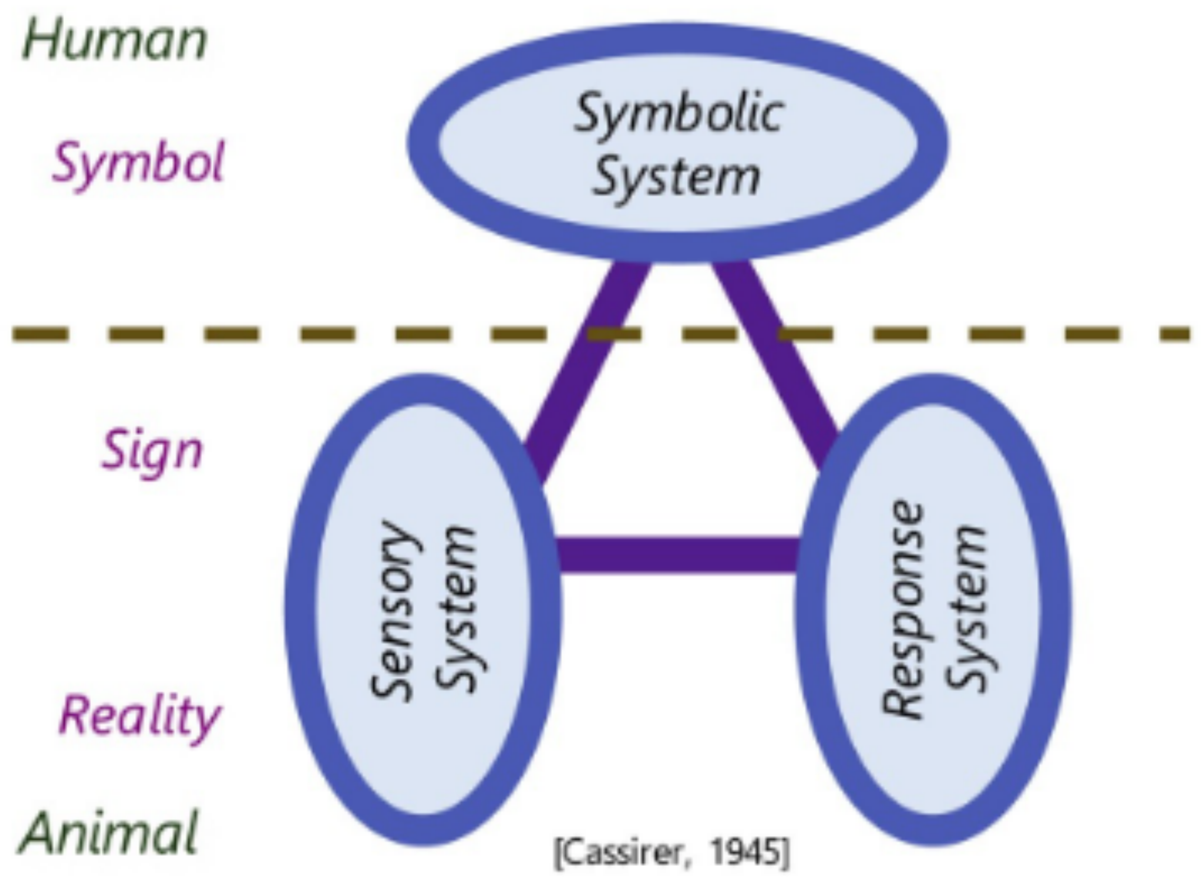
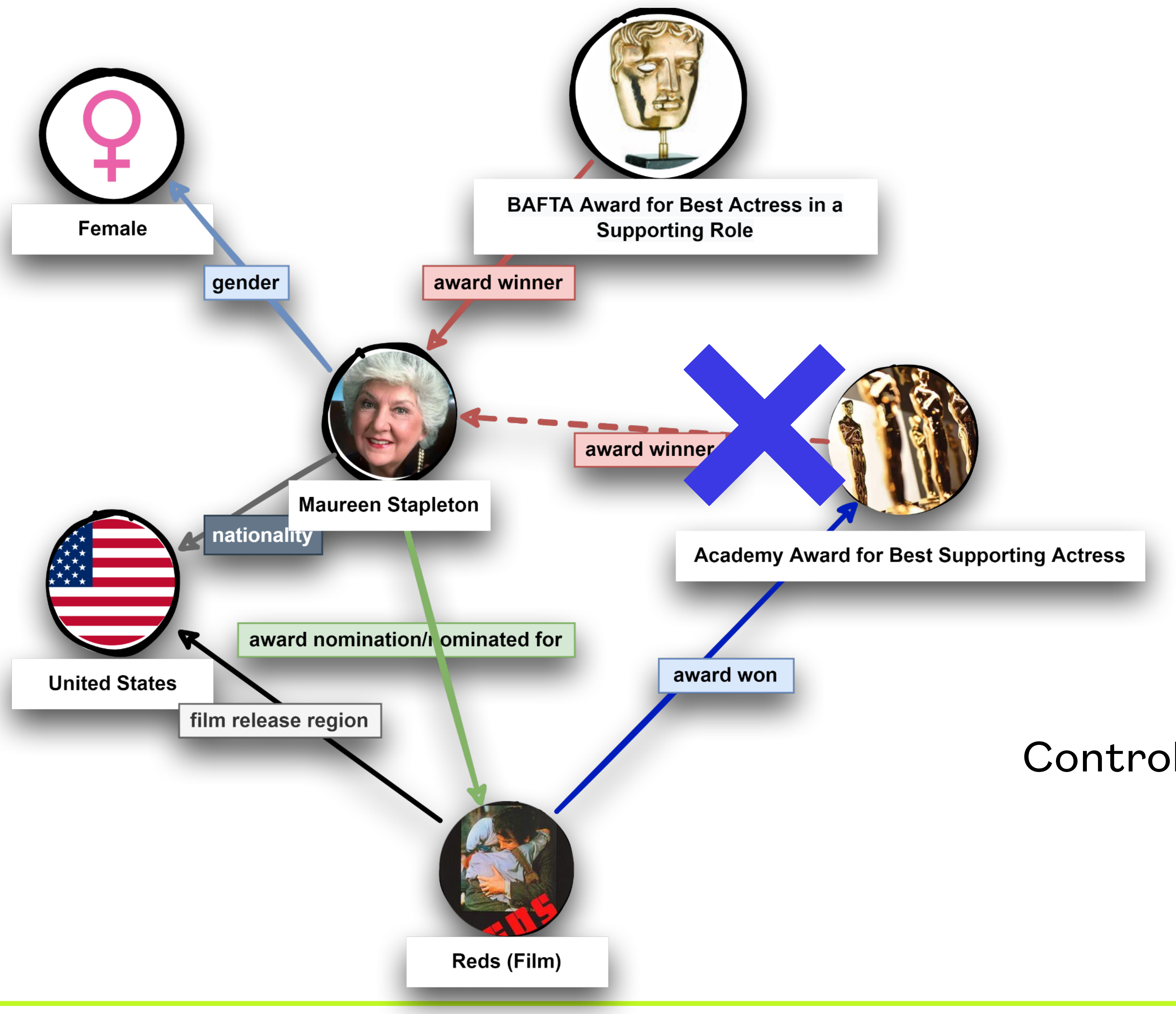


image source: Rafael Vieira Bretas, Yumiko Yamazaki, Atsushi Iriki. *Phase transitions of brain evolution that produced human language and beyond*, Neuroscience Research 2020

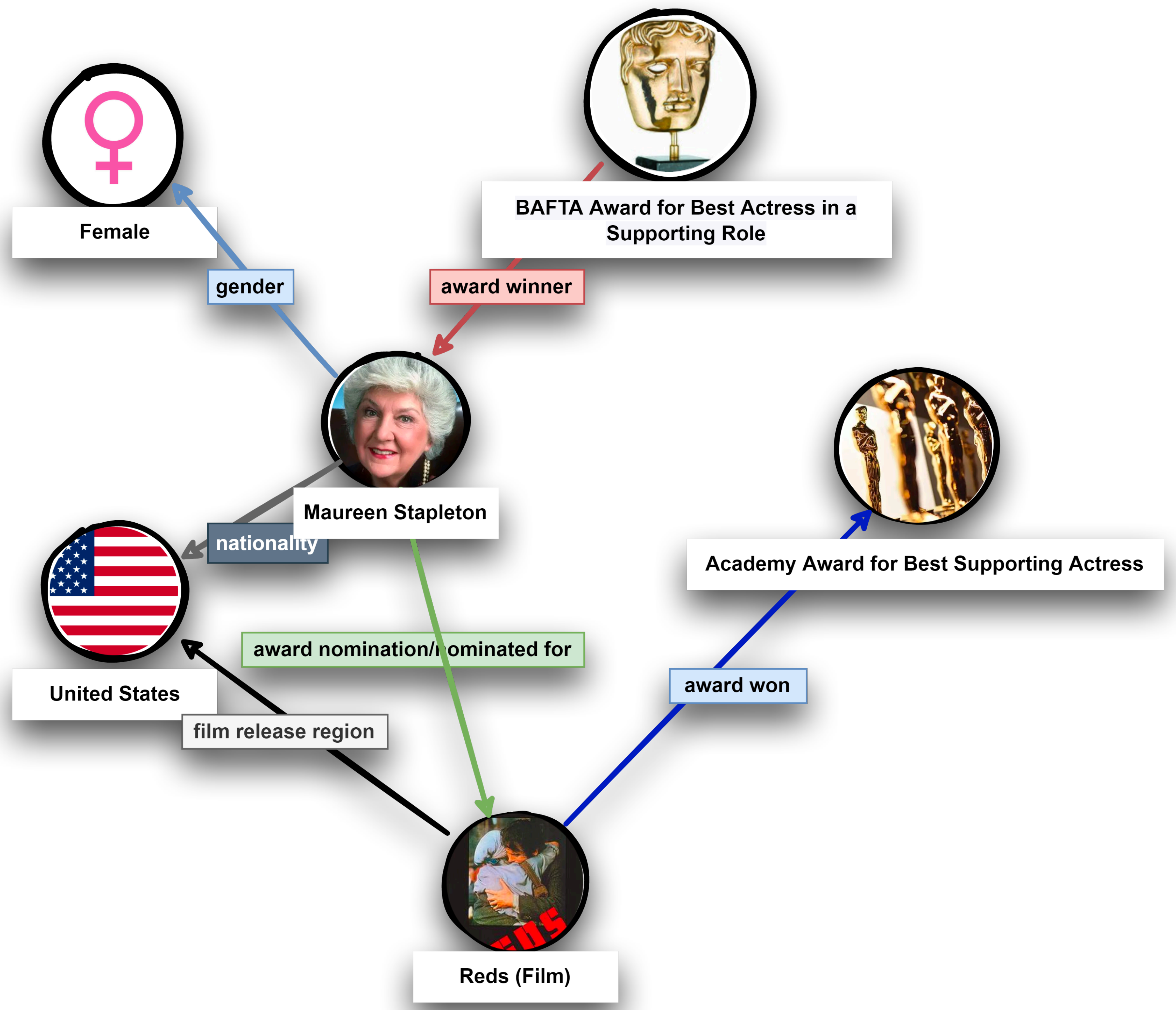
Research Theme

Controllability via a symbolic system to *structure* the reality

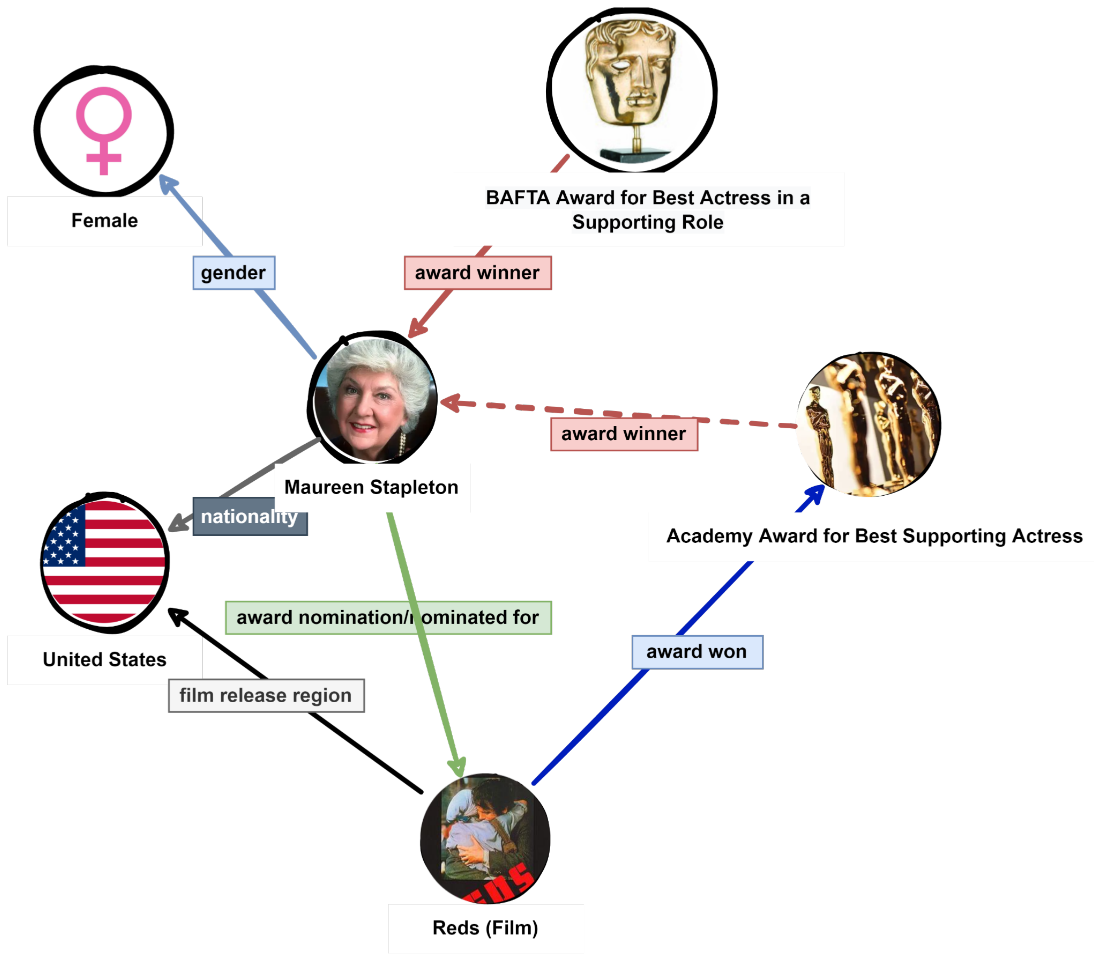


Controlling: delete, edit, update

As easy as overwriting local structures

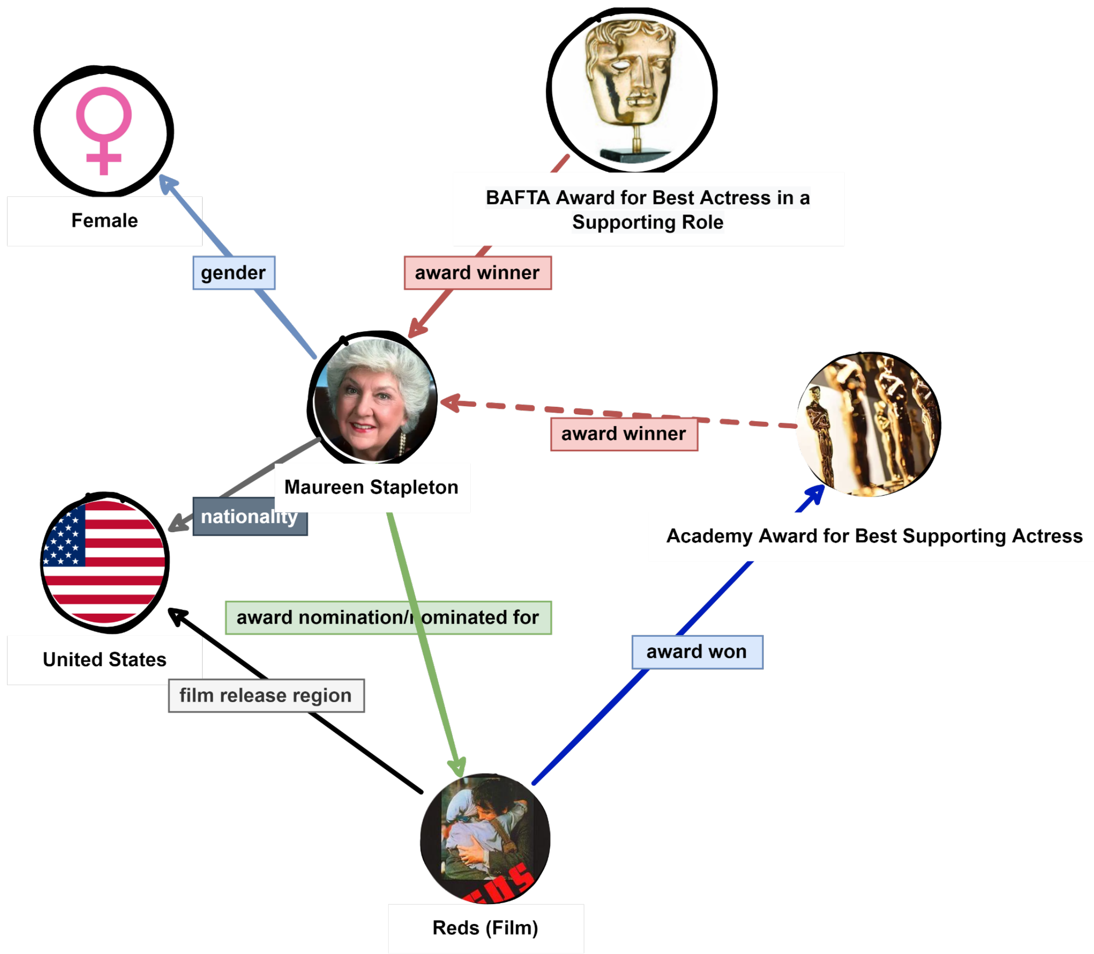


Structured vs Unstructured



	Pros	Cons
Structured	controllable (easy to update/ edit/remove), interpretable, reasoning, planning	construction cost, missing entries
Unstructured	generative! (can create answers for any questions), ingest huge data	hard to control (hallucination/ toxicity), expensive

Structured vs Unstructured



	Structured	Unstructured
Data Format	knowledge graph (KG), ontology etc	free-form text
Model Architecture	factorization, GNNs	Transformer-based language models
Learning Objective	entity prediction	(masked) language modeling

Bridge the two learning paradigms

However both systems are symbolic.

- For unstructured learning, in LLMs, the symbols are the tokens from each vocabulary of the language.
- For structured learning, in knowledge graphs the symbols are the entities/relations in each vocabulary of the graph

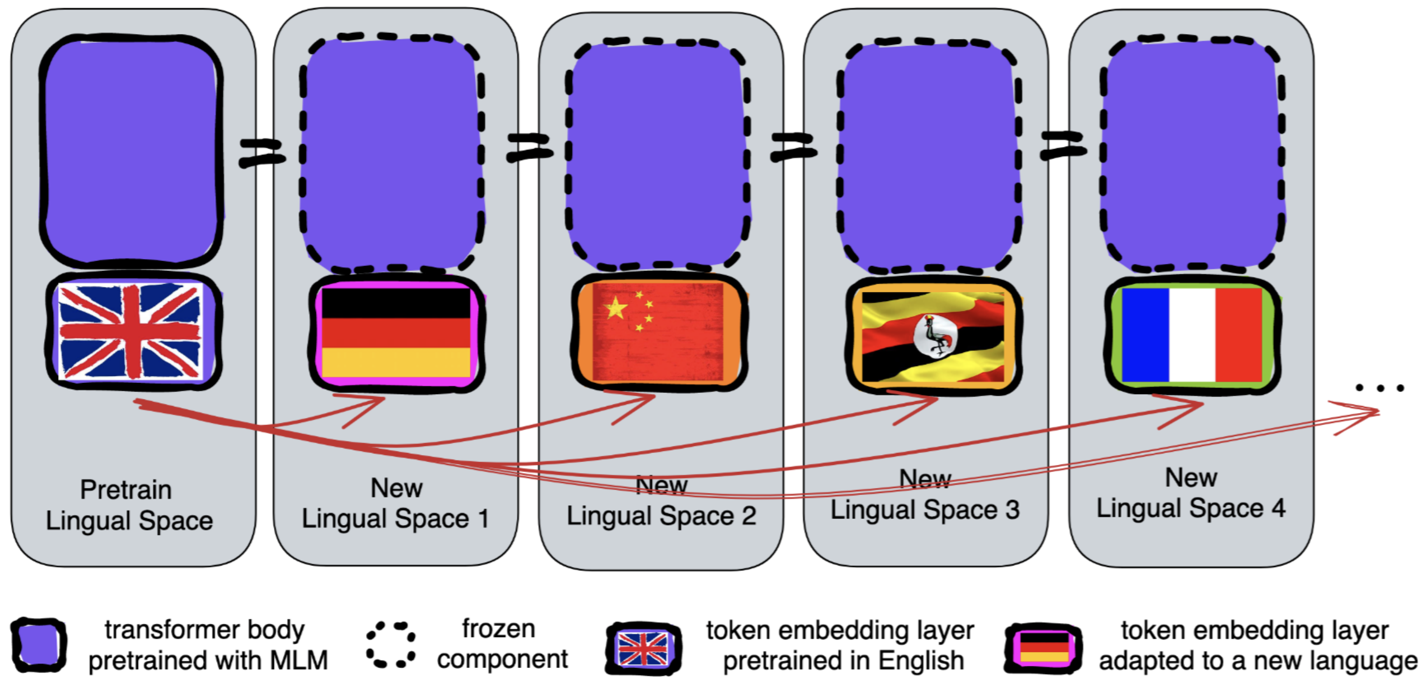
The difference is only in

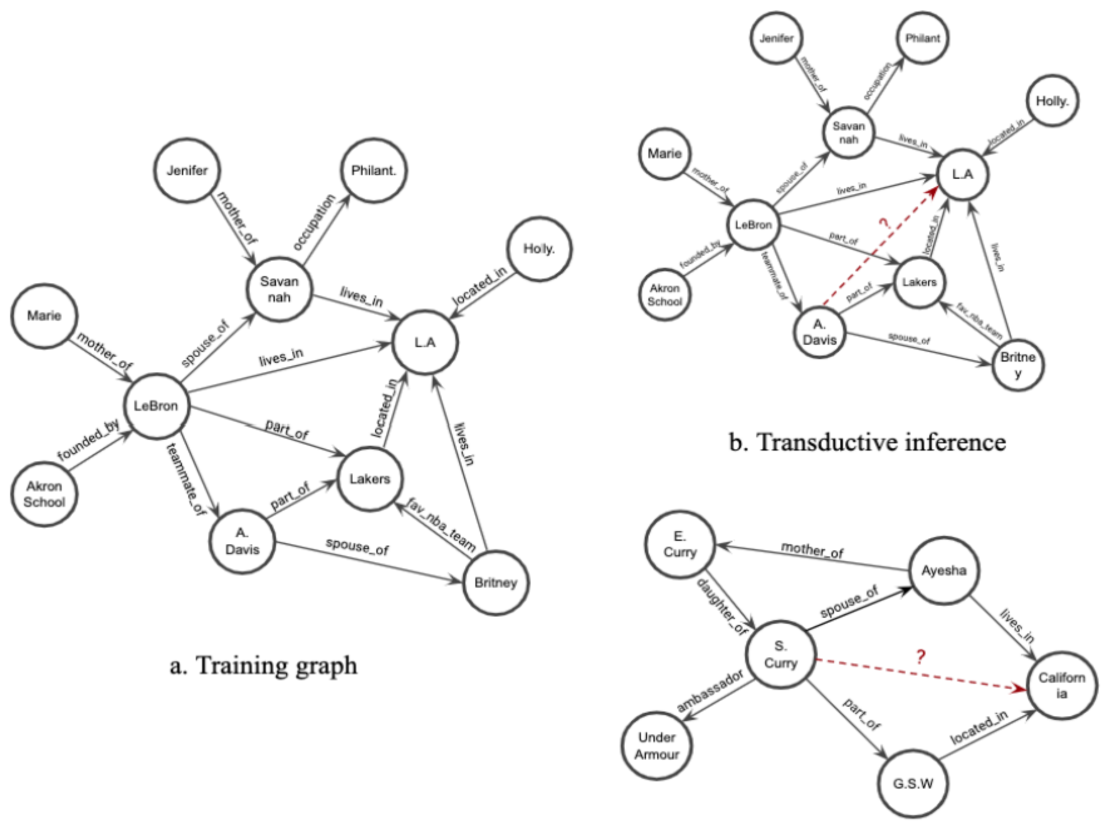
- Granularity of symbols
- Prebuilt structures (which characterizes the interaction between symbols)

	Structured	Unstructured
Expected Outcome	Find a good tradeoff between "representation" and its enabled "computation"	
Learning Objective	(Masked) language modeling works for both! [1]	
Architecture	Embeddings + "Body" + (Un)Embeddings	
Generalization	Embedding resetting increases model plasticity for both [2] [3]	
Interpretability	Un-cache the compute stored in embeddings leads to data graph reconstruction for both (under review)	

[1] CHEN ET AL 2021 RELATION PREDICTION AS AN AUXILIARY TRAINING OBJECTIVE FOR IMPROVING MULTI-RELATIONAL GRAPH REPRESENTATIONS.
[2] CHEN ET AL 2022 REFACTOR GNNS: REVISITING FACTORISATION-BASED MODELS FROM A MESSAGE-PASSING PERSPECTIVE
[3] CHEN ET AL 2023 IMPROVING LANGUAGE PLASTICITY VIA PRETRAINING WITH ACTIVE FORGETTING

Bridge the two learning paradigms





a. Training graph

b. Transductive inference

c. Inductive inference

	Structured	Unstructured
Expected Outcome	Find a good tradeoff between "representation" and its enabled "computation"	
Learning Objective	(Masked) language modeling works for both [1]	
Architecture	Embeddings + "Body" + (Un)Embeddings	
Generalization	**Embedding forgetting helps generalization to the unseen [2][3]**	
Interpretability	Un-cache the compute stored in embeddings leads to data graph reconstruction for both (under review)	

[1] CHEN ET AL 2021 RELATION PREDICTION AS AN AUXILIARY TRAINING OBJECTIVE FOR IMPROVING MULTI-RELATIONAL GRAPH REPRESENTATIONS.
[2] CHEN ET AL 2022 REFACTOR GNNS: REVISITING FACTORISATION-BASED MODELS FROM A MESSAGE-PASSING PERSPECTIVE
[3] CHEN ET AL 2023 IMPROVING LANGUAGE PLASTICITY VIA PRETRAINING WITH ACTIVE FORGETTING

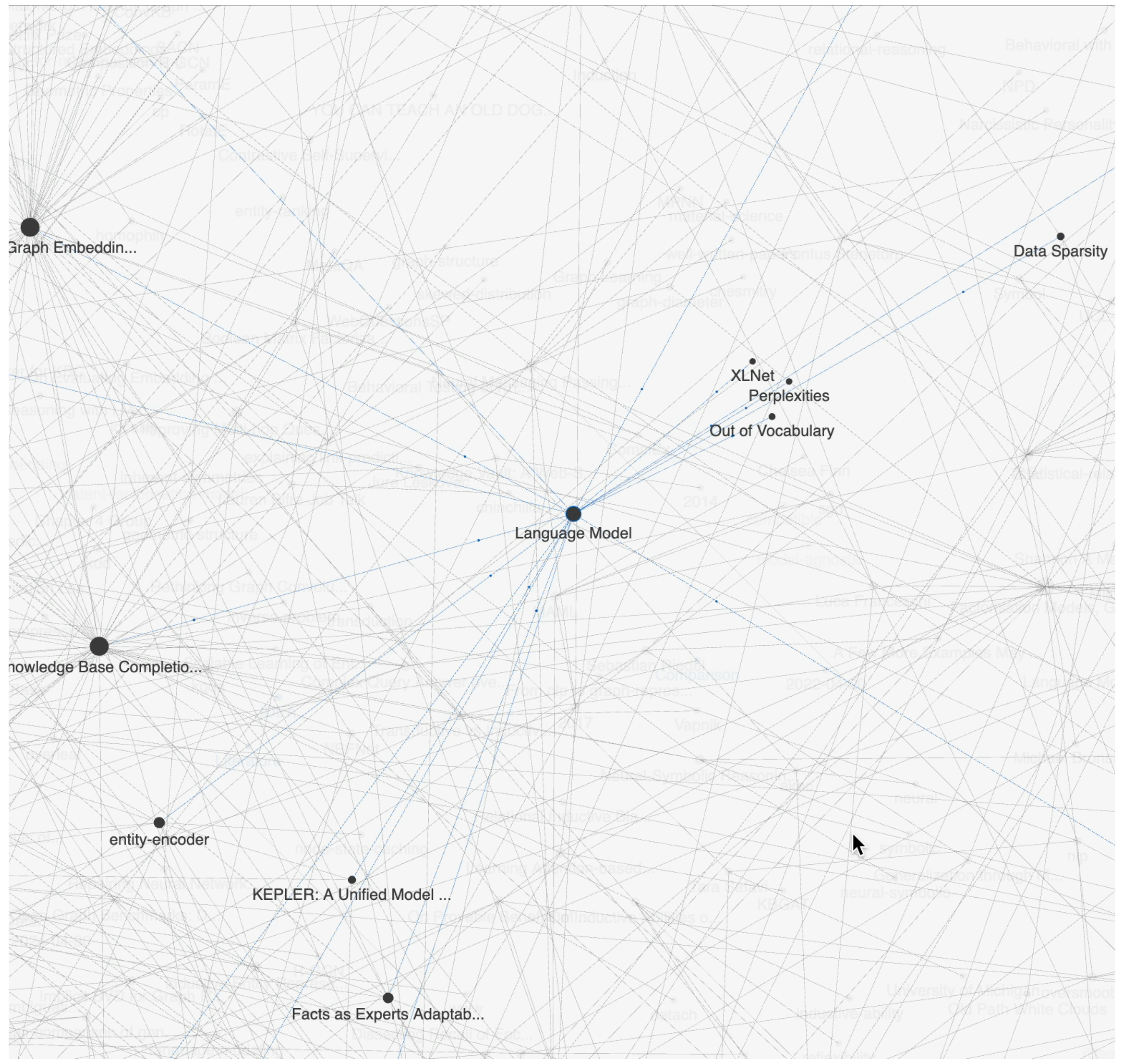
The Role of Embedding and How It Impacts Generalization (in short)

cat =>	1.2	-0.1	4.3	3.2
mat =>	0.4	2.5	-0.9	0.5
on =>	2.1	0.3	0.1	0.4

We propose the message-passage reframing of *symbol embeddings optimization*

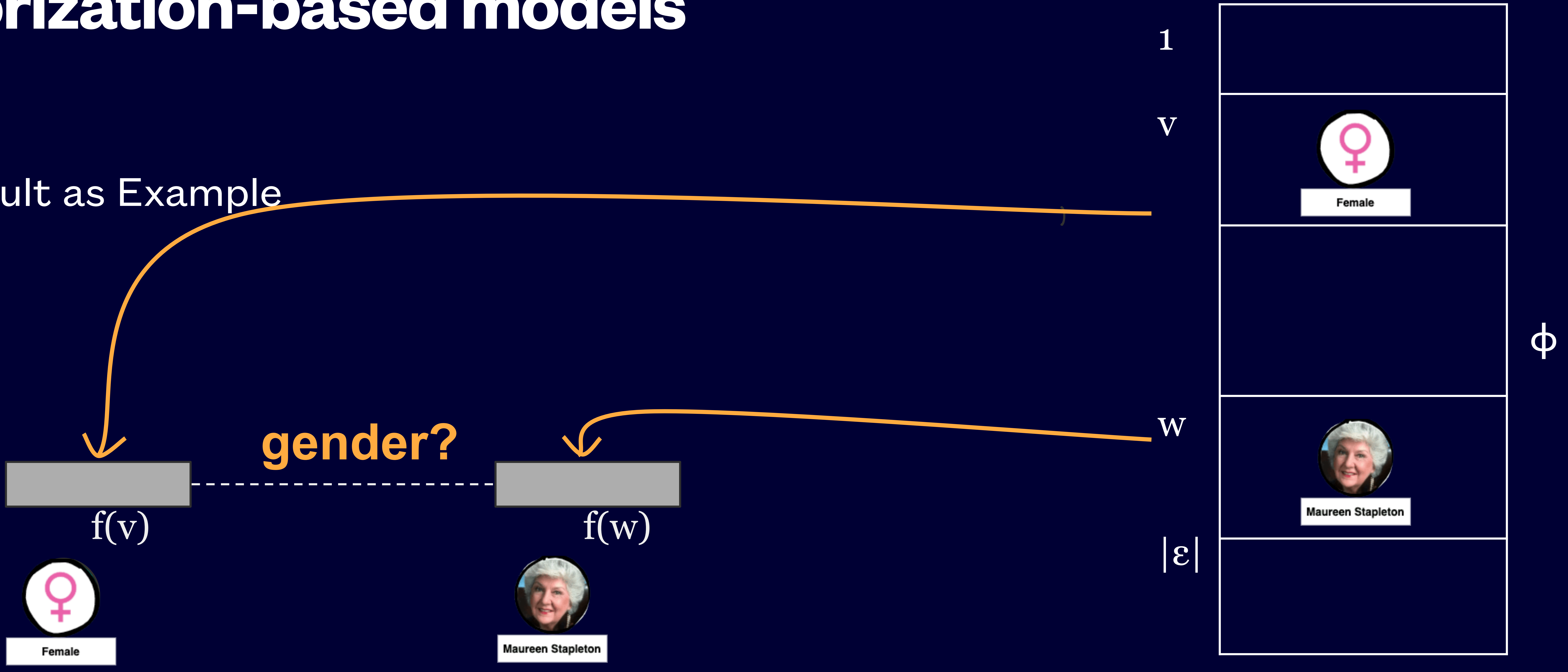
- symbol embeddings as memory which caches data traversal during training
- too much memory in old environments -> poor generalization in new environments

- So what?
  - *symbol embedding forgetting* helps generalization to the unseen
    - graphs with ReFactorGNN
    - languages with forgetting pretrained LMs
  - using GNN terminology:
    - “inductivise” transductive models



Embeddings for knowledge graph representation learning: factorization-based models

DistMult as Example



$$\Gamma_\theta(v, r, w) = \langle f_\phi(v), f_\phi(w), g_\psi(r) \rangle = \sum_{i=1}^{K} f_\phi(v)_i f_\phi(w)_i g_\psi(r)_i$$

[2] CHEN ET AL 2022 REFACTOR GNNS: REVISITING FACTORISATION-BASED MODELS FROM A MESSAGE-PASSING PERSPECTIVE NEURIPS 2022

L=2

$$\text{Message}(h^0[v], r, h^0[w]) = \begin{cases} h^0[w] \odot g(r) & \textit{if } (r, w) \in \mathcal{N}_{+}[v], \\ (1 - P_\theta(v|w, r))h^0[w] \odot g(r) & \textit{if } (r, w) \in \mathcal{N}_{-}^{1}[v]; \end{cases}$$

Implicit Message-Passing within FMs

Theorem 3.1 (Message passing in FMs). *The gradient descent operator (7) on the node embeddings of a DistMult model (4) with objective (3) and a multi-relational graph $(\mathcal{E}, \mathcal{T})$ induces a message-passing operator whose composing functions are:*

$$m^l[v,r,w] = \text{Message}(h^{l-1}[v], r, h^{l-1}[w]) = \begin{cases} h^{l-1}[w] \odot g(r) & \text{if } (r,w) \in \mathcal{N}_+[v], \\ (1 - P_\theta(v|w,r))h^{l-1}[w] \odot g(r) & \text{if } (r,w) \in \mathcal{N}_-[v]; \end{cases} \tag{8}$$

$$z^l[v] = \text{Aggregate}(\{m^l[v,r,w] \,:\, (r,w) \in \mathcal{N}[v]\}) = \sum_{(r,w)\in\mathcal{N}[v]} m^l[v,r,w]; \tag{9}$$

$$h^l[v] = \text{Update}(h^{l-1}[v], z^{l-1}[v]) = h^{l-1}[v] + \alpha z^{l-1}[v] - \beta n^{l-1}[v], \tag{10}$$

where, defining the sets of triples $\mathcal{T}^{+v} = \{(s,r,w) \in \mathcal{T} \,:\, s = v \wedge w \neq v\}$ and $\mathcal{T}^{-v} = \{(s,r,w) \in \mathcal{T} \,:\, s \neq v \wedge w \neq v\}$, $P_{\mathcal{T}^{+v}}$ and $P_{\mathcal{T}^{-v}}$ as their associated empirical probability distributions,

$$n[v] = \frac{|\mathcal{T}^{+v}|}{|\mathcal{T}|}\mathbb{E}_{P_{\mathcal{T}^{+v}}}\mathbb{E}_{u\sim P_\theta(\cdot|v,r)}\frac{\partial\Gamma(v,r,u)}{\partial h[v]} + \frac{|\mathcal{T}^{-v}|}{|\mathcal{T}|}\mathbb{E}_{P_{\mathcal{T}^{-v}}}P_\theta(v|s,r)\frac{\partial\Gamma(s,r,v)}{\partial h[v]}. \tag{11}$$

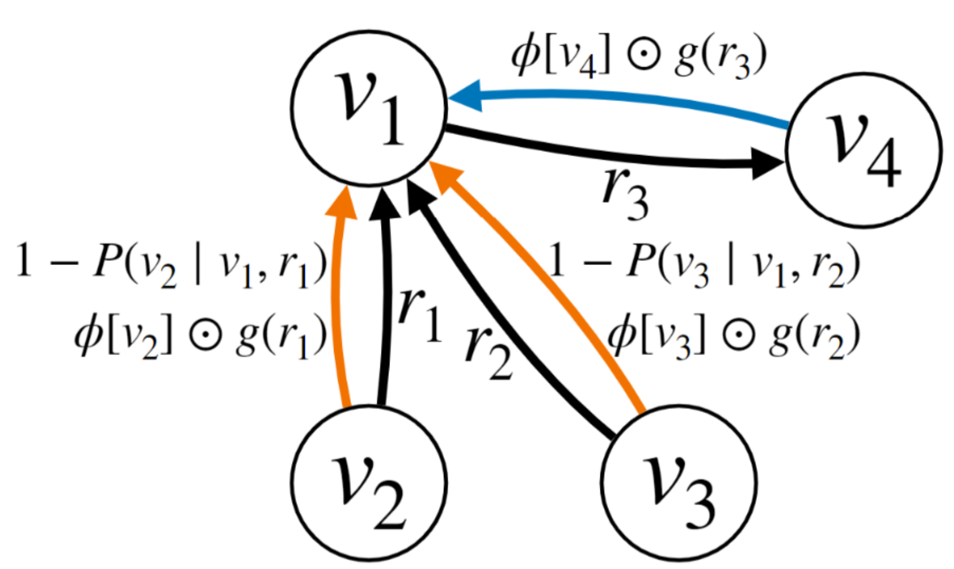
Extensions to other score functions: see lemma A.1 in the paper

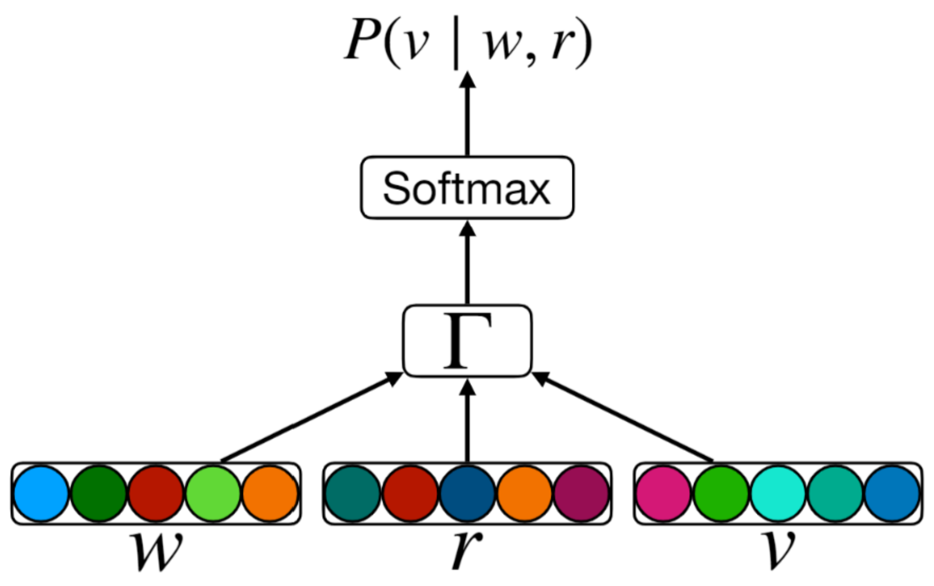
Implicit Message-Passing within FMs (layman summary)

Treat the node embedding layer as a *historical memory* of node states

One *gradient descent step over the embeddings* induces one message-passing layer

- in-coming and out-going neighbourhood
- relation-aware
- global normaliser



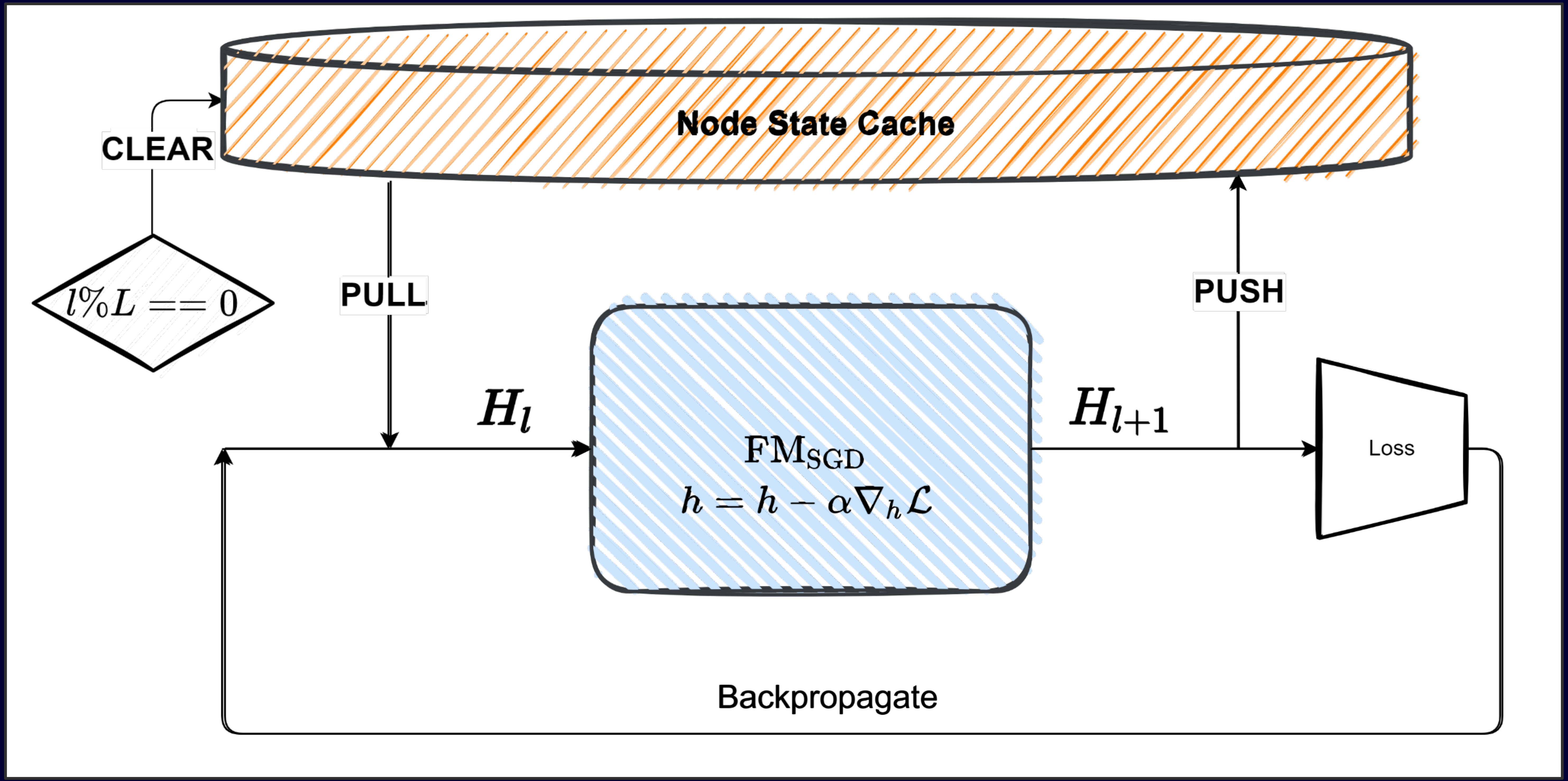


Such message-passing over data graph is "cached" into embeddings via accumulating the update vector into the history.

Tensor factorization
=
Graph neural networks

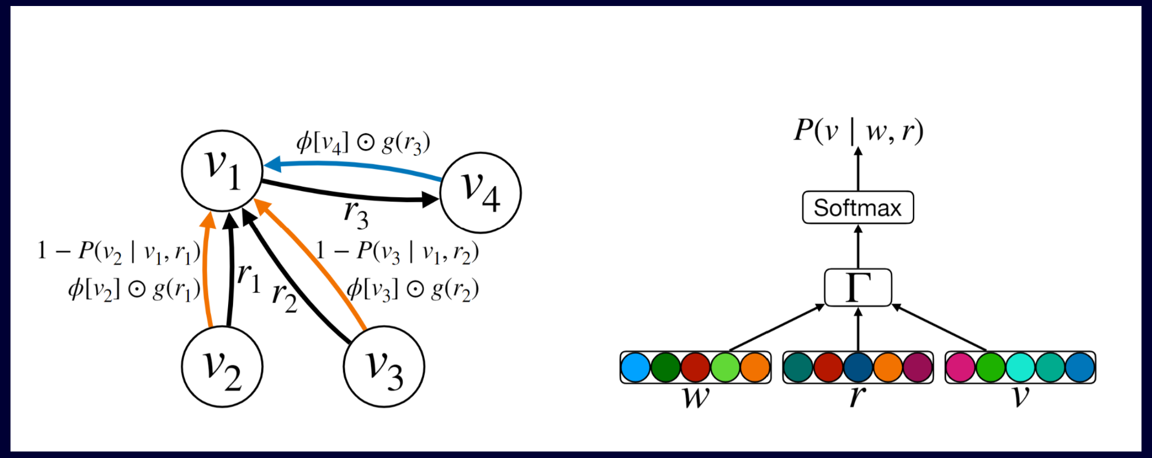
L=2

the message-passing rounds (some visualization of memory cleanup)

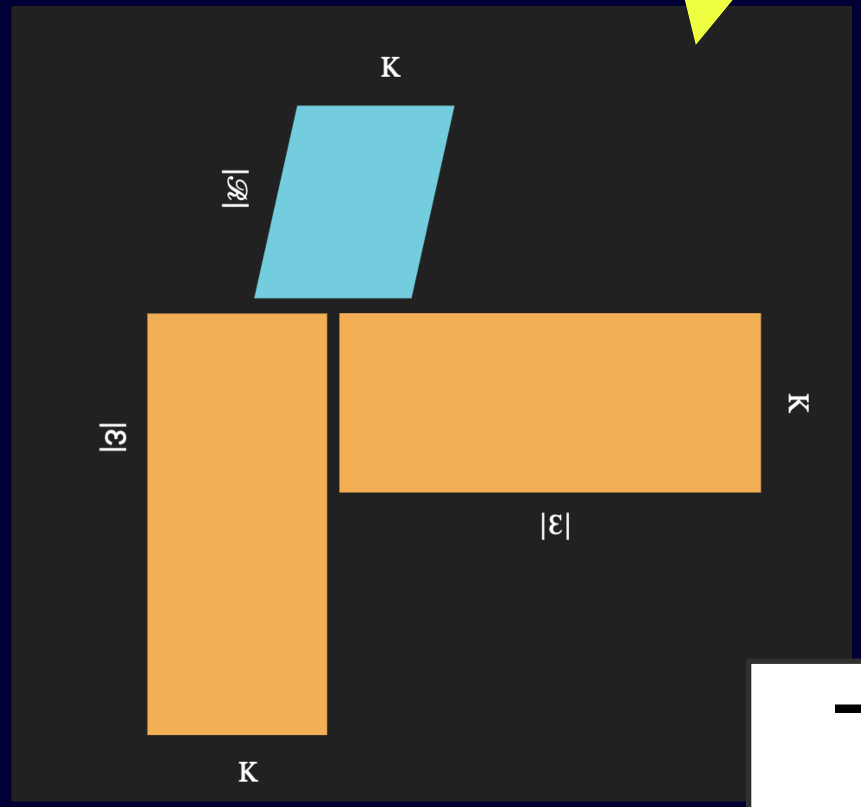


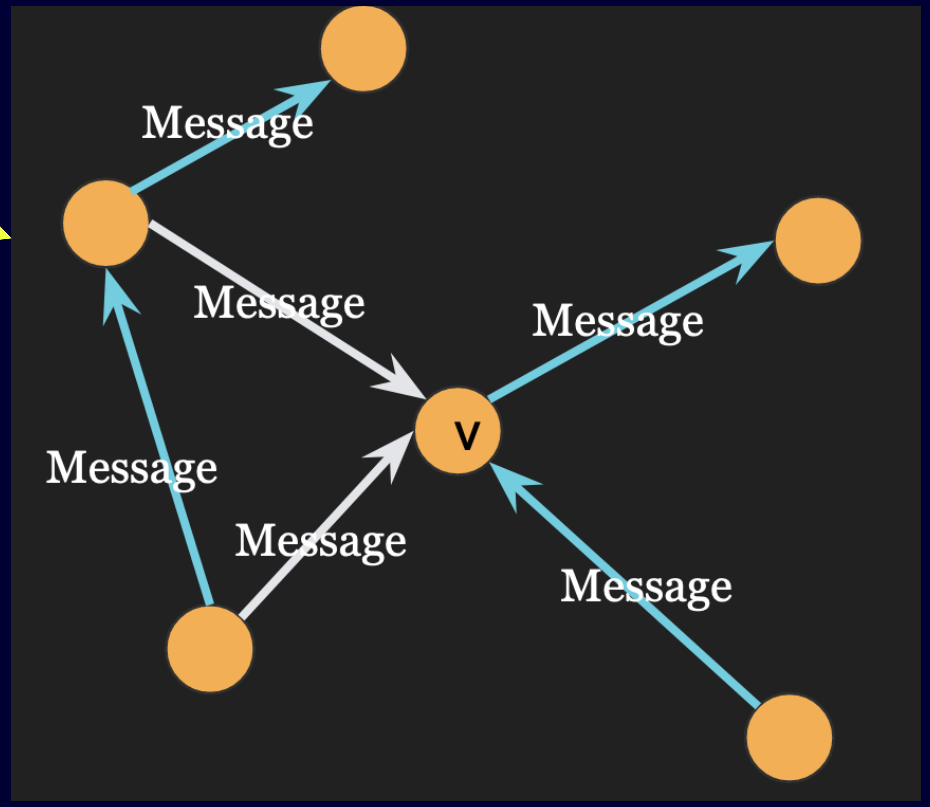
Controlling the rounds of message-passing compute

K = ∞



K = 6





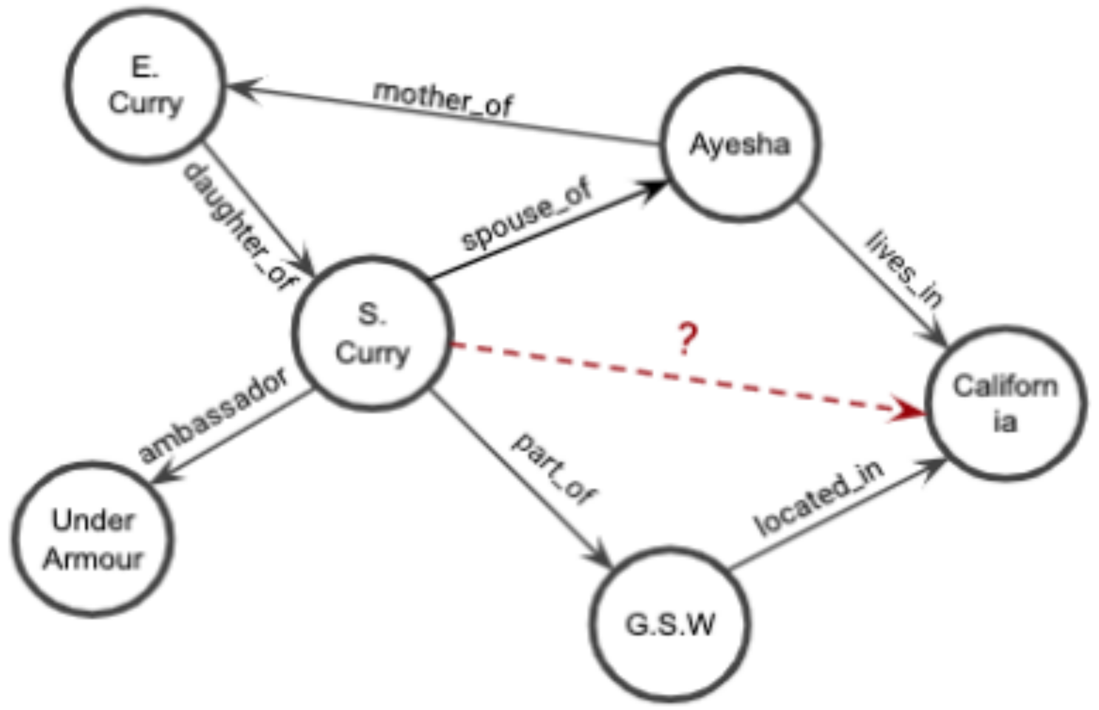
- Equivalently, “Inductivise” factorization models by truncating infinite to K message-passing
- Every reasoning is forced to use fixed number of hops neighboring information rather than memorize everything for reasoning

Implication

- Factorization methods are known to be transductive despite their impressive performance on link prediction
- Now we can make them inductive.
- Generalize to unseen nodes!

a. Training graph

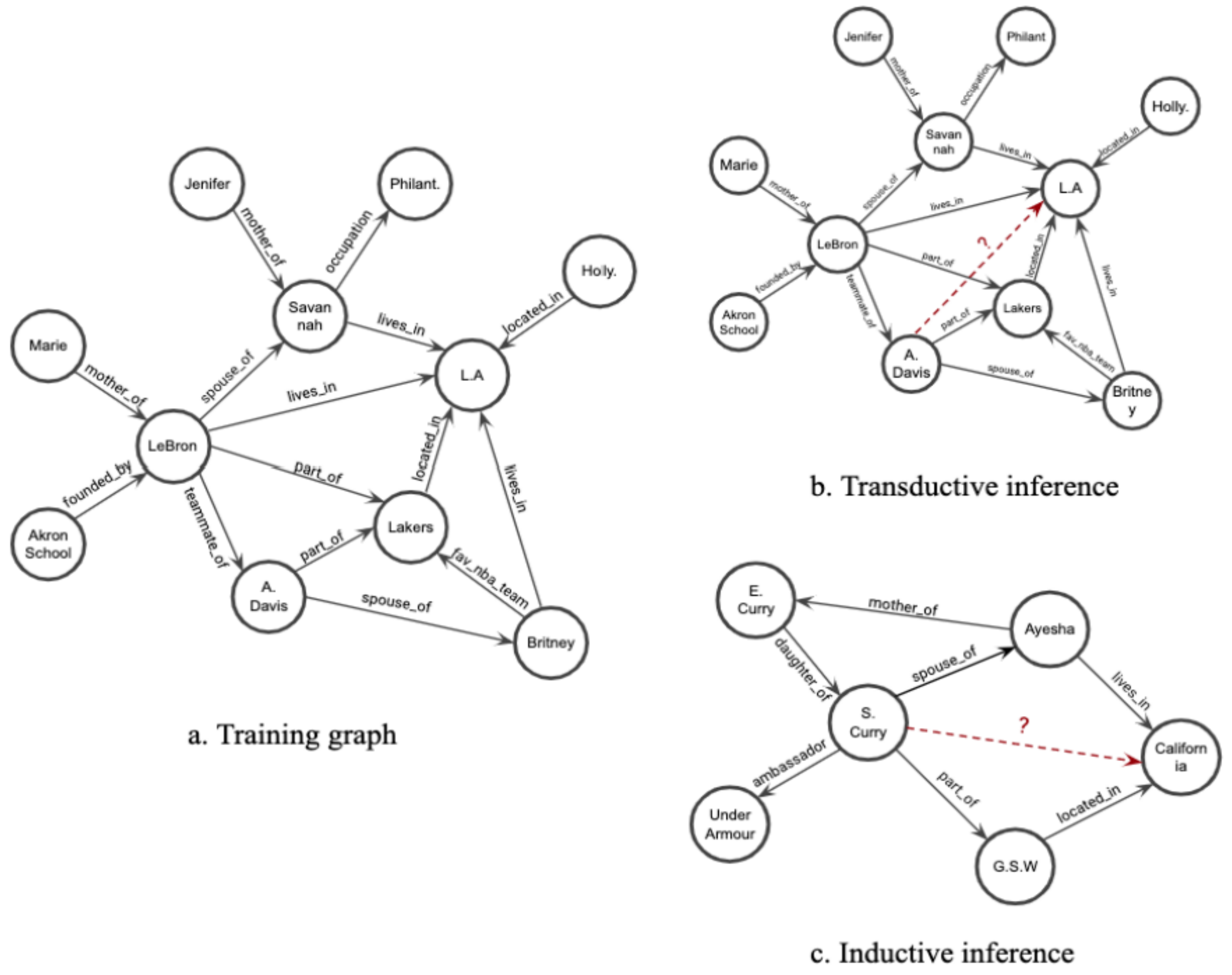
b. Transductive inference

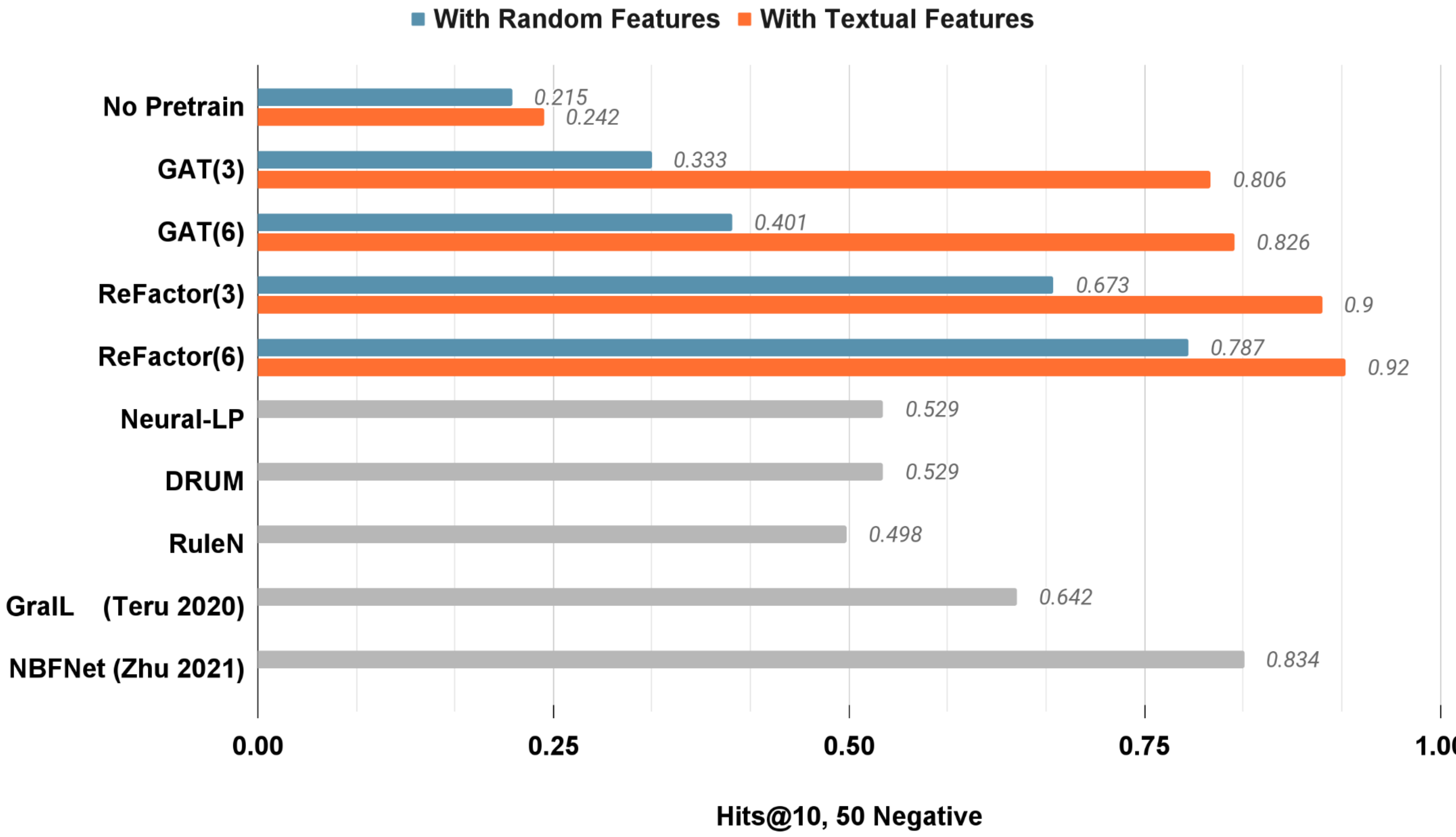


c. Inductive inference

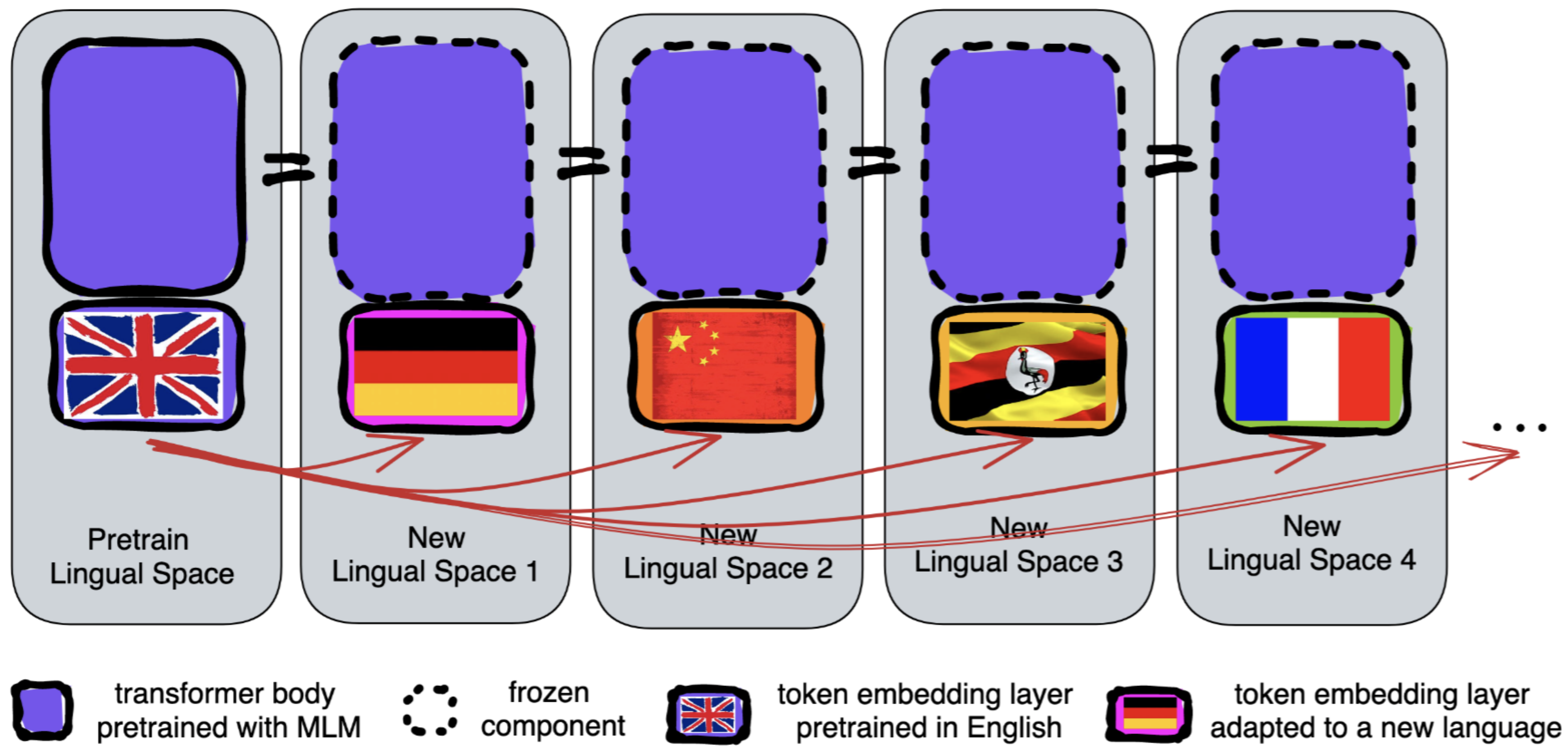
Results

- Generalize to unseen nodes!





Moving to languages



LMs struggle with generalization with under-represented languages.

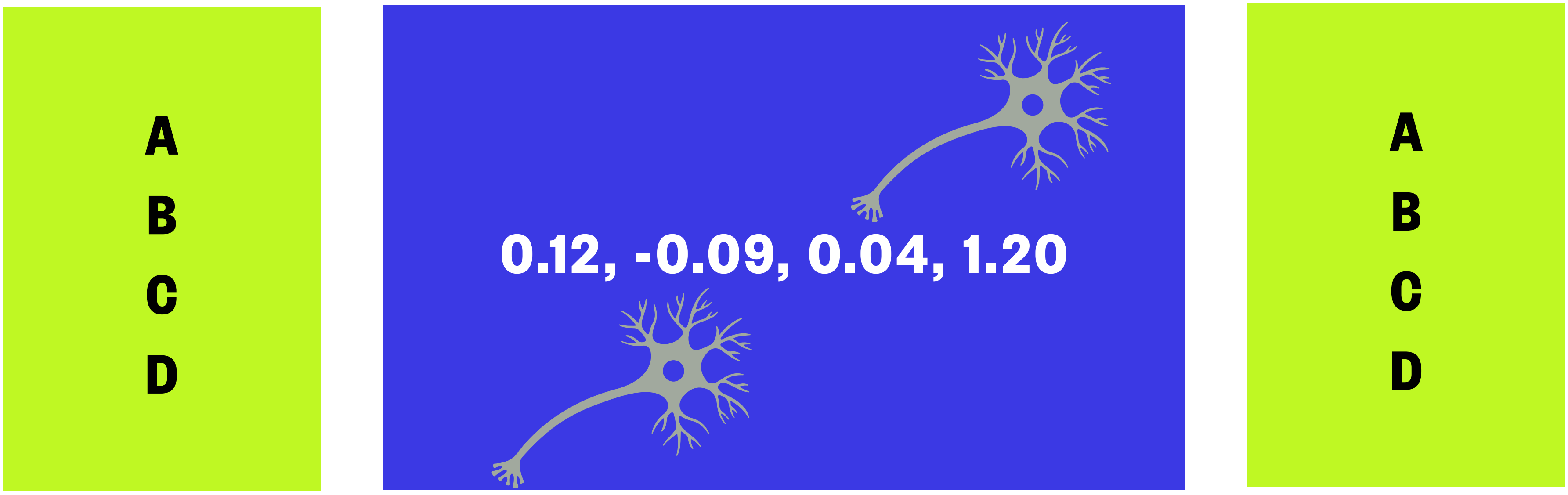
Updating them to new languages can be a headache.

Ideally, we want to avoid retraining.

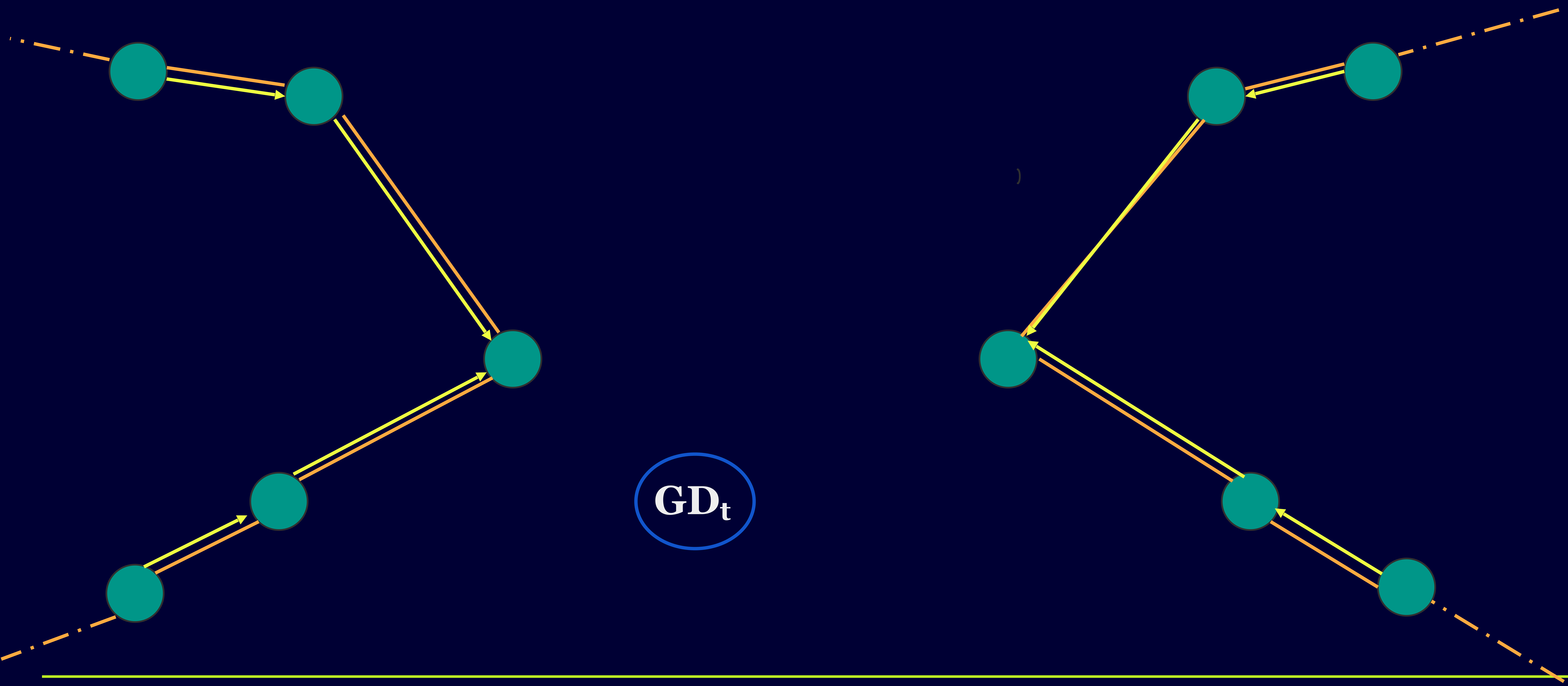
[3] CHEN ET AL 2023 IMPROVING LANGUAGE PLASTICITY VIA PRETRAINING WITH ACTIVE FORGETTING

Generalising to languages

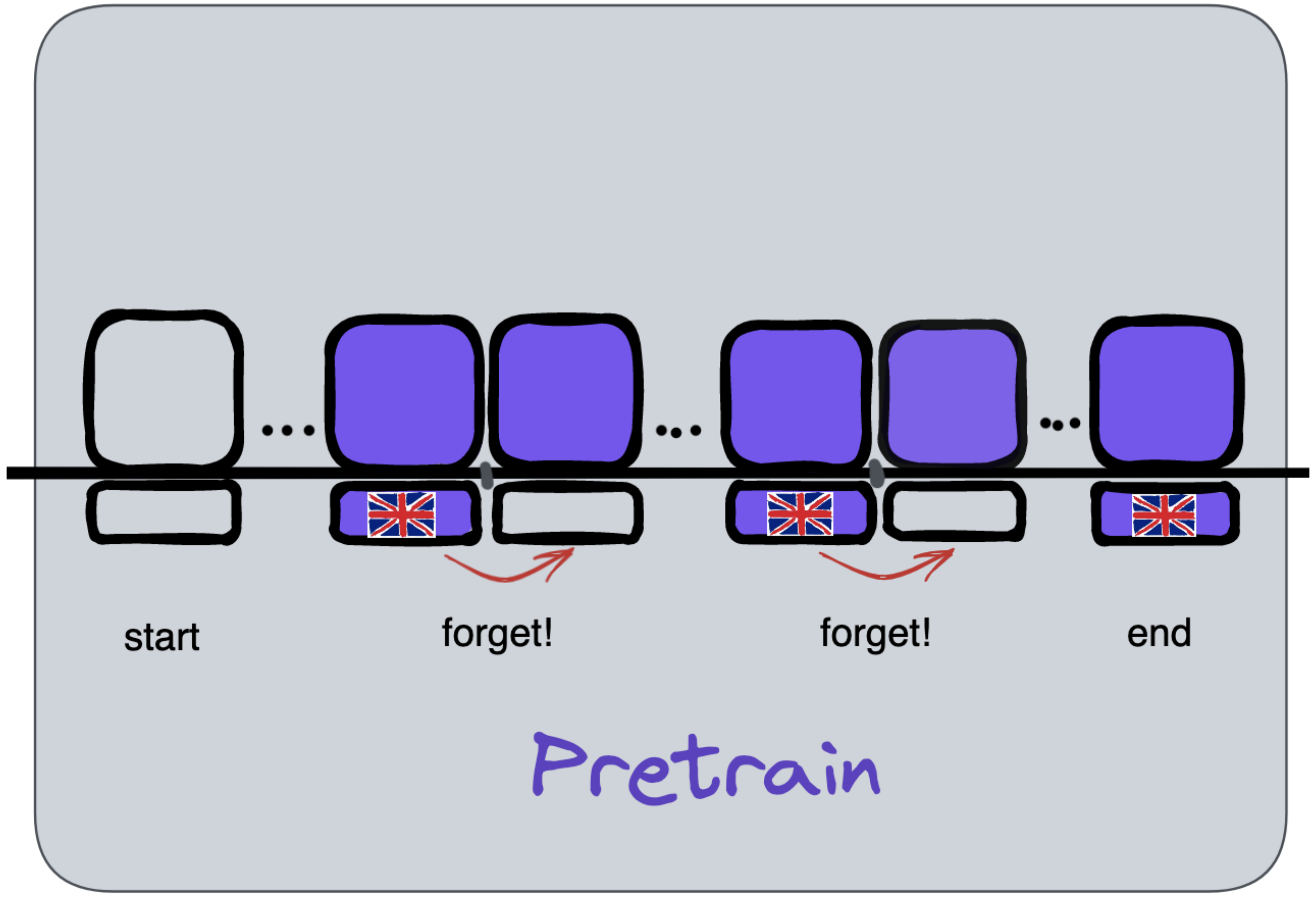
Every transformer-based language model begins with embeddings and end with (un)-embeddings.



[3] CHEN ET AL 2023 IMPROVING LANGUAGE PLASTICITY VIA PRETRAINING WITH ACTIVE FORGETTING


GD_t

Pretraining with Active Forgetting

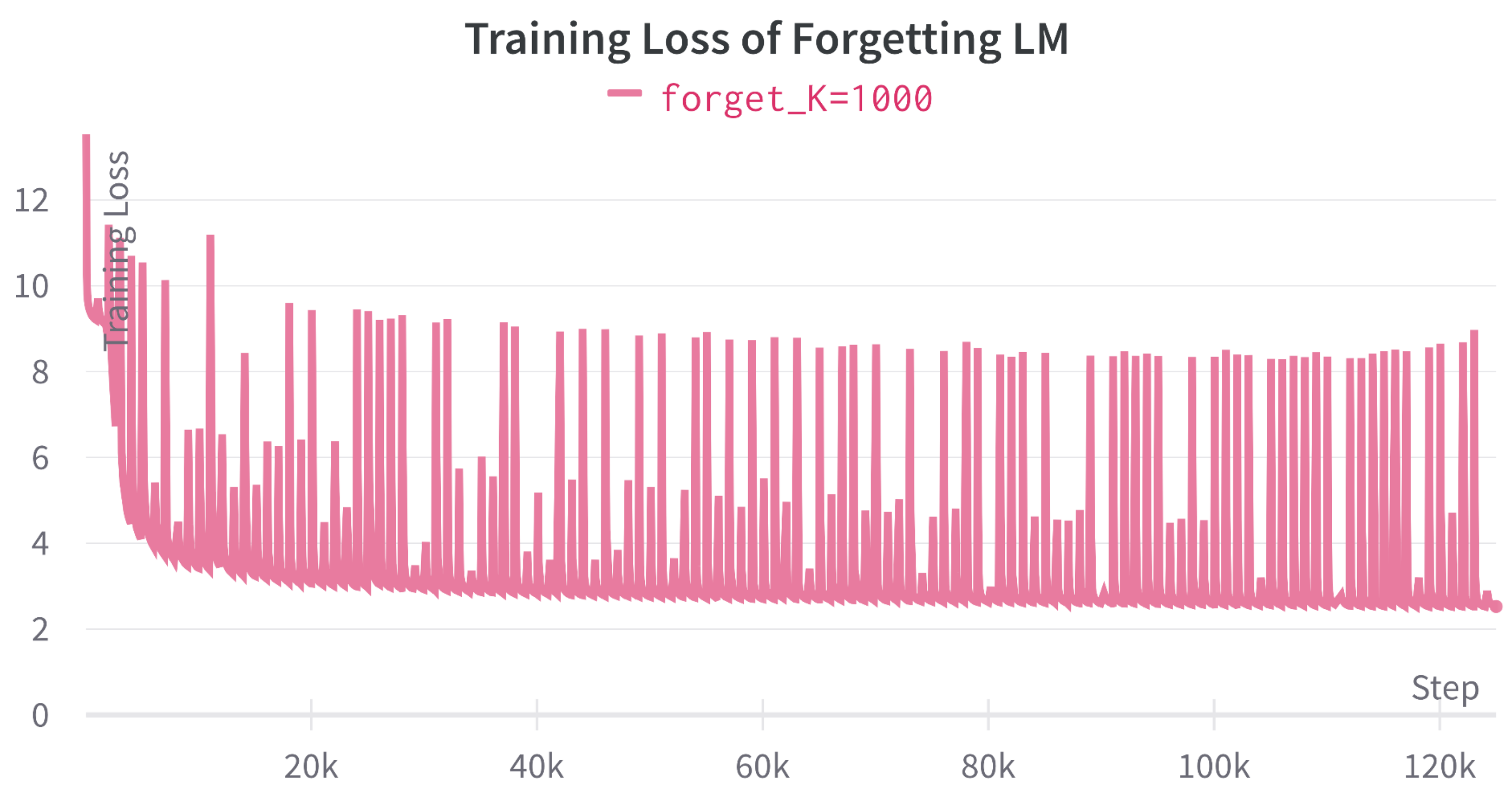


A cheap way of meta-learning LMs

- Simulating multiple language changes without actually crafting the data in new language
- Exposing the body to various embedding reinitialisation
- Encourage the body to encode more general knowledge instead of “shortcut” knowledge that is tied to certain embedding initialisation values

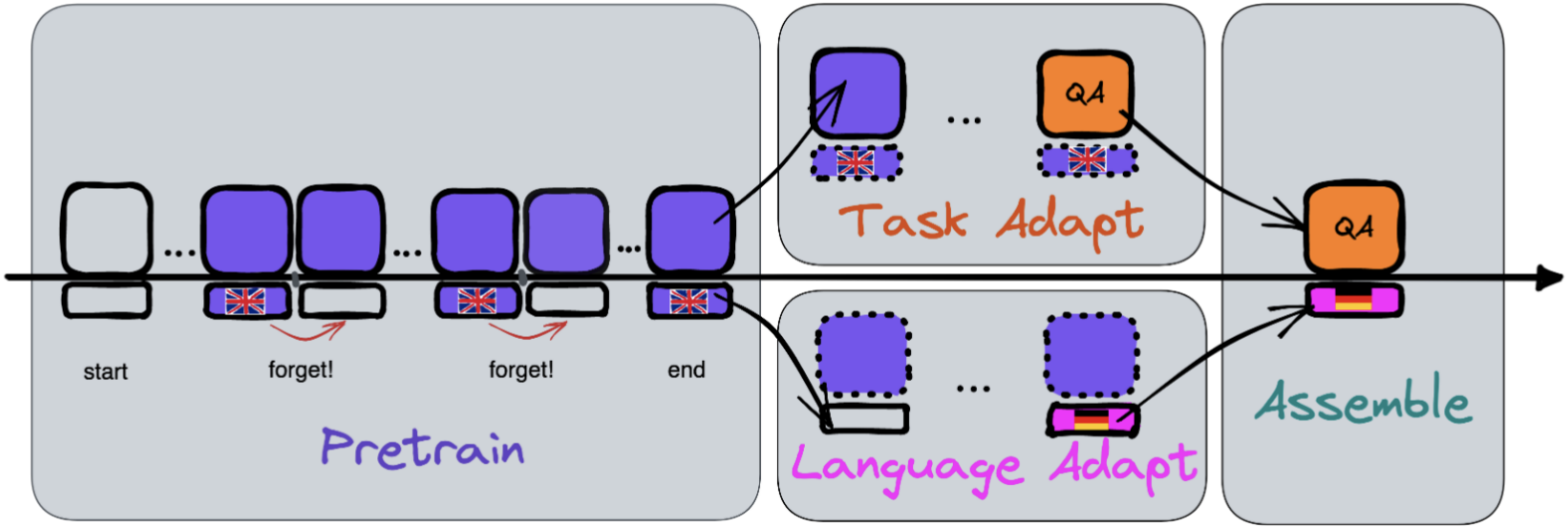
Pretraining with Active Forgetting

episodic learning curve, “spikes” when resetting



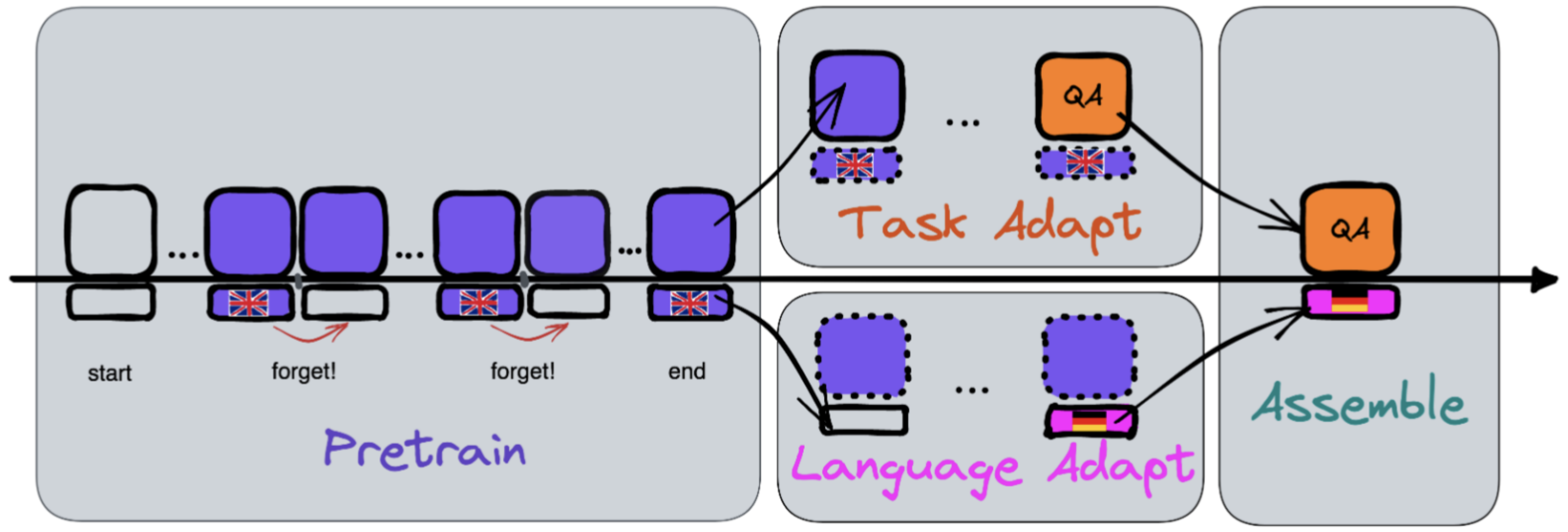
Results

- Generalising to unseen languages. Unsupervised zero-shot cross-lingual transfer!



Results

- Generalising to unseen languages with less data and compute!



On average

+21.2% on XNLI,

+33.8% on MLQA

+60.9% on XQuAD

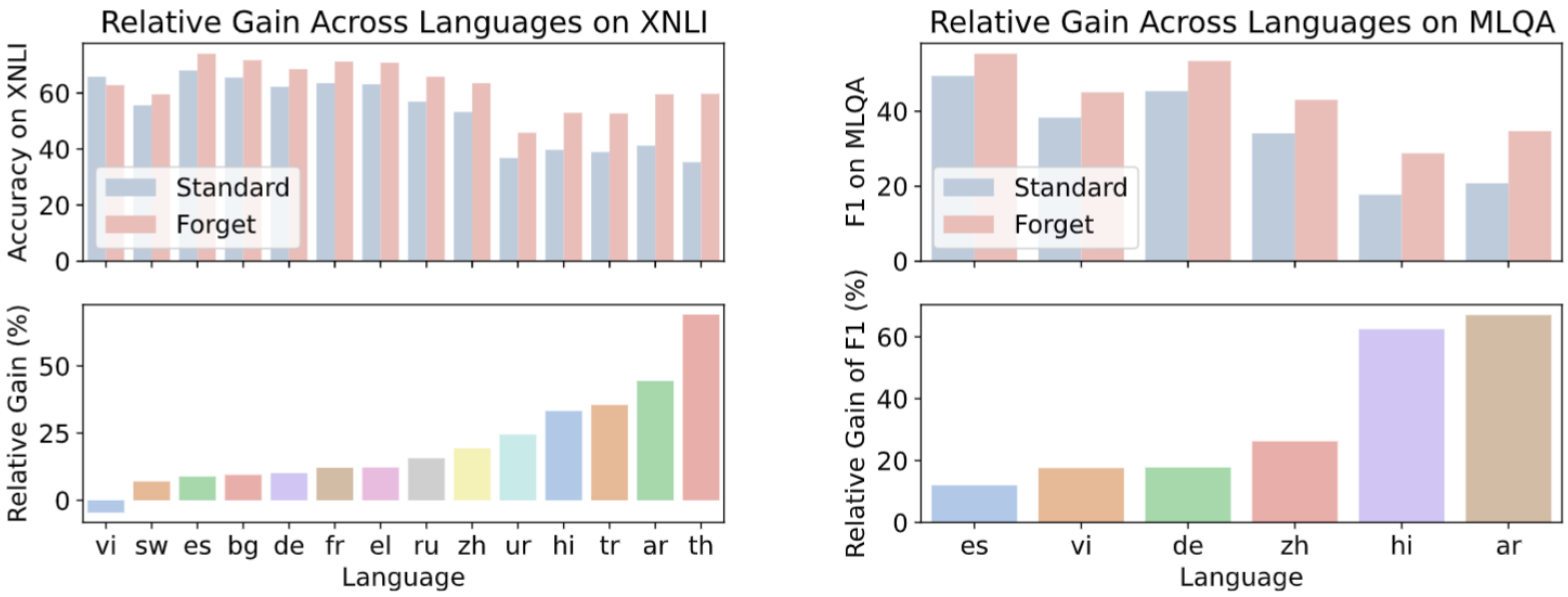
	XNLI (accuracy)	MLQA (F1)	XQUAD (F1)
Standard PLM	53.3	34.3	36.1
Forgetting PLM	**62.7**	**43.4**	**49.0**

+60.9%

Forgetting brings an average gain of 60.9% on XQuAD when generalizing to unseen lang

So what?

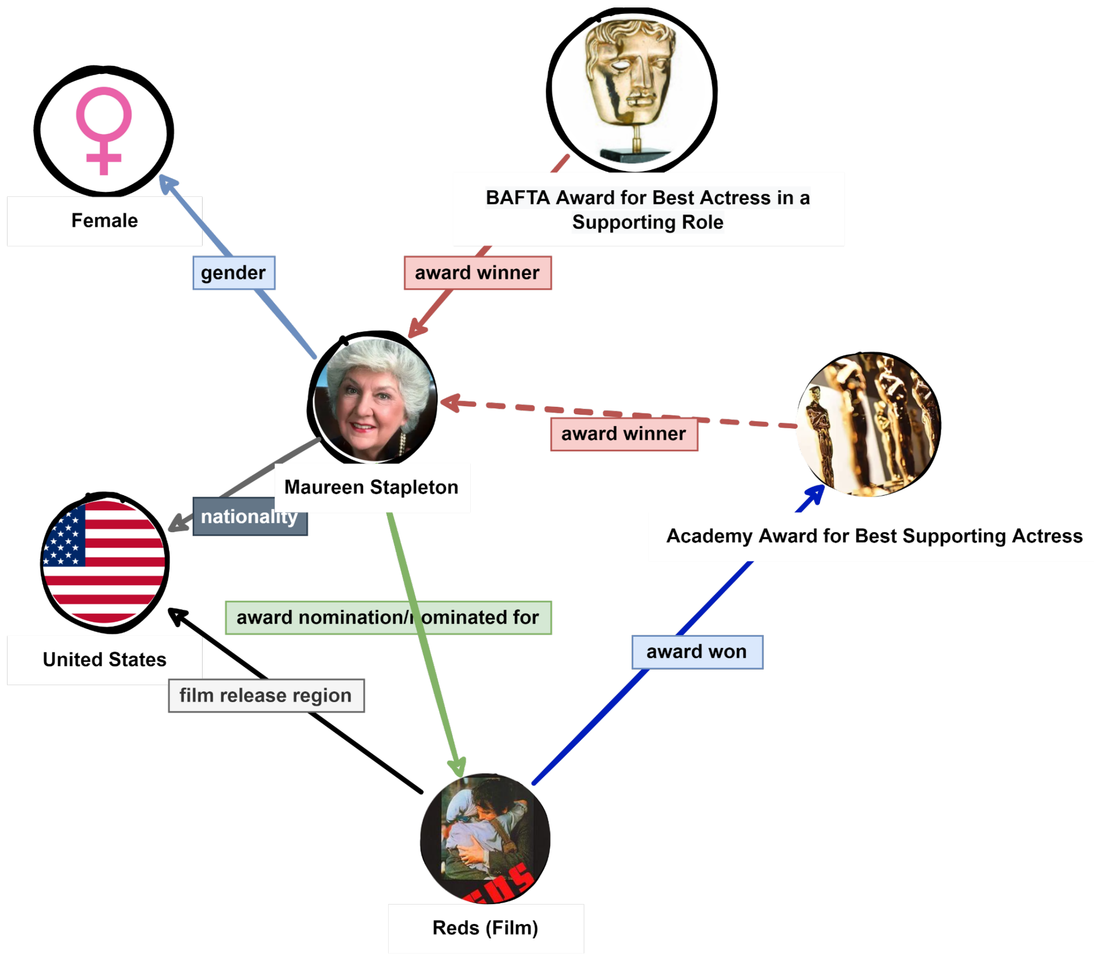
- Help low-resources languages!



Scaling increases model capacity while forgetting improves model plasticity -> easy to update to *new* XYZ

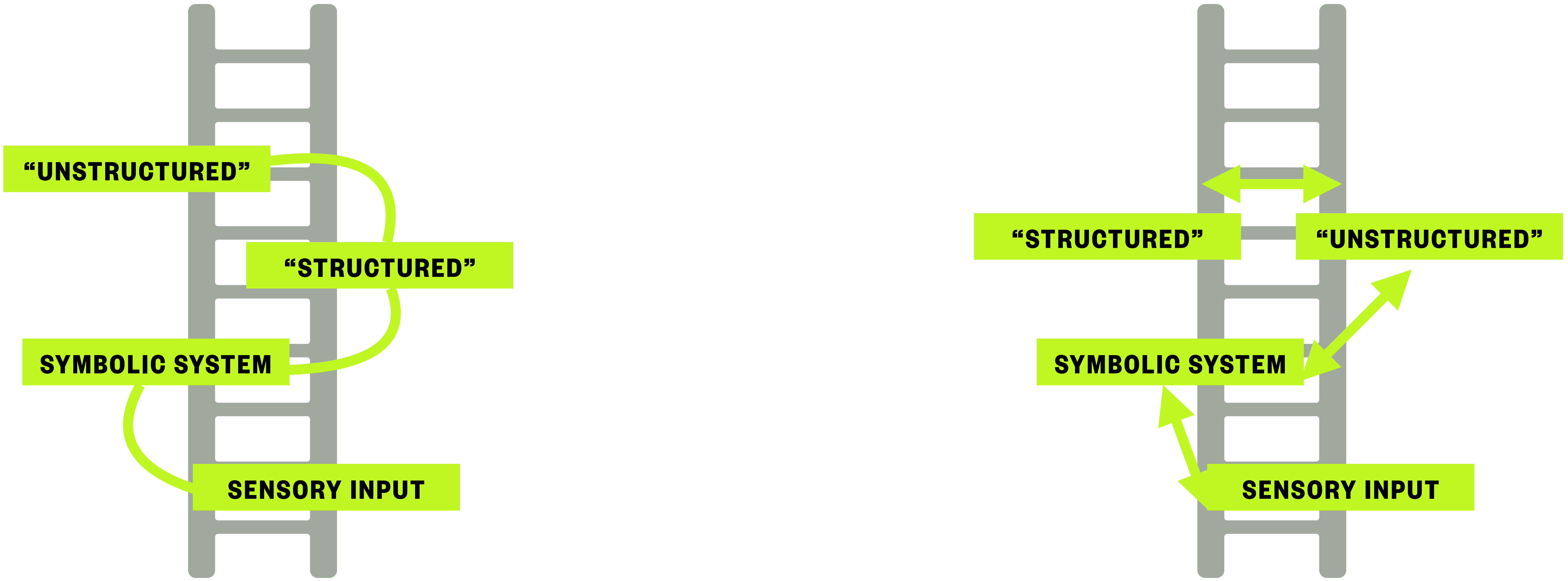
Research Vision

Structured + Unstructured



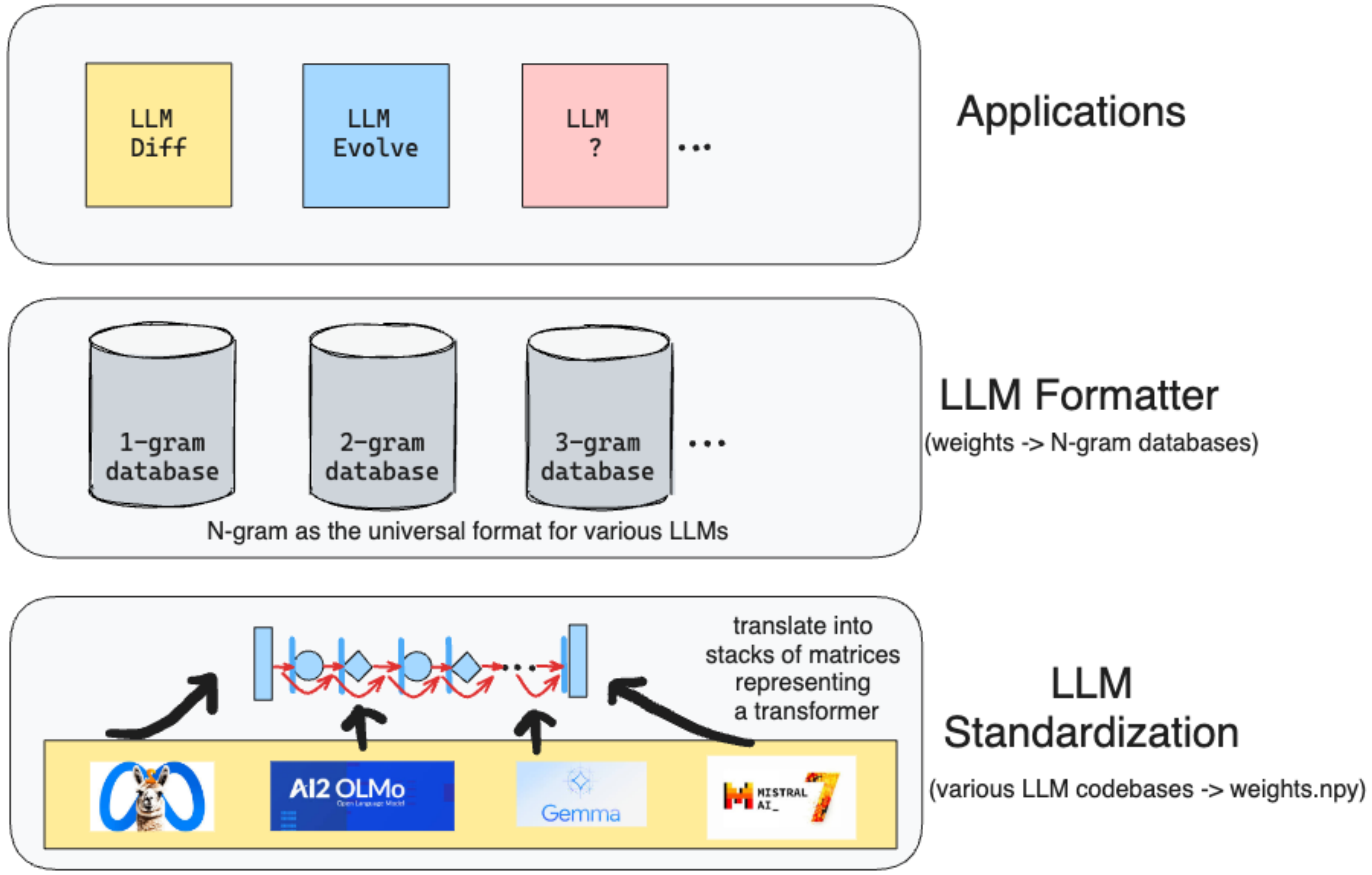
	Pros	Cons
Structured	controllable (easy to update/ edit/remove), interpretable, reasoning, planning	construction cost, missing entries
Unstructured	generative! (can create answers for any questions), ingest huge data	hard to control (hallucination/ toxicity), expensive

Towards more controllable AI via channelling structured and unstructured learning paradigms



Preliminary exploration (under review)

- We can identify the n-gram structures via *decomposing* model weights
  - Re-formatting LLMs into a universal interface of n-grams



Preliminary exploration

- We can identify the n-gram structures via *decomposing* model weights
  - Data-free, weights-only LLM pretraining examination

Table 1: Bi-gram evolution across pretraining steps for OLMo 7B. Each column represents a distinct step, while each row corresponds to a different rank. The table entries are the bi-grams at each step for each rank. The number of tokens seen in association with the pretraining steps is also annotated. The model gradually picks up meaningful bi-grams while starts from senseless bi-grams.

Rank	0K [**#steps**] 0B [**#tokens**]	100K 442B	200K 885B	300K 1327B	400K 1769B	555K 2455B
0	immortal	’ s	at least	&amp	&amp	&amp
1	ICUirling	at least	’ s	at least	its own	its own
2	ords architect	its own	&amp	its own	their own	their own
3	yaml Adam	okerly	your own	your own	at least	his own
4	231 next	VENT thanks	its own	their own	your own	make sure
5	clonal 条	iums	iums	more than	his own	your own
6	Charg@{	you’re	you’re	can’t	2nd	2nd
7	avoir careless	Everything v	2nd	his own	more than	at least
8	HOLD worsening	erna already	you guys	2nd	make sure	more than
9	Horse dismant	’my	more than	make sure	can’t	iums

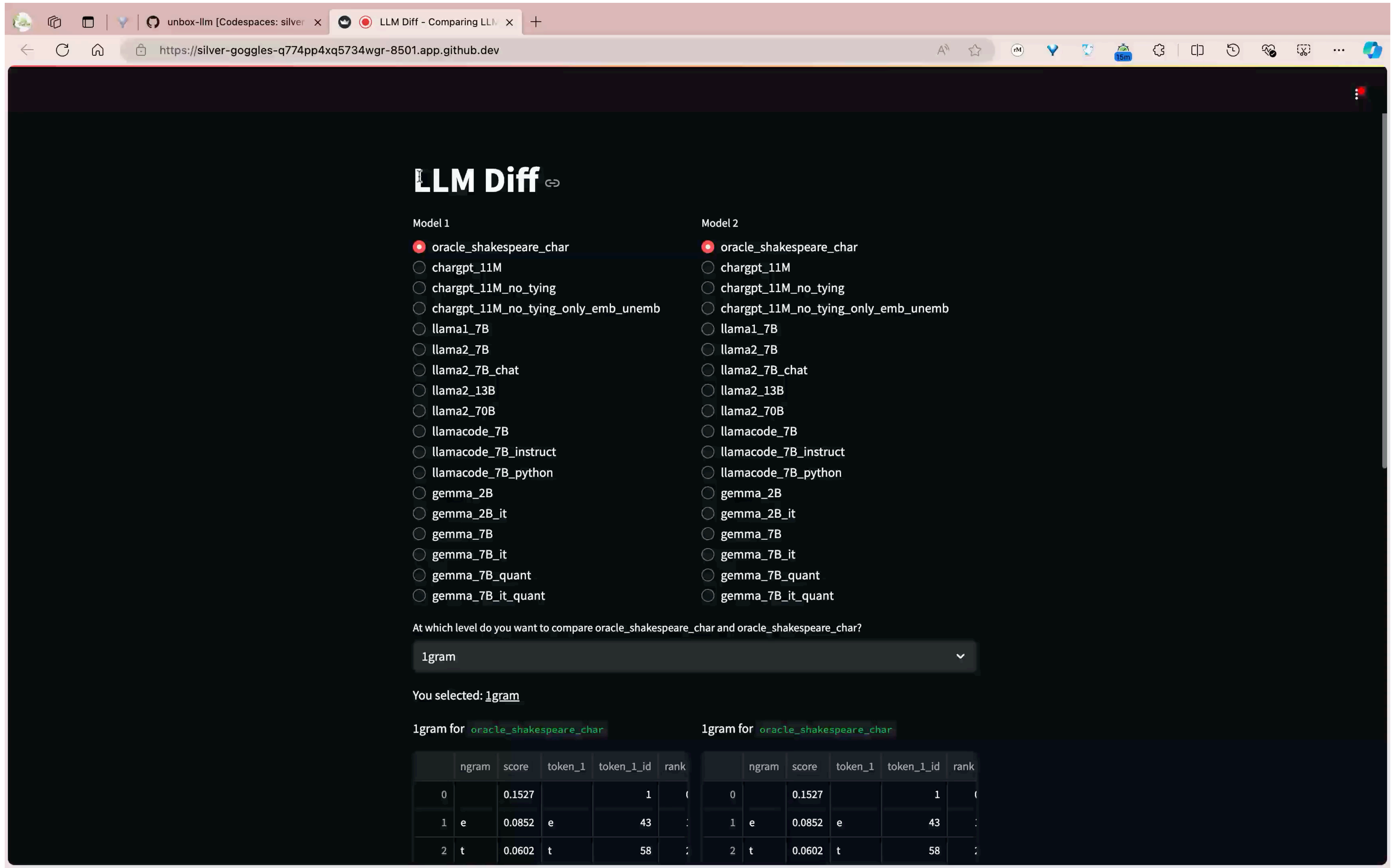
-

Preliminary exploration

- We can identify the n-gram structures via *decomposing* model weights
  - Domain-specific LLMs will reflect their magic data mixture and point us where to update.
  -

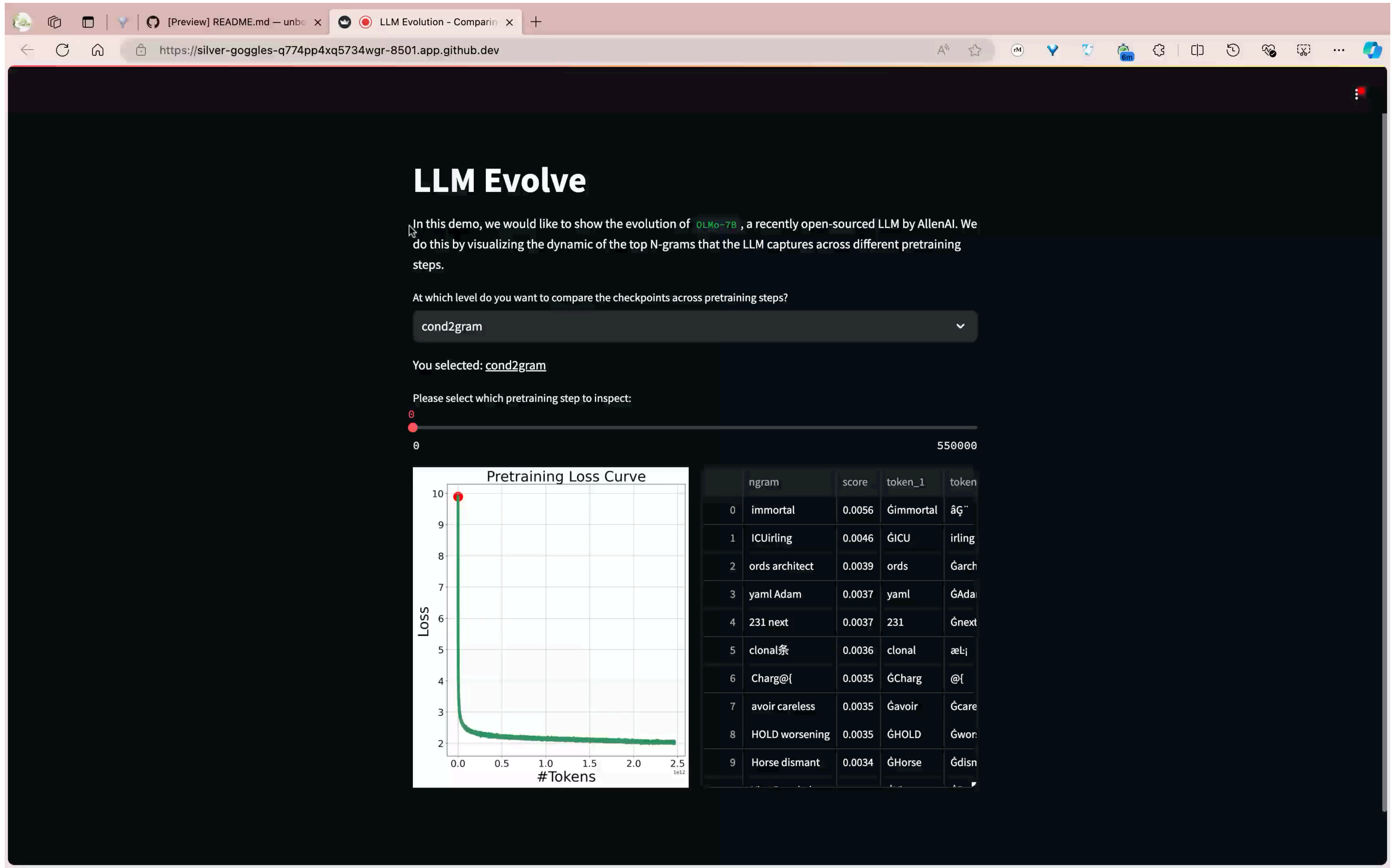
Rank	LLAMA2-7B	CodeLLAMA-7B	CodeLLAMA-Python-7B
0	(_more, _than)	(_like, wise)	(_like, wise)
50	(_Now, here)	(_just, ification)	(_Like, wise)
100	(_system, atically)	(_in, _case)	(_all, udes)
150	(_all, erg)	(_get, ters)	(_no, isy)
200	(_on, ions)	(któber, s)	(output, ted)
300	(_other, world)	(_all, ud)	(Object, ive)
350	(_Just, ified)	(gebiet, s)	(_as, cii)
400	(_trust, ees)	(_Protest, s)	(_can, nab)
450	(_at, he)	(_deploy, ment)	(_transport, ation)
500	(_book, mark)	(Class, room)	(Tag, ging)
550	(_from, 而)	(_access, ory)	(_personal, ized)
600	(_WHEN, ever)	(_In, variant)	(_excess, ive)
650	(_where, about)	(_I, _am)	(_Add, itional)
700	(ag, ged)	(add, itionally)	(_**, kwargs)
750	(_he, he)	(_invalid, ate)	(name, plates)
800	(_all, anto)	(div, ision)	(_select, ive)
850	(_Tom, orrow)	(_process, ors)	(_Assert, ions)
900	(_for, ays)	(_Program, me)	(blog, ger)
950	(_Bach, elor)	(_set, up)	(_can, cellation)

LLM Diff

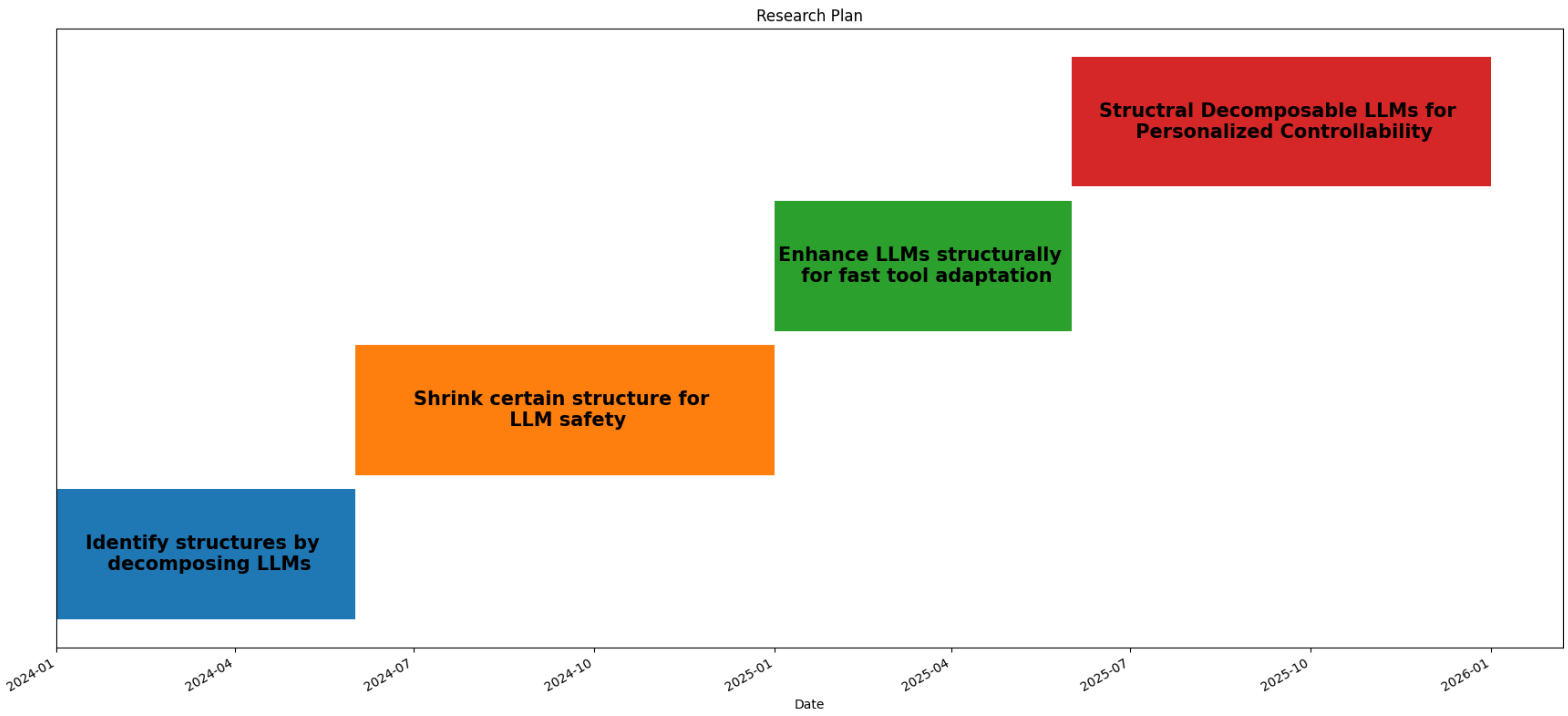


unbox-llm [Codespaces: silver
LLM Diff - Comparing LL
https://silver-goggles-q774pp4xq5734wgr-8501.app.github.dev
LLM Diff
Model 1
oracle_shakespeare_char
chargpt_11M
chargpt_11M_no_tying
chargpt_11M_no_tying_only_emb_unemb
llama1_7B
llama2_7B
llama2_7B_chat
llama2_13B
llama2_70B
llamacode_7B
llamacode_7B_instruct
llamacode_7B_python
gemma_2B
gemma_2B_it
gemma_7B
gemma_7B_it
gemma_7B_quant
gemma_7B_it_quant
Model 2
oracle_shakespeare_char
chargpt_11M
chargpt_11M_no_tying
chargpt_11M_no_tying_only_emb_unemb
llama1_7B
llama2_7B
llama2_7B_chat
llama2_13B
llama2_70B
llamacode_7B
llamacode_7B_instruct
llamacode_7B_python
gemma_2B
gemma_2B_it
gemma_7B
gemma_7B_it
gemma_7B_quant
gemma_7B_it_quant
At which level do you want to compare oracle_shakespeare_char and oracle_shakespeare_char?
1gram
You selected: 1gram
1gram for oracle_shakespeare_char
ngram score token_1 token_1_id rank
0 0.1527 1
1 e 0.0852 e 43
2 t 0.0602 t 58
1gram for oracle_shakespeare_char
ngram score token_1 token_1_id rank
0 0.1527 1
1 e 0.0852 e 43
2 t 0.0602 t 58

LLM Evolve


[Preview] README.md — unb
LLM Evolution - Comparin
https://silver-goggles-q774pp4xq5734wgr-8501.app.github.dev
LLM Evolve
In this demo, we would like to show the evolution of OLMo-7B, a recently open-sourced LLM by AllenAI. We do this by visualizing the dynamic of the top N-grams that the LLM captures across different pretraining steps.
At which level do you want to compare the checkpoints across pretraining steps?
cond2gram
You selected: cond2gram
Please select which pretraining step to inspect:
0
550000
Pretraining Loss Curve
Loss
#Tokens
ngram
score
token_1
token
immortal
0.0056
Ġimmortal
ICUirling
0.0046
ĠICU
irling
ords architect
0.0039
ords
yaml Adam
0.0037
yaml
231 next
0.0037
231
clonal
0.0036
clonal
Charg@(
0.0035
ĠCharg
avoir careless
0.0035
Ġavoir
HOLD worsening
0.0035
ĠHOLD
Horse dismant
0.0034
ĠHorse

Towards more controllable AI



Q & A

Thank you