DUAL PROCESS LEARNING: CONTROLLING USE OF IN-CONTEXT VS. IN-WEIGHTS STRATEGIES WITH WEIGHT FORGETTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models have the ability to perform in-context learning (ICL), allowing them to flexibly adapt their behavior based on context. This contrasts with in-weights learning (IWL), where memorized information is encoded in model parameters from iterated observations of the data (e.g., common sayings). An ideal model should be able to maintain both of these abilities. Despite their apparent ability to learn in-context, language models are known to struggle when faced with unseen or rarely seen tokens (Land & Bartolo, 2024). Hence, we study **structural in-context learning**, which we define as the ability of a model to execute in-context learning on arbitrary novel tokens – so called because the model must generalize on the basis of e.g. sentence structure or task structure, rather than content encoded in token embeddings. We study structural in-context algorithms on both synthetic and natural tasks using both toy models and MultiBERT models (Sellam et al., 2021). We find that structural ICL appears before quickly disappearing early in LM pretraining. While it has been shown that ICL can diminish during training (Singh et al., 2023), we find that prior work does not account for structural ICL. Building on Chen et al. (2024)'s active forgetting method used to help models learn new languages, we introduce a pretraining method that can modulate the preference for true structural ICL and IWL. Importantly, this allows us to induce a *dual process strategy* where in-context and in-weights solutions coexist within a single model. [1]

1 INTRODUCTION

A distinguishing trait of transformer language models (LMs) is their ability to perform 'in-context' learning (ICL) (Brown et al., 2020; Dong et al., 2023; Garg et al., 2023) – the ability to use context at inference time to adjust model behavior, without weight updates, to generalize to unseen input-output combinations. This ability enables models to flexibly accommodate variations in language. For instance, a model is likely to memorize that the token *green* is typically an adjective, yet still recognize that it is used as a noun in the sentence *The child sat on the main green* based on contextual information.

This flexibility breaks down on truly novel/unseen tokens. Much recent research has studied ICL algorithms in transformers (Chan et al., 2022b; Singh et al., 2023; Garg et al., 2023). This work focuses on ICL on heldout inputs that are imbued with semantic information. However, does ICL work on arbitrary inputs? Recent research suggests no: typical ICL algorithms fail when given undertrained (Land & Bartolo, 2024; Rumbelow & Watkins, 2023) or newly-introduced (e.g. when adding languages to an existing model) tokens (Chen et al., 2024). While these models appear to be performing task composition under the hood (Hahn & Goyal, 2023; Li et al., 2024), this still results in bizarre, non-deterministic behavior on queries such as asking GPT-3 to repeat back the string *SpaceEngineers* (Rumbelow & Watkins, 2023). We refer to these typical ICL algorithms as **conditional** ICL, as they break down when inputs have insufficient encoded information. In contrast, we define **structural ICL** to be the ability of a model to perform in-context learning on tokens without encoded information, defined more precisely in Section 2. We analyze this strong form of ICL along training in naturalistic and synthetic tasks.

---

[1] We release code here for reproducibility

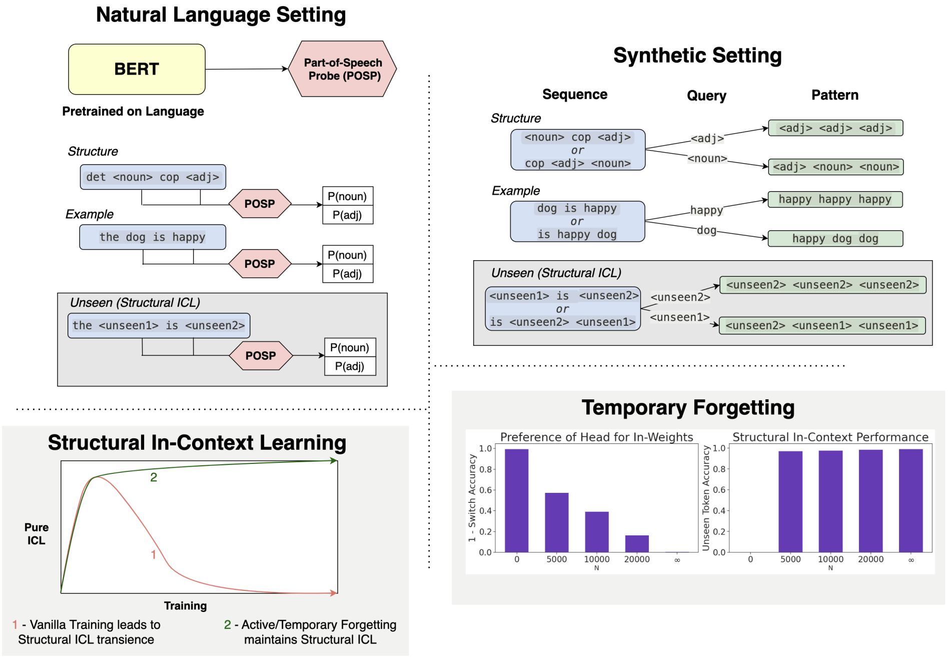


Figure 1: (Top Left) In our natural setting, we use a part-of-speech probe trained on BERT representations of sentences from Penn Treebank 3 and evaluate on templated examples (Section 3). (Top Right) In our synthetic setting, we train a small masked language model (MLM) on a grammar where the expected response is conditioned on the part-of-speech of the query (Section 4). (Bottom Left) An idealization of our main finding: structural ICL is transient (i.e. decays over training) in both natural and synthetic settings. Active/temporary forgetting maintains structural ICL in the synthetic setting. (Bottom Right) Our temporary forgetting procedure evokes structural ICL when applied for $N > 0$ steps, enabling generalization to unseen random tokens. In-weights preference is coarsely controllable by varying temporary forgetting parameter N.

Our research expands upon a burgeoning literature that uses the framework of ICL vs. in-weights learning (IWL) to study the development of transformers (e.g. Chan et al. (2022b), Singh et al. (2023), Reddy (2023)). Specifically, Chan et al. (2022b) finds that while ICL and IWL strategies are often in opposition, a ”sweet spot” language-like label distribution enables both ICL and IWL strategies to co-occur in the same model. This encoded *dual process* is crucial to current language models, allowing flexible, context-sensitive operations for out-of-distribution settings and memorized, static operations for ambiguous contexts or IID settings (Moskovitz et al., 2022; Kahneman, 2011; Miller, 2000)[2]. Building off this, Singh et al. (2023) finds that ICL slowly dissipates as models are overtrained; they discover that L2-regularization mitigates ICL transience, but instead leads to IWL transience. We utilize the framework proposed in this literature to dissect ICL emergence into structural vs. conditional ICL development. Moreover, we aim to translate insights from these studies to actionable strategies to improve models on non-language-like data distributions – specifically, we attempt to elicit powerful dual processes in arbitrary data distributions.

In our research, we find that structural ICL is *also* transient. However, while regularization provides a path to persistence in conditional ICL (Singh et al., 2023), it does not for structural ICL. Therefore, we propose an extension to active forgetting – a recent weight resetting technique introduced by Chen et al. (2024) to help augment models with new tokens – to make structural ICL persistent. Our modification allows us to coarsely control the strategies that the model adopts, enabling us to induce a dual process strategy: (structural) ICL for rare and unseen tokens and IWL for common tokens.

Our main contributions are:

- We define and study the concept of **structural ICL** in both large models and toy models. This allows for true generalization of in-context strategies for completely unseen tokens.

---

[2] We refer to in-weights learning v. in-context learning as a dual process. This connection is mainly intended to succinctly describe the phenomenon rather than draw concrete parallels to other dual processes.

We discover that both masked and autoregressive LMs exhibit a (limited) form of structural in-context learning that emerges early in training, but this ability quickly vanishes.

- We show active forgetting (Chen et al., 2024) maintains structural ICL in models. We introduce **temporary forgetting**, a straightforward extension of active forgetting that enables one to control how much a model relies on in-weights vs. in-context solutions.
- We demonstrate that when training with skewed token distributions, temporary forgetting enables us to induce a *dual process strategy* where our model uses an in-weights solution for frequently-seen tokens in the head of the distribution and a (structural) in-context solution for rare tokens in the tail.

2 DEFINITIONS

In-Context vs. In-Weights Learning We follow Reddy (2023), which defines in-weights learning (IWL) to be "query-response relationships encoded in the weights of the network" while in-context learning (ICL) emerges due to "common structural element[s]" and "can be exploited to perform zero-shot learning on novel tasks that share this structure."

We formulate our ICL prediction task as $\mathbb{P}(y \mid \mathbf{p}_{1:n}; \mathbf{z}_{1:n}; \mathbf{M}_{0:l})$ where y are the label(s), $\mathbf{p}_{1:n}$ is the set of positional embeddings and $\mathbf{z}_{1:n}$ is the set of word embeddings for a sequence of length n, and $\mathbf{M}_{0:l}$ is a length l transformer.

Within this framework, word embeddings are purely in-weight representations, which are enriched with context information by attention layers.

Structural vs. Conditional ICL We define structural ICL precisely via an empirical test: a model exhibits structural ICL if it can employ analogical reasoning from context in a way that is robust to arbitrary embeddings. For one or more word embeddings at specified position(s) $i \in I$, we replace $\mathbf{z}_i \xrightarrow{\text{replace}} \mathbf{z}_{\text{random}}$. This removes the in-weight signal of the word embedding and forces reliance on in-context information and structural analogy.

We state that a model can perform **conditional ICL** when it succeeds on prediction task $\mathbb{P}(y \mid \mathbf{p}_{1:n}; \mathbf{z}_{1:n}; \mathbf{M}_{0:l})$ when the ordered set $\mathbf{z}_{1:n}$ remains unmodified. This is the standard ICL setting studied by (Chan et al., 2022b; Singh et al., 2023; Garg et al., 2023; Akyürek et al., 2024). Note that a model exhibiting conditional ICL does not imply that the same model will exhibit structural ICL. Recent research suggests various models fail on undertrained "glitch tokens" that do not possess ample identity information z_i ((Rumbelow & Watkins, 2023; Land & Bartolo, 2024)).

Head vs. Tail In skewed token distributions, we refer to the most frequently occurring tokens (typically $\approx 10\%$) in a distribution of examples as the **head** of the distribution and the least frequently occurring tokens (typically $\approx 10\%$) as the **tail**. As token distributions increase in skew, tail tokens have less probability of being seen. This dichotomy relates very closely to our analysis of structural ICL in that tail tokens can be thought of as somewhere between fully-trained tokens and random tokens. By solving performance on random tokens, we can rectify ICL on rare tail tokens.

3 (STRUCTURAL) IN-CONTEXT LEARNING IS TRANSIENT

Recent work has discovered that conditional ICL capabilities slowly degrade over the course of long training in a synthetic setting (Singh et al., 2023). Inspired by this work, we reproduce it and track the structural ICL capabilities for encoder-only LMs in a naturalistic syntax probing task. We find that structural ICL *rapidly* degrades to completely random performance after relatively few training steps, while conditional ICL abilities remain present. To perform this developmental analysis, we study the various intermediate checkpoints released from the MultiBERTs (Sellam et al., 2021), averaging all of our results across seeds 0, 1, and 2. We calculate error bars in Figure 2 as ± 1 standard error of the mean (SEM).

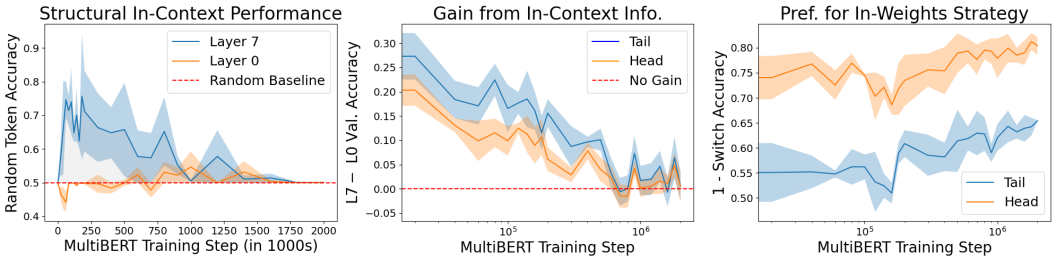


Figure 2: (Left) We exhibit the transience of structural ICL by examining the Random Token Accuracy over time. (Middle) We show the trend of memorization of tail versus head of distribution over training steps by examining the difference in Layer 7 Accuracy, where both in-context and in-weights strategies are possible, and Layer 0 Accuracy, where only an in-weights strategy is possible; (Right) We display the preference for in-weights strategy when conflicting with in-context strategy over time.

3.1 TASK

We design a task that employs templated stimuli to assess part of speech to tokens, permitting both ICL and IWL solutions. For instance, in the sentence *the dog is happy*, there are at least two ways of determining that *dog* is a noun: (1) memorize that the token identity "dog" is a noun or (2) extract that *dog* is the subject of the sentence from the context. Each dataset contains sentences that obey the template: The `<noun>` is `<adj>` (e.g. The dog is happy).

Our evaluation datasets are defined as follows:

1. **Head/Tail**: Templated examples where tokens are sampled from the most/least frequent 1500 nouns and most/least frequent 1500 adjectives in the training set of PTB-3.
2. **Head/Tail Switch**: Templated examples where tokens are sampled as in the "Head"/"Tail" dataset, but where nouns appear in the adjective position and adjectives appear in the noun position (e.g., *The happy is dog*).
3. **Random Token**: Templated examples where "nouns" and "adjectives" are sampled from a set of 1,500 randomly initialized tokens. This metric evaluates structural ICL performance[3].

For each layer and MultiBERT step, we train a binary POS probe on representations of nouns and adjectives from sentences in the training set of Penn Treebank 3 (PTB-3) (Marcus et al., 1993). For multi-token words, we average representations across tokens. See Appendix A.1 for additional details about our probing setup. Note that the MultiBERTs are trained following Devlin et al. (2019) on a combination of BookCorpus (Zhu et al., 2015) and English Wikipedia collected by Turc et al. (2019). As such, the distribution of the training data is fixed, and our experiments are constrained to the natural distribution of language. As BookCorpus does not have POS tags readily accessible, we employ PTB-3 to estimate the noun and adjective distribution of the training data. We defined nouns and adjectives as words that appeared as each POS, respectively, over 80% of the time. We chose 1500 examples as this is $\approx 10\%$ of the number of unique nouns.

3.2 TRAINING DYNAMICS

We examine (1) structural in-context learning and (2) the tradeoff between in-context and in-weight strategies over the course of training.

Structural ICL We find that the MultiBERTs are initially able to perform structural ICL, but that this capability is transient. In Figure 2 (Left), we present results from a probe trained on representations from Layer 7 as this layer achieves the highest probing validation performance on PTB-3. This is consistent with prior research which demonstrates that syntactic structures are encoded in the middle layers of MLMs (Tenney et al., 2019; Limisiewicz & Mareček, 2020). Furthermore, results across all layers are presented in Appendix A.2. Structural ICL transience is evident as probe performance on random tokens tend to spike early in MultiBERT training before dropping to chance

---

[3] We are able to generate novel labels not seen during train time because the embedding and unembedding matrices are tied in the MultiBERT models.

by the end of training. These results suggest that there is an inductive bias toward structural ICL that diminishes as information is encoded in the embeddings. As structural ICL confers the ability to generalize to rare and new tokens, transience raises questions about how we can train models that maintain this ability throughout training.

In-Context vs. In-Weights Strategies Much like Singh et al. (2023), we observe that conditional ICL strategies dissipate over training, as more information is encoded in token embeddings. We approximate the use of in-context information for determining POS as the difference in performance between Layer 0 (the embedding layer) and Layer 7. Layer 0 must rely only on in-weights information as there is no in-context information available; in contrast, Layer 7 uses contextualization to achieve higher performance (Tenney et al., 2019; Hewitt et al., 2021). The benefit of in-context information disappears more quickly for the head of the distribution than the tail, likely because there are far more gradient updates to head token embeddings.[4] As the benefit of the model's use of in-context information dissipates, we observe that the model shifts from an in-context to an in-weights strategy in Figure 2 (Right). In other words, models becomes more reliant on in-weights strategies and less reliant on in-context strategies over the course of training. This finding aligns with Singh et al. (2023).

4 SYNTHETIC TASK: DISTRIBUTION IMPACTS IN-CONTEXT LEARNING

We develop a synthetic masked language modeling task to characterize how data distributional parameters affect structural ICL, conditional ICL, and IWL. Our synthetic task requires the model to determine which of two classes a word belongs to. This may be derived either from in-context information or by memorizing token identity-class associations in the embedding layer. We draw analogies between these classes and POS in natural language.

Our vocabulary contains tokens that represent nouns, adjectives, and a copula (i.e. *is*). Each sentence is created by selecting (1) a `sequence` S, (2) a `query` Q, and (3) a response `pattern` P. Our MLM is trained to predict $\mathbb{P}(P_i|S, Q)$ for all $i \in \{0, \ldots, |P|-1\}$ (i.e. the probability of each pattern token). The `sequence` and `pattern` are arbitrary and designed so that no exceedingly simple heuristic may solve this task.

- **`sequence`** S: Either `<noun> <copula> <adj>` or `<copula> <adj> <noun>`.
- **`query`** Q: Either the `<noun>` or `<adj>` from the sequence.
- **`pattern`** P: Either `<adj> <noun> <noun>` if the `query` is a `<noun>` or `<adj> <adj> <adj>` if the `query` is an `<adj>`.

This task is designed such that the model must make a POS classification on the query token, and then perform an additional language-like operation conditioned on that classification (copying specific token identities in a specific order). See Appendix A.8 for more details. See Figure 1 and Appendix A.9 for examples.

We parameterize the task with vocabulary size v, the sampling distribution skew for nouns/adjectives α (where we select `<noun>, <ad>` $\sim$ Zipf(α)), and the ambiguity of token POS ε. The ambiguity parameter determines the percentage of tokens can act as both as noun and an adjective, and is inspired by the inherent ambiguity of POS in natural language. For our primary experiments, we fix $\varepsilon = 0.10$. Note, we find that ε must be greater than zero for an in-context solution to emerge at all. We compare our skewed distribution results to sampling tokens from a Uniform distribution.

In this task, an ICL solution to derive the POS of the `query` may achieve perfect accuracy by utilizing in-context information (e.g. a *copula* is always followed first by an adjective, then a noun). In contrast, an IWL solution to derive the POS of the `query` may achieve at most an accuracy of $(1 - \varepsilon/2)$ due to ambiguous tokens. To account for this, we evaluate our models only on tokens that are not ambiguous; thus, both an ICL and IWL solution could achieve perfect accuracy. (Ambiguous tokens always use an ICL solution.)

Our task is formatted in a cloze-style where each token in the pattern is masked. We employ a BERT-style MLM (Devlin et al., 2019) to predict the identities of these masked tokens, with hyperparameters

---

[4] We observe that performance gain due to the model's use of in-context information decreases across a wide range of syntactic phenomena as embeddings are enriched during training. We term this the "Pushdown Phenomenon" and explore it more thoroughly in Appendix A.7.

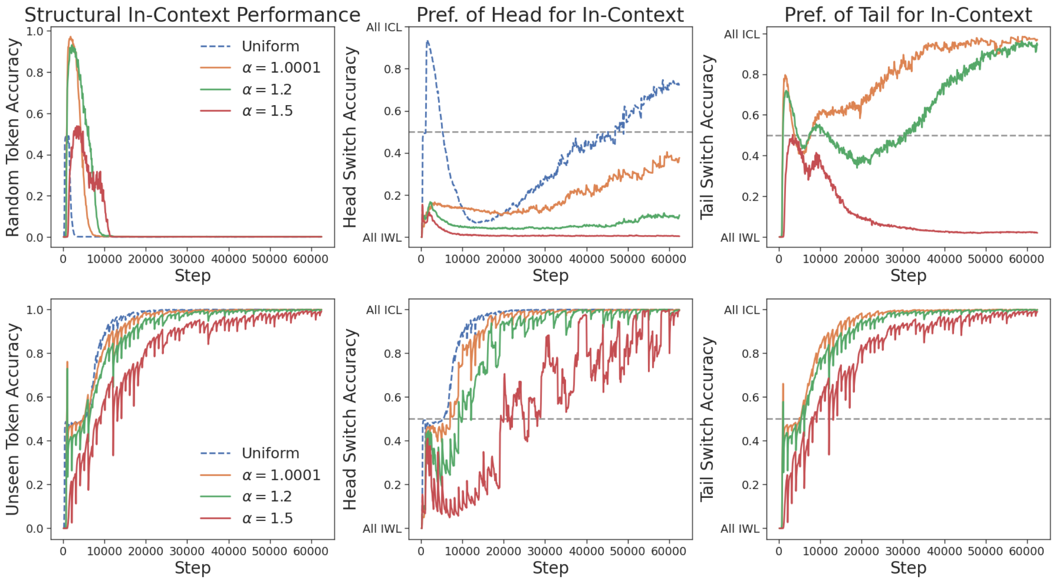


Figure 3: (Top) In-context performance by distribution with **vanilla training**; (Bottom) In-context performance by distribution with **active forgetting**. The parameters used are $v = 10000, \varepsilon = 0.10$. Note that the Uniform distribution does not have a head or a tail, so its results are in the head graphs. Vanilla training results in structural ICL transience (Top Left), but conditional ICL is asymptotically nonzero (Top Middle, Top Right). Examples with head tokens usually prefer IWL (Top Middle) while examples with tail tokens usually prefer ICL (Top Right). In contrast, active forgetting preserves structural ICL and removes all preference for IWL across distributions (Bottom Row). With active forgetting, examples with head tokens usually prefer and examples with tail tokens both prefer ICL (Bottom Middle, Bottom Right). In skewed distributions ($\alpha = 1.5$), the loss of structural ICL results in a loss of all ICL (Top Middle, Top Right - Red Line).

described in Appendix A.10. Near-perfect validation accuracy is achieved after $< 60,000$ steps on all experimental settings.

In addition to performance on a randomly selected validation set, we create datasets to evaluate the model's preferred strategy throughout training, similar to Section 3. All examples in these datasets contain novel `<adj>, <noun>` pairs. Much like our naturalistic setting metrics in Section 3.1, we create Tail, Head, Head Switch, Tail Switch, and Unseen Token Accuracy metrics. In this setting, our head and tail metrics use the top and bottom 10% of the token distribution by count, respectively.

4.1 TRAINING DYNAMICS

Structural ICL is Not Conditional ICL We reproduce the results from the natural language setting presented in Section 3: structural in-context solutions emerge quickly, but are transient. This is shown by the early peak of Random Token Accuracy, followed by its steep drop, a trend which holds across all tested distributions in Figure 3 (Top Left). As such, both the syntactic and naturalistic settings align with our idealized graph of structural ICL transience exhibited in Figure 1 (Bottom Left). However, the disappearance of a structural in-context algorithm occurs more quickly here than in our MultiBERT experiments, likely due to the simplicity of our synthetic task.

Critically, the top row of Figure 3 shows that even though structural ICL performance degrades quickly, conditional ICL abilities remain. Across all distributions, both the head and the tail show reliance on conditional ICL asymptotically (Figure 3 Top Middle, Top Right) while structural ICL remains zero. In a modified version of the Chan et al. (2022b) task, we find that the same trend of structural ICL disappearance and conditional ICL continuation remain consistent (Figure 6, Left).

Structural ICL has Practical Importance In highly skewed distributions (e.g. Zipf $\alpha \geq 1.5$) where tail tokens are very rare and head tokens are very common, the disappearance of structural ICL results

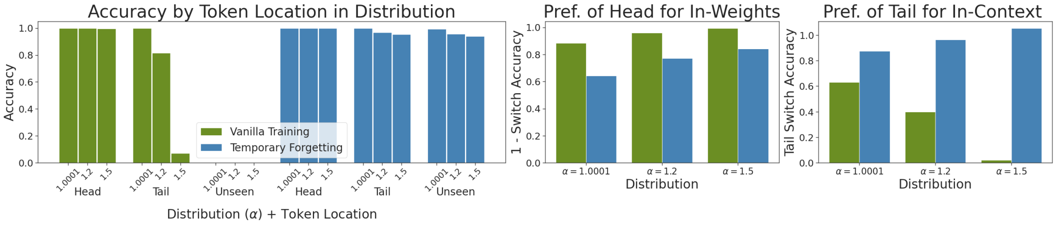


Figure 4: (Left) Temporary forgetting achieves near perfect random token performance (structural in-context) asymptotically among distributions. Tail performance of skewed distributions is poor (Left - Green) after vanilla training (i.e. standard training); in contrast, tail performance is almost perfect after temporary forgetting (Left- Blue). (Right) Temporary forgetting can asymptotically hold preference for an in-weights strategy in the head of the distribution while holding preference for an in-context strategy in the tail of the distribution (i.e. learn dual processes). Parameters used are $v = 10000, \varepsilon = 0.10$ and optimal hyperpameters k, N over gridsearch.

in a total loss of ICL abilities (Figure 3 Top - Red Line). Common tokens are memorized resulting in high overall performance of examples; however, tail tokens fail altogether (See Figure 4, Left). Even when conditional ICL abilities remain in less skewed distributions, the least-frequent subset of tail tokens have poor performance. We theorize that this finding is analogous to the "glitch token" issue that plagues current language models (Land & Bartolo, 2024). Structural ICL would rectify performance on these undertrained tokens.

In-Context Learning conflicts with In-Weights Learning Typically, conditional ICL and IWL are in conflict. Increasing the skew of a distribution increases the pressure toward an IWL strategy. Conversely, examples with tokens drawn from a Uniform sampling distribution show a comparatively higher conditional ICL preference (and thus lower IWL preference) than any Zipfian sampling distribution in Figure 3, Top Middle. Among Zipfian sampling distributions, the model's strategy varies based on whether the adjective and noun are in the head or the tail of the token distribution, much like in our naturalistic task.[5] As in our naturalistic setting, we find head tokens prefer IWL while tail tokens prefer conditional ICL. We will explore how to mitigate this competition Section 6.

5 MAINTAINING STRUCTURAL ICL WITH ACTIVE FORGETTING

In Sections 3 and 4, we have demonstrated structural ICL is transient across models and tasks. In an effort to promote structural ICL persistence, we utilize a recently-introduced training procedure: *active forgetting* (Chen et al., 2024). Note we refer to *vanilla training* as the standard training procedure without special interventions.

Active Forgetting When training a model using active forgetting, we re-initialize the embedding matrix every k steps during training. The intuition behind this is that the model *must* employ in-context strategies to achieve high accuracy, each token's embedding is no longer guaranteed to have in-weight information. The unseen/undertrained tokens that were before out-of-distribution now are in-distribution, an effect which we further explore in Appendix A.14.

Results We test $k = 100, 1000, 5000$ and settle on $k = 1000$, as this worked well in our preliminary exploration. Training our models with active forgetting promotes asymptotic structural ICL across all tested skews, enabling the model to approach perfect performance on the random Unseen Token Set (See Figure 3, Bottom Left). Given random embeddings representing a noun and an adjective, the model can now (1) derive the POS of these tokens by ICL and (2) output novel labels corresponding to the identity of these embeddings in the desired pattern.[6] Note that we see a slightly more stochastic version of our idealized trend from Figure 1, Bottom Left due to the resetting mechanism. We find that this trend of active forgetting preserving structural ICL holds for other task-model combinations, such as a modified Chan et al. (2022b) task. (See Figure 6, Middle).

---

[5] Additional experiments exploring the effect of ambiguity are located in Appendix A.12 and the effect of vocabulary size are located in Appendix A.13.

[6] This relies on the embedding and unembedding matrix being tied

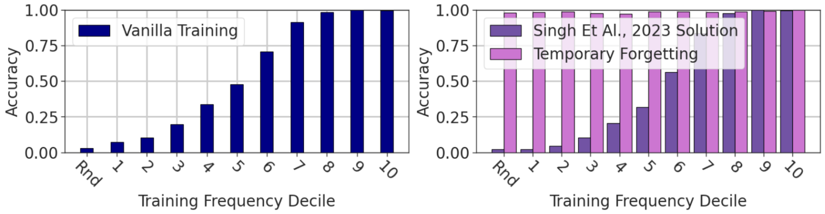


Figure 5: Performance by token decile and on random tokens (Rnd). With vanilla training in a skewed distribution (Zipfian $\alpha = 1.5$), low decile tokens show poor performance; however, overall performance remains good because these tokens are rare. Temporary forgetting preserves structural ICL to solve performance on tail, undertrained, and unseen tokens compared with Singh et al. (2023)'s L2-regularization procedure, which was proposed to preserve conditional ICL.

As the skew of the distribution of nouns and adjectives increases, there is greater pressure to memorize the head of the distribution (as these tokens are observed more frequently). Thus, it takes longer for the model to exhibit a preference towards in-context solutions for head tokens (e.g. almost 60,000 steps for the $\alpha = 1.5$ setting) and there is a much larger drop-off in performance after every instance of forgetting the embedding matrix.

6 DUAL PROCESS LEARNING WITH TEMPORARY FORGETTING

While active learning successfully induces a structural ICL strategy, our model loses the ability to memorize information in its embeddings. This is detrimental in a variety of cases, such as when in-context information is insufficient to generate an appropriate response. An optimal model would encode a *dual process strategy*: maintaining a structural ICL solution while also memorizing useful linguistic properties.

Temporary Forgetting We modify the paradigm of active forgetting to attempt to induce a bias for structural in-context strategies in the tail of the distribution while preserving the in-weights solutions for frequently-observed tokens. We introduce **temporary forgetting**, where we perform active forgetting every k steps for the first N steps ($N >> k$) of training. After this point, we allow the embedding matrix to train as normal. As a baseline, we compare to Singh et al. (2023)'s solution to conditional ICL transience, L2 regularization. Crucially, we wish to understand whether L2 regularization helps maintain *structural* ICL, which was not tested in the original work.

Results We find that by varying N, we can vary the model's dependence on in-weights information on frequently seen tokens while maintaining structural ICL performance as displayed in Figure 1, Bottom Right (parameters used are $v = 10000, \varepsilon = 0.10, \alpha = 1.5$). At the extremes, setting N to be very large mimics the behavior of active forgetting and setting N to be small only *sometimes* maintains structural ICL performance. We can control the preference for IWL versus ICL on observed tokens by modifying N (See Figure 1, Bottom Right).

Thus, temporary forgetting enables a model to successfully encode two distinct strategies for the same task. We can now induce this behavior for any distribution $\alpha \geq 1.0$, while also inducing structural ICL behavior on *all* distributions we test (See Figure 4, Right).[7] Note that the control granted by temporary forgetting over head IWL preference has limits – we can push up to almost 90% the original IWL preference while maintaining a high tail ICL preference as seen in Figure 4. In contrast, we find that the strategy suggested by (Singh et al., 2023) does *not* eliminate structural ICL transience: undertrained and random tokens induce very poor performance, as seen in Figure 5.

Temporary forgetting imparts an incentive that significantly enhances our ability to balance between in-context and in-weights strategies, overcoming inherent biases in naturally occurring data. After a critical period, we can stop the forgetting mechanism and retain structural ICL abilities.

---

[7] Distributions where $\alpha \leq 1.0$ would likely only rely on an in-context strategy

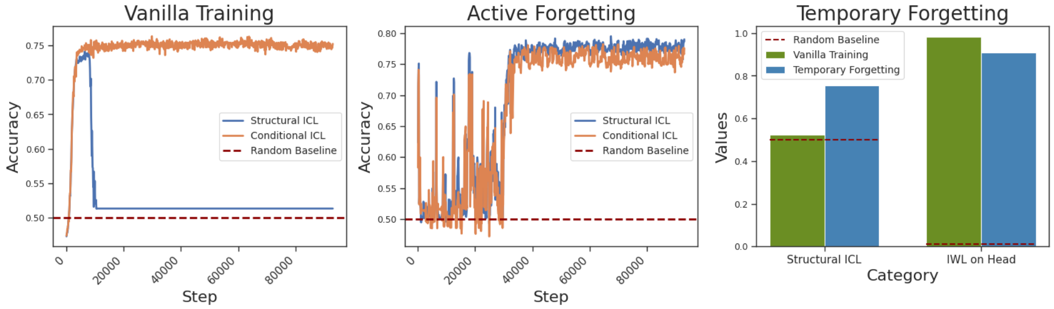


Figure 6: Results from our replication study on the Chan et al. (2022b) task with an autoregressive transformer. (Left) With vanilla training, structural ICL is transient while conditional ICL remains asymptotically. (Middle) Training with active forgetting preserves structural ICL. (Right) When fitting our model to a skewed token distribution (Zipfian $\alpha = 3$), vanilla training results in memorization of head tokens and random performance on structural ICL; in contrast, temporary forgetting evokes a dual process, which significantly improves structural ICL performance while preserving IWL on common tokens. Task details and additional experiments found in Appendix A.4.

7 REPLICATION USING CHAN ET AL. (2022B) TASK

We replicate our main findings using an autoregressive transformer on a task similar to Chan et al. (2022b). Notably, we modify the task presented in Chan et al. (2022b) to enable us to examine structural ICL (See Appendix A.4).[8] We find that the phenomena described in Sections 3, 4.1, 5, and 6 all extend to this new task.

In the original task of Singh et al. (2023) (the same as Chan et al. (2022b)'s task), undertrained/random tokens were not tested. This prevented the possibility of observing specific *structural* transience, which remained unsolved (See Figure 5). We replicate their autoregressive task, but introduce randomly initialized embeddings for heldout classes. Our results are shown in Figure 6, which shows that the structural ICL transience remains an issue that is remedied by active/temporary forgetting.

8 RELATED WORK

In Context v. In Weights A body of recent literature closely examines in-weights versus in-context learning (Chan et al., 2022b;a; Reddy, 2023; Raparthy et al., 2023; Fu et al., 2024). The emergence of in-context learning abilities in transformers has been shown to depend on the distributional properties of the training data such as burstiness, training class rarity, and dynamic meaning (Chan et al., 2022b; Reddy, 2023). While we employ a similar analytical framework to this work, we (1) consider truly random heldout inputs and novel outputs/labels, (2) evaluate on large, natural language models, and (3) consider structural ICL (defined in Section 2). Additionally, while slow transience of conditional ICL has been noted in Singh et al. (2023), we find abrupt transience of structural ICL and introduce temporary forgetting to (1) preserve structural ICL and (2) solve what both Singh et al. (2023) and Chan et al. (2022b) suggest to be an extremely useful behavior: the co-existence of in-context learning and in-weights learning.

More broadly, the conflict between context-dependent and context-independent (or reflexive) solutions has been well-studied in the cognitive and computational neuroscience literature (Russin et al., 2024; Rougier et al., 2005; Russin et al., 2022). A key feature of human intelligence, termed *cognitive control*, is the ability to maintain dual strategies and flexibly deploy either one in response to particular stimulus. Any artificial system that aspires to producing human-like behavior must therefore be capable of maintaining both of these solutions.

Weight Forgetting To Help Learn. While most literature on *forgetting* characterizes this phenomenon as undesirable (Kemker et al., 2017; Kirkpatrick et al., 2017; McCloskey & Cohen, 1989;

---

[8] This modification ensures the tail distribution of these tokens are undertrained/untrained and thereby resemble the ”glitch tokens” of Rumbelow & Watkins (2023)

Ratcliff, 1990), recent neuroscience literature has shown that *intentional* forgetting may have positive roles in certain contexts (Srivastava et al., 2014; Pastötter et al., 2008; Levy et al., 2007; Anderson & Hulbert, 2021). Intentional forgetting in neural networks is accomplished by resetting a subset of parameters during training. On computer vision tasks, this resetting procedure has been shown to help low compute and data resource generalization (Alabdulmohsin et al., 2021; Taha et al., 2021; Ramkumar et al., 2023). Additionally, Zhou et al. (2022) show that a *forget-and-relearn* paradigm helps language emergence. Our method of forgetting embeddings is directly inspired by Chen et al. (2024), which shows forgetting during pretraining boosts linguistic plasticity for multilingual learning. As far as we know, we are the first to propose using forgetting to induce ICL.

9 DISCUSSION

The ability to flexibly deploy in-context and in-weights algorithms has been described as an "important and useful [behavior] for a model," as it enables models to both memorize information about commonly-seen classes and generalize to new classes Chan et al. (2022b). However, it has proven difficult to ensure that model's reliably acquire both forms of processing. While prior work celebrates the ability to maintain dual strategies even for a limited set of distributions (and go so far as to suggest "engineer[ing] data distributions to evoke this behavior in models" (Chan et al., 2022b)), the present work demonstrates a method for engendering both solutions across a range of distributions. Moreover, we extend the scope of the in-context algorithms under consideration: while prior work focuses on what we term *conditional* in-context learning, we achieve *structural* in-context learning. This allows the model to generalize its in-context algorithms to unseen and undertrained tokens.

Structural In-Context Learning One of our key findings is the transience of structural ICL in LMs. Initially, models exhibit a strong ability to leverage structural ICL, generalizing algorithms to unseen tokens. However, this capability dissapears as training progresses, suggesting an initial inductive bias towards structural ICL that wanes as the model learns. This transience limits generalization on rare tokens and new tokens. We find that active forgetting maintains structural ICL by repeatedly reinitializating the embeddings. Our temporary forgetting training procedure enables a dual process strategy through strategic re-initialization of weights. This enables adaptability while still leveraging accumulated knowledge. Specifically, structural ICL could significantly improve performance on abstract/symbolic reasoning tasks, which is important for various use cases including low-resource language ICL and code generation.

Implications for Model Training and Application Our findings are useful to design training protocols that result in flexible models. A significant reason for the success of LMs is their capacity for ICL and IWL strategies to co-exist, a behavior that organically occurs with a moderately skewed Zipfian distribution. However, most natural domains such as protein discovery, network traffic, and video recording face even more skew, breaking down this ideal behavior. Our temporary forgetting technique facilitates a dual process strategy regardless of skew, which could potentially bring some of the profound success of LMs to other domains.

Future Directions and Limitations The research opens up several avenues for future investigation. One significant limitation is that our temporary forgetting experiments were not performed on natural LMs. Our compute resources limited such experiments, but we believe this is a critical future step to refining this training intervention. Another potential limitation of our work is that the optimal hyperparameters to temporary forgetting are not known a priori; while we do not explore this, our experiments suggest performance is relatively robust to hyperparameter selection. Finally, another avenue of fruitful future research may be the translation of structural ICL algorithms into symbolic systems. As structural ICL does not rely on the content of the input, it should be possible to use techniques like circuit analysis (Räuker et al., 2023) to reverse-engineer an explicit symbolic representation of the algorithm that the neural network uses to solve a task.

Conclusion This study deepens our understanding of a model's adoption of structural ICL, conditional ICL, and IWL strategy during training. The techniques introduced here not only enhance our theoretical understanding but also offer practical tools for improving model training and functionality in real-world applications.(Gentner, 1983).

REFERENCES

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: Architectures and algorithms, 2024. URL `https://arxiv.org/abs/2401.12973`.

Ibrahim Alabdulmohsin, Hartmut Maennel, and Daniel Keysers. The impact of reinitialization on generalization in convolutional neural networks, 2021.

Michael C Anderson and Justin C Hulbert. Active forgetting: Adaptation of memory by prefrontal control. *Annual Review of Psychology*, 72(1):1–36, 2021. doi: 10.1146/annurev-psych-072720-094140. URL `https://doi.org/10.1146/annurev-psych-072720-094140`.

Nora Belrose, Quintin Pope, Lucia Quirke, Alex Mallen, and Xiaoli Fern. Neural networks learn statistics of increasing complexity, 2024.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020. URL `https://arxiv.org/abs/2005.14165`.

Stephanie C. Y. Chan, Ishita Dasgupta, Junkyung Kim, Dharshan Kumaran, Andrew K. Lampinen, and Felix Hill. Transformers generalize differently from information stored in context vs in weights, 2022a.

Stephanie C. Y. Chan, Adam Santoro, Andrew K. Lampinen, Jane X. Wang, Aaditya Singh, Pierre H. Richemond, Jay McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers, 2022b.

Yihong Chen, Kelly Marchisio, Roberta Raileanu, David Ifeoluwa Adelani, Pontus Stenetorp, Sebastian Riedel, and Mikel Artetxe. Improving language plasticity via pretraining with active forgetting, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey on in-context learning, 2023.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. When bert forgets how to POS: amnesic probing of linguistic properties and MLM predictions. *CoRR*, abs/2006.00995, 2020. URL `https://arxiv.org/abs/2006.00995`.

Jingwen Fu, Tao Yang, Yuwang Wang, Yan Lu, and Nanning Zheng. How does representation impact in-context learning: An exploration on a synthetic task, 2024. URL `https://openreview.net/forum?id=JopVmAPyx6`.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023.

Dedre Gentner. Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170, 1983. ISSN 0364-0213. doi: https://doi.org/10.1016/S0364-0213(83)80009-3. URL `https://www.sciencedirect.com/science/article/pii/S0364021383800093`.

Michael Hahn and Navin Goyal. A theory of emergent in-context learning as implicit structure induction, 2023. URL `https://arxiv.org/abs/2303.07971`.

John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1419. URL `https://aclanthology.org/N19-1419`.

John Hewitt, Kawin Ethayarajh, Percy Liang, and Christopher Manning. Conditional probing: measuring usable information beyond a baseline. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 1626–1639, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.122. URL `https://aclanthology.org/2021.emnlp-main.122`.

Daniel Kahneman. *Thinking, fast and slow*. macmillan, 2011.

Ronald Kemker, Angelina Abitino, Marc McClure, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. *ArXiv*, abs/1708.02072, 2017. URL `https://api.semanticscholar.org/CorpusID:22910766`.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. ISSN 1091-6490. doi: 10.1073/pnas.1611835114. URL `http://dx.doi.org/10.1073/pnas.1611835114`.

Sander Land and Max Bartolo. Fishing for magikarp: Automatically detecting under-trained tokens in large language models, 2024.

Benjamin J. Levy, Nathan D. McVeigh, Alejandra Marful, and Michael C. Anderson. Inhibiting your native language: The role of retrieval-induced forgetting during second-language acquisition. *Psychological Science*, 18(1):29–34, 2007. ISSN 09567976, 14679280. URL `http://www.jstor.org/stable/40064573`.

Jiaoda Li, Yifan Hou, Mrinmaya Sachan, and Ryan Cotterell. What do language models learn in context? the structured task hypothesis, 2024. URL `https://arxiv.org/abs/2406.04216`.

Tomasz Limisiewicz and David Mareček. Syntax representation in word embeddings and neural networks – a survey, 2020.

Linguistic Data Consortium. Ontonotes release 5.0. `https://catalog.ldc.upenn.edu/LDC2013T19`, 2013. Accessed on December 10, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, jun 1993. ISSN 0891-2017.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower (ed.), *Psychology of Learning and Motivation*, volume 24 of *Psychology of Learning and Motivation*, pp. 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8. URL `https://www.sciencedirect.com/science/article/pii/S0079742108605368`.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal Dependency annotation for multilingual parsing. In Hinrich Schuetze, Pascale Fung, and Massimo Poesio (eds.), *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 92–97, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL `https://aclanthology.org/P13-2017`.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Earl K Miller. The prefontral cortex and cognitive control. *Nature reviews neuroscience*, 1(1):59–65, 2000.

Ted Moskovitz, Kevin Miller, Maneesh Sahani, and Matthew Botvinick. A unified theory of dual-process control, 11 2022.

Liam Parker, Emre Onal, Anton Stengel, and Jake Intrater. Neural collapse in the intermediate hidden layers of classification neural networks, 2023.

Bernhard Pastötter, Karl-Heinz Bäuml, and Simon Hanslmayr. Oscillatory brain activity before and after an internal context change - evidence for a reset of encoding processes. *NeuroImage*, 43: 173–81, 08 2008. doi: 10.1016/j.neuroimage.2008.07.005.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In Sameer Pradhan, Alessandro Moschitti, and Nianwen Xue (eds.), *Joint Conference on EMNLP and CoNLL - Shared Task*, pp. 1–40, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL `https://aclanthology.org/W12-4501`.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Vijaya Raghavan T. Ramkumar, Elahe Arani, and Bahram Zonooz. Learn, unlearn and relearn: An online learning paradigm for deep neural networks, 2023.

Akshay Rangamani, Marius Lindegaard, Tomer Galanti, and Tomaso A Poggio. Feature learning in deep classifiers through intermediate neural collapse. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28729–28745. PMLR, 23–29 Jul 2023. URL `https://proceedings.mlr.press/v202/rangamani23a.html`.

Sharath Chandra Raparthy, Eric Hambro, Robert Kirk, Mikael Henaff, and Roberta Raileanu. Generalization to new sequential decision making tasks with in-context learning, 2023.

Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97 2:285–308, 1990. URL `https://api.semanticscholar.org/CorpusID:18556305`.

Gautam Reddy. The mechanistic basis of data dependence and abrupt learning in an in-context classification task, 2023.

Nicolas P Rougier, David C Noelle, Todd S Braver, Jonathan D Cohen, and Randall C O'Reilly. Prefrontal cortex and flexible cognitive control: Rules without symbols. *Proceedings of the National Academy of Sciences*, 102(20):7338–7343, 2005.

Jessica Rumbelow and Matthew Watkins. Solidgoldmagikarp (plus, prompt generation). *LessWrong*, 2023. URL `https://www.lesswrong.com/posts/aPeJE8bSo6rAFoLqg/solidgoldmagikarp-plus-prompt-generation`.

Jacob Russin, Maryam Zolfaghar, Seongmin A Park, Erie Boorman, and Randall C O'Reilly. A neural network model of continual learning with cognitive control. In *CogSci... Annual Conference of the Cognitive Science Society. Cognitive Science Society (US). Conference*, volume 44, pp. 1064. NIH Public Access, 2022.

Jacob Russin, Ellie Pavlick, and Michael J Frank. Human curriculum effects emerge with in-context learning in neural networks. *arXiv preprint arXiv:2402.08674*, 2024.

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks, 2023.

Thibault Sellam, Steve Yadlowsky, Jason Wei, Naomi Saphra, Alexander D'Amour, Tal Linzen, Jasmijn Bastings, Iulia Turc, Jacob Eisenstein, Dipanjan Das, Ian Tenney, and Ellie Pavlick. The multiberts: BERT reproductions for robustness analysis. *CoRR*, abs/2106.16163, 2021. URL `https://arxiv.org/abs/2106.16163`.

Aaditya K Singh, Stephanie C.Y. Chan, Ted Moskovitz, Erin Grant, Andrew M Saxe, and Felix Hill. The transient nature of emergent in-context learning in transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=Of0GBzow8P`.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

Ahmed Taha, Abhinav Shrivastava, and Larry Davis. Knowledge evolution in neural networks, 2021.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1452. URL `https://aclanthology.org/P19-1452`.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models, 2019.

Hattie Zhou, Ankit Vani, Hugo Larochelle, and Aaron Courville. Fortuitous forgetting in connectionist networks, 2022.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 19–27, 2015. doi: 10.1109/ICCV.2015.11.

A Appendix / supplemental material

A.1 Probing Setup

We provide probing background in this section, borrowing some notation from Elazar et al. (2020).

Given a set of labeled data of points $X = x_1, \ldots x_n$ and task labels $Y = y_1, \ldots, y_n$, we analyze a model f that predicts the labels Y from $X : \hat{y_i} = f(x_i)$. We assume that this model is composed of two parts: (1) an encoder h that transforms input x_i into a learned representation vector $\mathbf{h}_{x_i}$ and (2) a classifier c that is used for predicting $\hat{y_i}$ based on $\mathbf{h}_{x_i}$, such that $\hat{y_i} = c(h(x_i))$. We refer by *probe* to the classifier c and refer by *model* to the model from which the encoder h is a subset of.

Given this setup, we evaluate a particular model's performance across various layers and training steps for our POS task. Each encoder h is associated with a specific training step and layer $h^{t,l}$. We probe the residual stream after layer l.

In this research, we are interested in the model's choice of strategy at a particular time step. That is, we seek to describe the change in prediction of $\hat{y_i}$ due to varying t, l of encoder $h^{t,l}$. Accordingly, we fix c as a single linear fully-connected layer.

A.2 Structural ICL across Layers

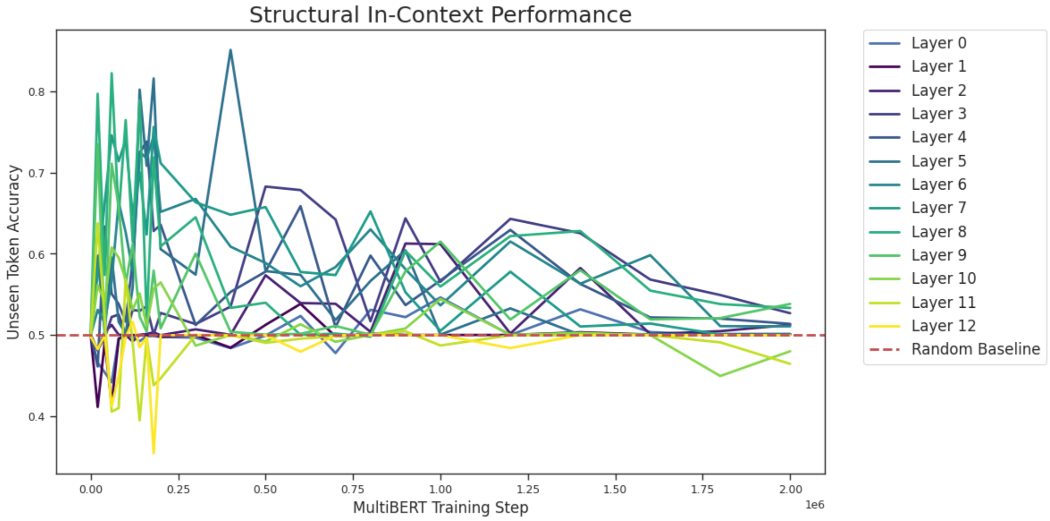


Figure 7: We find that structural ICL is transient across all layers of MultiBERTs (seeds 0, 1, 2 averaged). The middle layers show the most structural ICL during early in training, whereas very early and very late layers remain about random throughout training.

We find that structural ICL consistently approachs random levels as training progresses across layers in the MultiBERTs. This signifies that the model fully loses the ability to process unseen tokens as training continues. This is likely the reason for the “glitch tokens” described in Land & Bartolo (2024), for which LMs fail to output sensible content.

A.3 STRUCTURAL ICL IN GENERATIVE DECODER-ONLY LANGUAGE MODELS

A.3.1 SYLLOGISM TASK

We use a syllogism task that requires symbolic reasoning based on the context to show that (1) structural ICL is transient in a decoder-only transformer based on generation and (2) a variant of temporary forgetting can remedy structural ICL on a real natural langauge model.

Our task is formulated as follows: Our task requires abstract reasoning on untrained tokens in a decoder-only transformer. The model must complete the following syllogism.

All `<X>` are `<Y>`.
All `<Y>` are `<Z>`.
Therefore, all `<X>` are __

The correct answer is `<Z>`. We examine accuracy, which we define as the probability of choosing `<Z>` compared to the probability of choosing `<Y>`. We test *baseline performance* over training steps where `<X>`, `<Y>`, `<Z>` are chosen from the set of tokens representing A-Z, and we test *unseen token performance* by replacing `<X>` with an unseen token in this formulation (`<Y>`, `<Z>` are still chosen from A-Z).

A.3.2 STRUCTURAL ICL IS TRANSIENT IN PYTHIA 1.4B

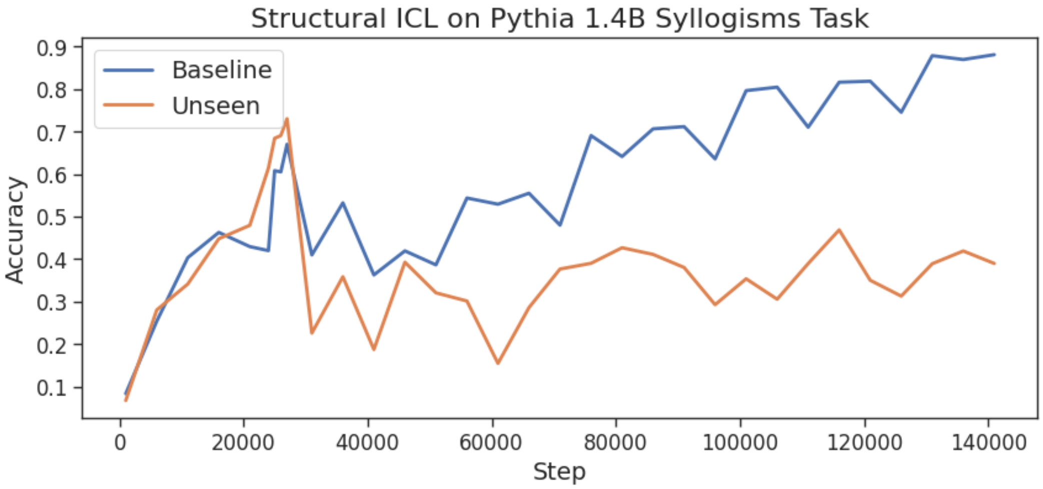


Figure 8: We find that structural ICL is transient for the decoder-only Pythia-1.4B on a syllogisms task. Performance on common tokens continues improving to near-perfect accuracy. Results averaged over three trials.

We find that structural ICL consistently spikes and then approaches below random levels as training progresses across layers in the Pythia-1.4B model (Biderman et al., 2023), as shown by the unseen token accuracy in Figure 8. The model loses the ability to perform syllogisms on unseen tokens as training continues. We chose the Pythia-1.4B model to show the generalizability of our finding to natural language decoder-only models. We employ publicly released training checkpoints to run our experiments.

A.3.3 PROBABILISTIC TEMPORARY FORGETTING FIXES STRUCTURAL ICL IN GPT-2

We finetune GPT-2 large (Radford et al., 2019) on Wikitext (Merity et al., 2016) sentences taken from Wikepedia articles for 2000 steps. Note that unseen token syllogism performance on the pretrained GPT-2 large is even worse than on the pretrained Pythia 1.4B. To accommodate the fine-tuning setting, we use a probabilistic variant of temporary forgetting: every step, we replace tokens in the batch with $p = 0.10$ with randomly initialized embeddings. After the step, we set the embedding matrix

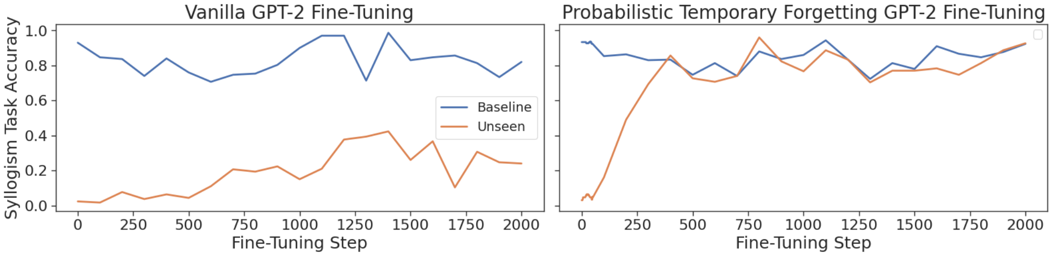


Figure 9: We find that structural ICL is transient for the decoder-only Pythia-1.4B on a syllogisms task. Performance on common tokens continues improving to near-perfect accuracy. Results averaged over three trials.

back to its original values, hence maintaining the spirit of temporary forgetting. In this method, our pretrained embeddings remain unchanged.

After fine-tuning with probabilistic temporary forgetting on Wikipedia sentences, we find that syllogism accuracy with unseen tokens jumps from 0.02 to 0.927 while the baseline syllogism accuracy goes from 0.933 to 0.923, as seen in Figure 9. In addition, when we fine-tune without probabilistic temporary forgetting (i.e. vanilla fine-tuning), we see that unseen token syllogism accuracy remains substantially below-random. Our probabilistic temporary forgetting rectifies structural ICL on a downstream task in a real natural language model.

A.4 AUTOREGRESSIVE TRANSFORMER SYNTHETIC SETTING

To show the broadness of our structural ICL results, we also replicate our findings using a modified version of the synthetic task presented in Chan et al. (2022b).

A.4.1 MODIFIED CHAN ET AL. (2022B) TASK

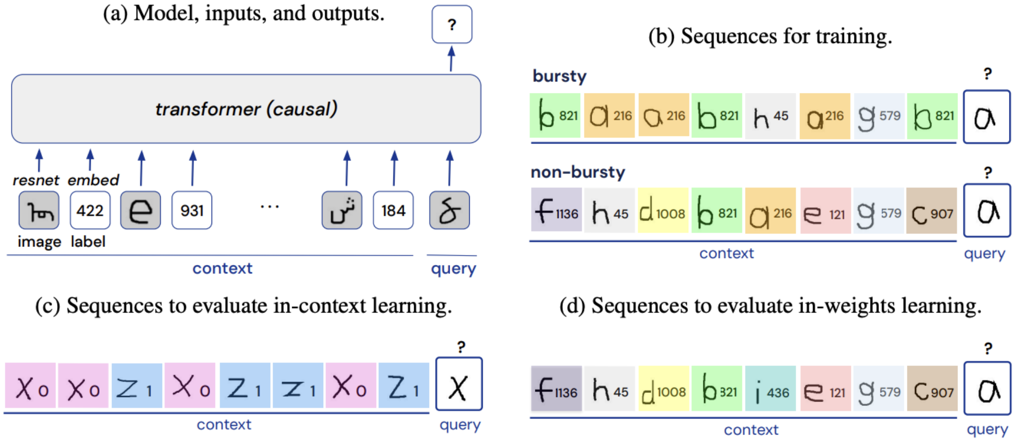


Figure 10: This is the setup of Chan et al. (2022b) (Figure 1 of their paper). We use a similar setup, but with token embeddings that are learned during training rather than ResNet encodings of Omniglot.

Similar to Chan et al. (2022b)'s setup, we have training data comprised of sequences of tokens and labels where the context is made up of the first 16 elements (8 token-label pairs), and the final element is the ‘query’ token. The aim of the model is to predict the correct label for the query. There are 1600 tokens, each mapped to a label, however with a ambiguity probability of 0.05 is mapped to a different label randomly chosen from the set of labels (closely resembling the multiplicity of labels experiments to promote ICL). Sequences are bursty, with the query-label pair as well a different token-label pair each occurring 3 times in the context. We evaluate the trained models on three types of sequences to measure (1) structural ICL, (2) conditional ICL, and (3) IWL.

Again borrowing from Chan et al. (2022b), our context for the ICL setups is a random ordering of two token-label pairs with 4 examples each, and the query is selected randomly from one of the two tokens. While label-pairs are fixed in training (with $p = 0.95$), the labels for the two tokens are randomly re-assigned to either 0 or 1 for each sequence. We calculate few-shot accuracy by considering only probabilities assigned to 0 and 1 (resulting in a chance of 0.5). In evaluating structural ICL, we generate sequences consisting of random tokens and labels while conditional ICL sequences consisted of tokens previously seen by the model during training. We test on tokens drawn from uniform and zipfian distributions, where experiments are with a Zipf $\alpha = 1.0001$ token sampling distribution unless otherwise specified.

To measure IWL, we considered non-bursty sequences where the query-label is not located in the context. The only way for a model to correctly predict the label is to rely on information in weights as we ensured unique, non-query token-label pairs in the context.

Note that the difference from Chan et al. (2022b)'s setup is that we use randomly initialized tokens embeddings rather than Omniglot Resnet-encoded images and our autoregressive transformer is also smaller. This enables us to test for structural ICL by replacing token identities with random vectors. Another method for us to test structural ICL could have been to use random images, but this would not have been analogous to the issue of undertrained/unseen ”glitch tokens” in language models, unlike our current setup

A.4.2 MODEL DESCRIPTION

We use a 4-layer GPT-2 architecture as our autoregressive transformer with 4 attention heads per decoder layer and an embedding size of 64 (Radford et al., 2019). To optimize, AdamW with a learning rate of 5×10^{-5} and a linear warmup schedule with 1/10 of the total number of steps as warmup steps (Loshchilov & Hutter, 2019).

We ensure that on a validation similar to the training set, there is near-perfect performance by the completion of training.

A.4.3 VANILLA TRAINING

We find across setting that settings where ICL arises, there is structural ICL and it disappears abruptly with vanilla training. This is true for different levels of burstiness (0.8, 0.95, 1.0), different levels of ambiguity (0.05, 0.10, 0.20), and different distributions (Uniform, Zipf with $\alpha = 1.0001, 1.5, 2, 3$). In-weights learning varies based on the distribution.

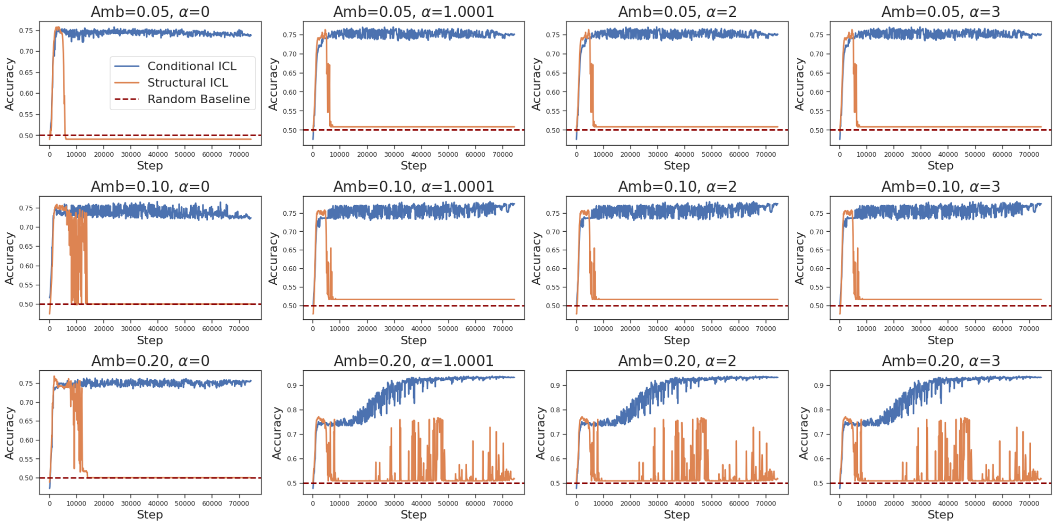


Figure 11: Structural ICL disappears while conditional ICL remains across different combinations of ambiguity and skew in our autoregressive few-shot task described in Appendix A.4. Interesting, skewed distributions with high ambiguities show some variance in structural ICL accuracy after the initial disappearance.

A.4.4 ACTIVE FORGETTING

Active forgetting preserves structural ICL, but completely removes any use of IWL. We see this across tested distributions (Uniform, Zipf with $\alpha = 1.0001, 2$). We use $k = 500$ because this worked well with initial experiments (although the other tested parameters of $k = 1000, 2000$ also worked almost equivalently).

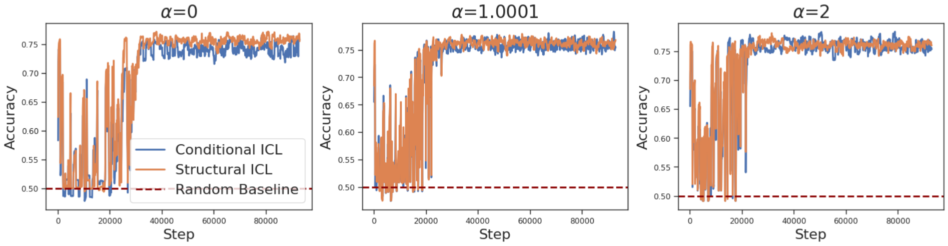


Figure 12: Active forgetting preserves structural ICL across different skews in our autoregressive few-shot task described in Appendix A.4. Interesting, increasing the skew seems to make active forgetting converge quicker.

A.4.5 TEMPORARY FORGETTING

In our temporary forgetting setting, we use a burstiness parameter of 0.95 for experiments. We use $k = 1000, N = 8000$ because these parameters worked well in initial experiments. We did not spend much time optimizing parameters. We tested whether we could evoke a dual process of ICL and IWL across distributions (Zipf with $\alpha = 1.0001, 2, 3$), as seen in Figure 13. This is in contrast to active forgetting, where we cannot learn information in-weights (Figure 14), and vanilla training, where we cannot asymptotically perform above a random baseline for structural ICL (Figure 11).

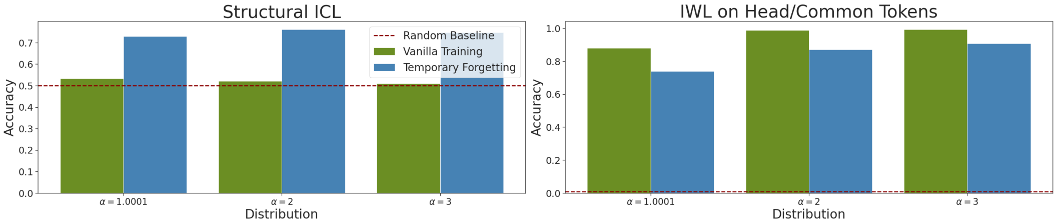


Figure 13: Temporary forgetting preserves structural ICL across different skews in our autoregressive few-shot task described in Appendix A.4, as opposed to vanilla training (i.e. standard training). In addition, it enables IWL for common tokens instead of completely removing it like active forgetting. It achieves about 90% the IWL use for these. Note we consider the smaller set between top 100 tokens and 90% of the probability when choosing common tokens to evaluate IWL on.

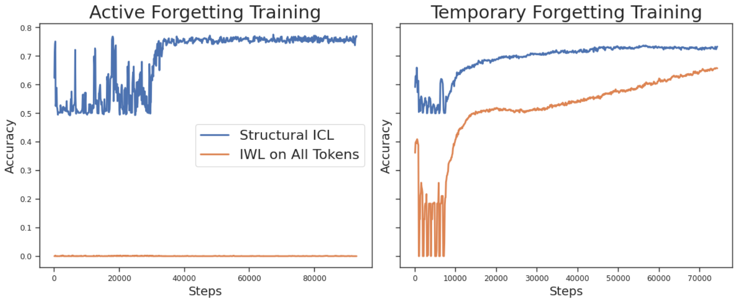


Figure 14: Temporary forgetting enables us to learn IWL while preserving structural ICL, whereas active forgetting forces only structural ICL. This is seen by the developmental accuracies in this figure (note $k = 500$ for active forgetting whereas $k = 1000, N = 8000$ for temporary forgetting).

A.5 DUAL PROCESSES FOR SKEWED DISTRIBUTIONS

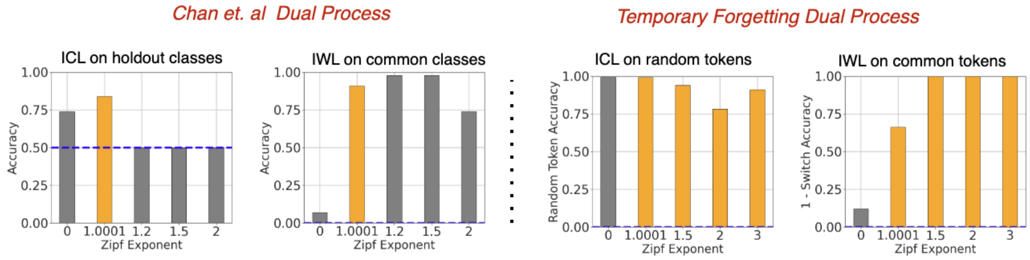


Figure 15: Temporary forgetting's ability to invoke dual processes (in yellow) on various distributions of our synthetic POS task compared with Chan et al. (2022b) observational baseline. Structural ICL and IWL are able to be co-occur in networks now trained on data distributions of any skew with $\alpha \geq 1$, as opposed to being limited to a specific "sweet spot" distribution.

A.6 PUSHDOWN DATASETS

We use the train/dev splits from the English UD Treebank for the *c-pos*, *f-pos*, and *dep* tasks McDonald et al. (2013); the train/dev splits from Ontonotes-v5 in the CoNLL-2012 Shared Task format for the *ner*, *phrase start*, and *phrase end* tasks Linguistic Data Consortium (2013); Pradhan et al. (2012); the train/dev splits from Penn Treebank-3 for the *depth* and *dist* tasks Marcus et al. (1993); and generated token sequences for the *prev*, *dup*, and *ind* tasks.

We reproduce baselines from Elazar et al. (2020) to verify the correctness of our probing setups for *c-pos, f-pos, ner, dep, phrase start* and *phrase end* and from Hewitt & Manning (2019) for *depth* and *dist*.

A.7 PUSHDOWN SIGNATURE OBSERVATION IN SYNTAX

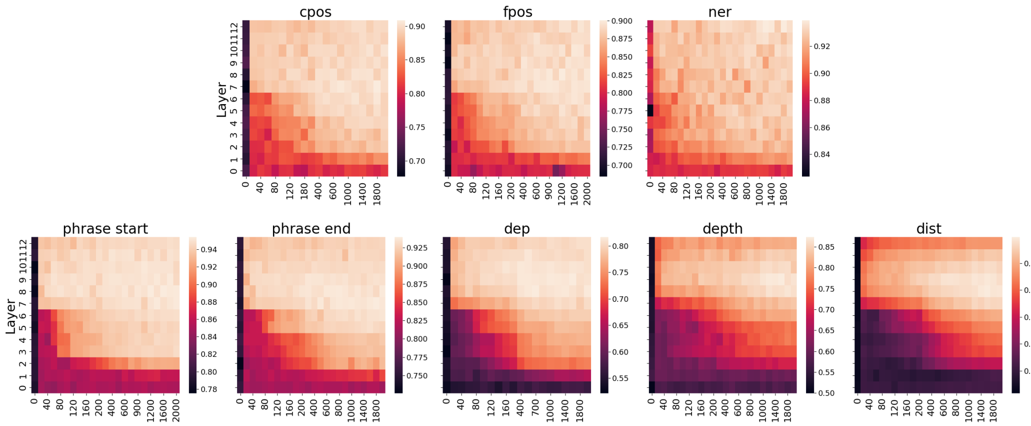


Figure 16: The "Pushdown Phenomenon" is observed across syntactic features, suggesting that a transition from IC to IW strategies happens across these features. In early steps of training, representing syntactic information occurs in later layers, which are more contextualized. However, as training progress, the same properties are better encoded in earlier layers due to memorization of token-level and n-gram level information. The n-gram level information requires attention to build, which explains why performance in *dep, depth*, and *dist* does not propagate all the way to embeddings.

The "Pushdown Phenomenon" suggests that in early steps of training, computing token-wise syntactic properties occurs in later layers, which have more in-context information. However, as training progress, the same properties are better encoded in earlier layers until only the first couple layers are required for representing syntactic properties.

We examine whether the ”Pushdown Phenomenon” exists in various syntactic properties in BERT. To do so, we employ our probing setup (Appendix A.1) for the tasks of named entity recognition (*ner*), coarse part of speech (*c-pos*), fine-grained part of speech (*f-pos*), dependency parsing (*dep*), syntactic constituency boundaries which indicate the start and end of a phrase (*phrase start, phrase end*), depth in the parse tree (*depth*), and distance in the parse tree (*dist*). We probe each property across the axes of (1) training time steps and (2) layers. We repeat this process for three seeds of the MultiBERTs (Sellam et al., 2021). For all tasks, we probed all layers of MultiBERT seeds 0, 1, and 2 for timesteps from 0 to 200,000 increasing by 20,000; 200,000 to 1,000,000 increasing by 100,000; and 1,000,000 to 2,000,000 increasing by 200,000. If a specific word is composed of multiple subword tokens, we follow Hewitt & Manning (2019) and average the encoding across tokens.

We observe the ”Pushdown Phenomenon” in all our examined tasks. However, we find that across tasks, syntactic information is ”pushed down” at different rates. Early layer accuracy increases approximately follow a pattern of *ner* → *phrase start* → *cpos/fpos* → *phrase end* → *dep* → *depth* → *dist*. We leave it to future work to explore whether this timing is a function of (1) complexity of high-achieving rules/heuristics consistent with Belrose et al. (2024) or (2) a naturally occurring dependency hierarchy of syntactic relationships suggestive of implicit curriculum learning. One possible intuition for why the ”Pushdown Signature” of memorization often coincides with poor maintenance of in-context strategies might be neural collapse (Parker et al., 2023; Rangamani et al., 2023), although this should be further investigated by future experimentation.

A.8 SYNTHETIC DATA GENERATION FORMULATION

Our synthetic data generation can be formally representated as a probabilistic context-sensitive grammar (PCSG). Mathematically, we parameterize our vanilla PCSG (without POS ambiguity) as follows:

$$\mathbf{G} = (N, \Sigma, P, S, \alpha, v)$$

where $N = \{S, Q, Q_N, Q_A, P_N, P_A\}$ is the set of nonterminal symbols, $\Sigma = \{N_{init}, A_{init}, N_r, A_r, C\}$ is the set of terminal symbols, S is the starting point (and notationally also represents sequence), and α, v characterize the sampling probability distribution of our terminal symbols. Our production rules P are

$$F \to \begin{cases} S\ Q_N\ P_N \\ S\ Q_A\ P_A \end{cases} \quad \text{with eq. prob.}$$

$$S \to \begin{cases} N_{init}\ C\ A_{init} \\ C\ A_{init}\ N_{init} \end{cases} \quad \text{with eq. prob.} \qquad Q \to \begin{cases} Q_N \\ Q_A \end{cases} \quad \text{with eq. prob.}$$

$$Q_N \to N_r \qquad Q_A \to A_r$$

$$P_N \to A_r\ A_r\ A_r \qquad P_A \to A_r\ N_r\ N_r$$

with terminal symbols sampled from

$$N_{init} \sim \text{Zipf}\left(\alpha, 0, \frac{v}{2} - 1\right) \qquad A_{init} \sim \text{Zipf}\left(\alpha, \frac{v}{2}, v - 1\right) \qquad C \to v$$

$$N_r \to N_{init} \qquad A_r \to A_{init}$$

N_{init} captures a specific token that corresponds to a token and all references to N_r use this token exactly, enforcing strict consistency.

Note our sampling distribution *Zipf* is a truncated Zipfian parameterized by the tuple (α, s, e) with a probability mass function of

$$\mathbb{P}(X = k) = \frac{k^{-\alpha}}{H(\alpha, e - s)} \text{ for } k = s, s + 1, \dots, e, \text{ where } H(\alpha, n) = \sum_{k=1}^{n} k^{-\alpha}$$

We select tokens for `<noun>` $\in \left\{0, 1, \dots \frac{v}{2} - 1\right\}$ and `<adj>` $\in \left\{\frac{v}{2}, \frac{v}{2} + 1, \dots v - 1\right\}$. Thus, given a particular vocabulary size v and Zipf parameter α, `<noun>` $\sim$ Zipf $\left(\alpha, 0, \frac{v}{2} - 1\right)$ and

`<adj>` $\sim$ Zipf $\left(\alpha, \frac{v}{2}, v-1\right)$. To add further control to this setting, we introduce the parameter ε to describe ambiguity in the solution - that is, a proportion of ε tokens in each of $n = 10$ bins grouped by probability mass do not have a fixed POS but instead may be a noun or adjective with equal likelihood.

Note that when $\alpha = 0$, this distribution degenerates into Unif(s, e) and when $\varepsilon = 0$, each token has a fixed identity.

A.9 SYNTHETIC POS TASK EXAMPLES

Before approaching more complex tasks, it is prerequisite to understand phenomenon in a controlled and adjustable environment. Here, we clarify the design of our synthetic POS task a bit. Our task is designed to 1) minimally emulate a subtask performed in language models: Part-of-Speech tagging while 2) controlling for various confounds. In particular (1) it does not allows heuristics based on token position and (2) is not deterministic based on the query.
Here are a couple clarifying examples (`<sequence>    <query>` → `<pattern>`):

1. (a) `is happy dog   dog` → `happy dog dog`
   (b) `dog is happy   dog` → `happy dog dog`
   Note that in this example, we show that using two templates rules out a simple position-based. If model assumes the noun occupies the 3rd position of the sequence, then in the second sentence, the model will believe `happy` is the noun and falsely predict a response pattern of `dog dog dog`.
2. (a) `dog is happy   dog` → `happy dog dog`
   (b) `dog is sad   dog` → `sad dog dog`
   Note that in this example, both queries are `dog`, yet the predicted pattern is different. Context is necessary for correct prediction.

A.10 TOY MODEL

We employ a 6-layer BERT model across the synthetic setting experiments. Experiments were performed with an MLM as less prior work has examined syntactic tasks with autoregressive models and structure is much more difficult to intuit in autoregressive models as they are only exposed to an ordered subset of the tokens in a sentence. This model has 1 attention head per layer, 64-dimensional hidden dimensions, 128-dimensional intermediate representations, and tied weights for the embedding and unembedding layers. We optimize model parameters with AdamW with a learning rate of 5×10^{-5} (Loshchilov & Hutter, 2019). We chose a thin and long representation to examine how representations evolve after each attention operation (for better granularity). The hidden dimension sizes were decided per a minimax strategy, i.e. this representation dimensionality was the smallest such that we achieved near perfect accuracy on a validation set for the downstream task. Future work should better examine the effect of representation size on in-context vs. in-weights learning.

A.11 PERFORMANCE BY TOKEN DECILE

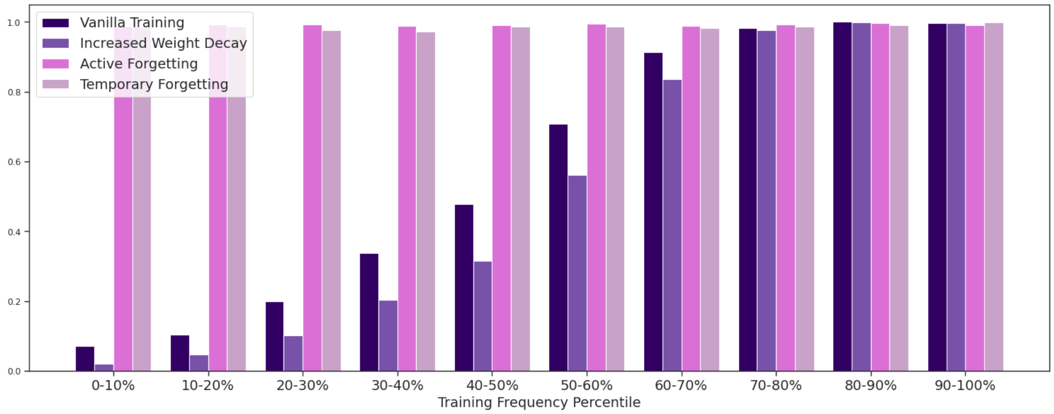
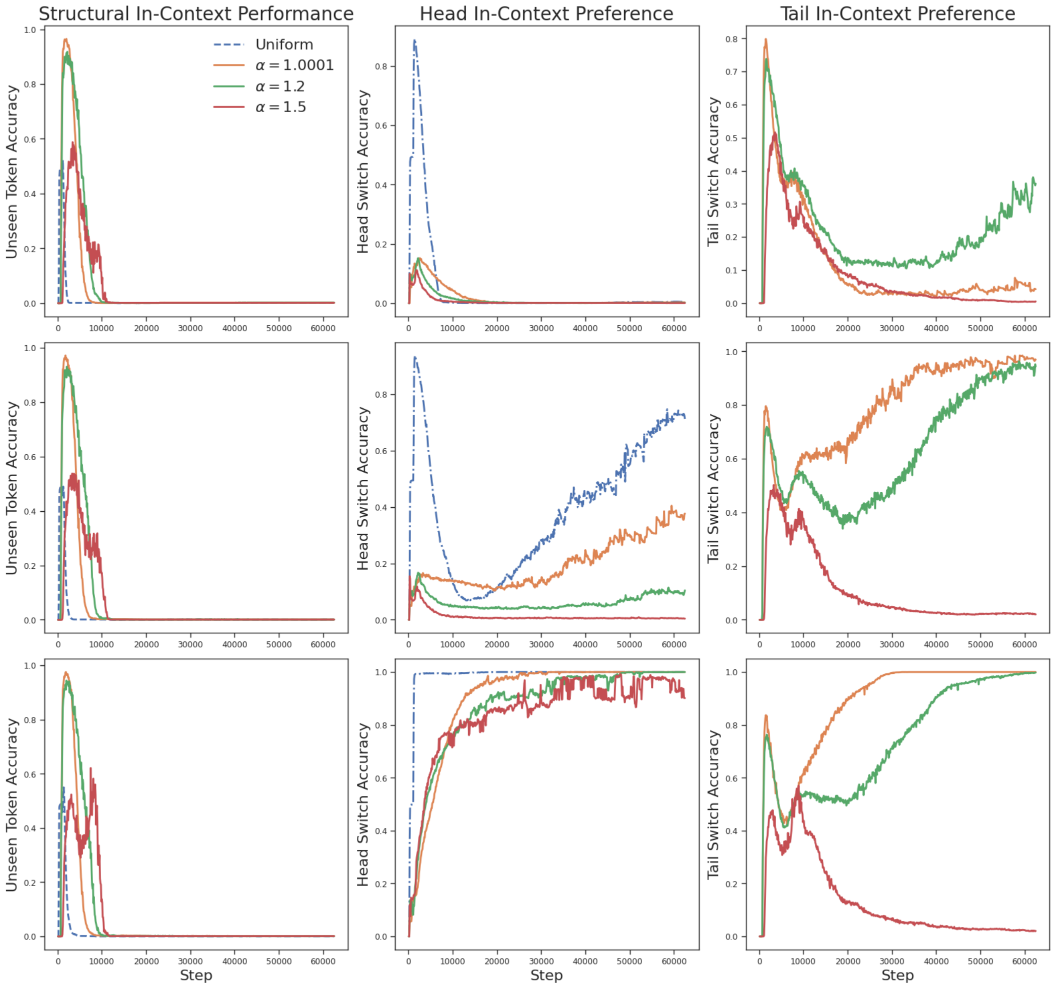


Figure 17: Increased weight decay has little/no effect on the failure of the structural ICL strategy (we increase weight decay from 0.01 to 0.1). In contrast, active temporary forgetting boosts rare token validation accuracy significantly, as seen in the tail of the distribution. Parameters are $v = 10000, \varepsilon = 0.10, \alpha = 1.5$

We find that on highly skewed distributions, the tail of the distribution suffers immensely due to undertraining. This phenomenon cannot be rectified by Singh et al. (2023)'s method of promoting asymptotic ICL. However, we find that both active forgetting and temporary forgetting correct this behavior to boost performance on tail tokens in skewed distributions from near-zero to near-perfect levels.

A.12 AMBIGUITY (ε) EXPERIMENTS



Structural In-Context Performance

Head In-Context Preference

Tail In-Context Preference

Figure 18: (Top) $\varepsilon = 0.01$, (Middle) $\varepsilon = 0.10$, (Bottom) $\varepsilon = 0.50$. Overall in-context strategy is dependent by amount of ambiguity in the labels. With 50% of the tokens as ambiguous, all unambiguous tokens use an in-context strategy; with 10%, there is a mixed strategy dependent on where in the distribution the example is; with 1%, almost unambiguous tokens use a memorized strategy. The vocab size is $v = 10000$

In all of our ambiguity experiments, structural ICL is transient (even whe 50% of tokens are ambiguous). The ambiguity parameter significantly alters the models overall strategy. With a low ambiguity parameter, the model prefers memorization (IWL strategy) of umambiguous tokens and with a high ambiguity parameter, the model prefers an ICL strategy. Across all ambiguity parameters, there is a difference in tail and head behavior.

A.13 VOCABULARY SIZE (v) EXPERIMENTS

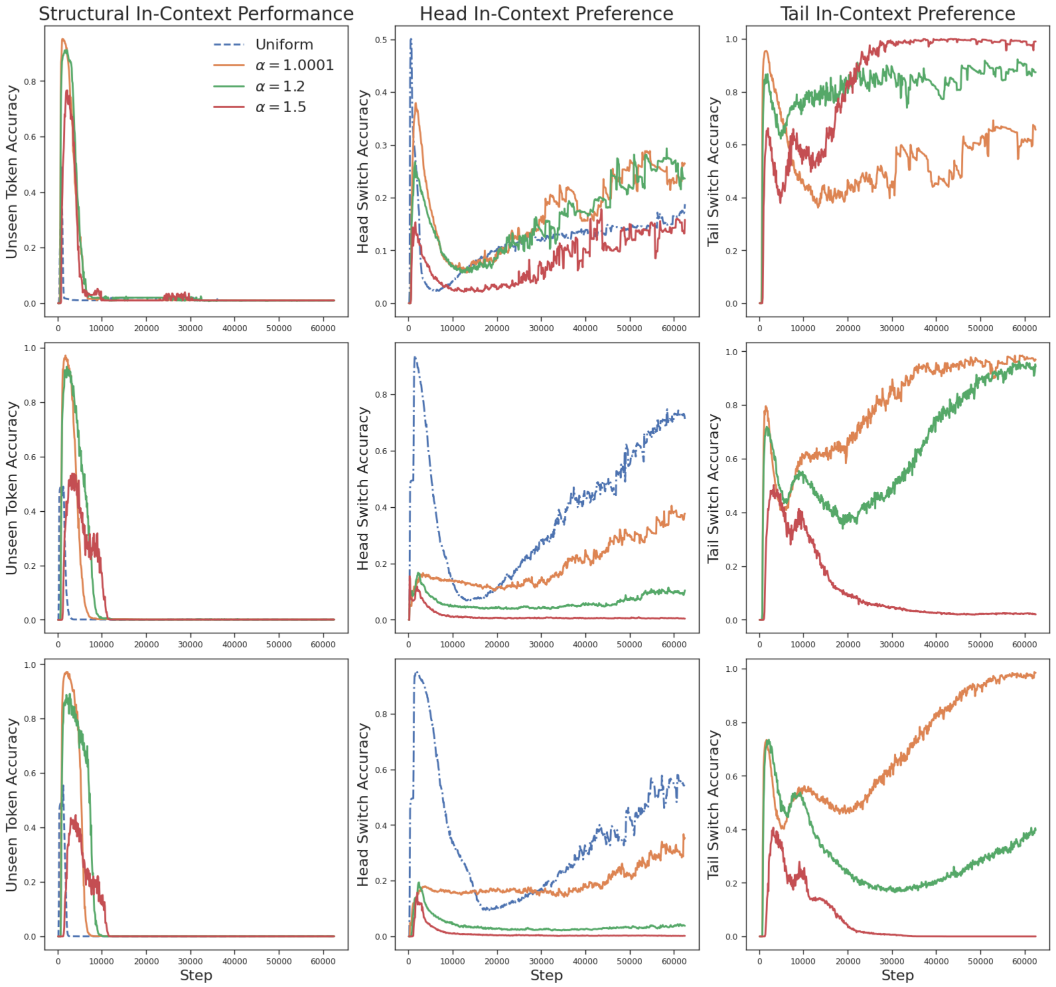


Figure 19: (Top) $v = 1000$, (Middle) $v = 10000$, (Bottom) $v = 20000$. The strength of an in-context solution depends on the interaction between vocabulary size v and skewedness of the distribution α. Too small of a vocabulary size (i.e. $v = 1000$) encourages more memorization in general but fixes performance in $\alpha = 1.5$ setting. The ambiguity is $\varepsilon = 0.10$.

In all of our vocabulary experiments, structural ICL is transient. As expected, we find that vocabulary size has a similar effect to the skewedness of the distribution. That is, increasing the vocabulary without bound would lead to poor tail ICL performance. Too small of a vocabulary size seems to increase ICL among very skewed distributions but decrease ICL among all other distributions.

A.14 EMBEDDING ANALYSIS

We perform qualitative analyses on the embeddings produced by vanilla training (i.e. standard training without modification), active forgetting, and temporary forgetting in order to better understand how these training regimens impact model representations. These analyses, consisting of principal component analysis (PCA) and probing for POS, are located in Appendix A.15.

After vanilla training, the learned embeddings cluster according to their POS, far from the distribution of randomly-initialized tokens. We train a linear probe on these learned embeddings, and find that it can almost perfectly partition nouns and adjectives. Note that the disappearance of structural ICL occurs at the same time as the probe achieves above-random POS probing (i.e. memorization).

As expected, we do not see any structure in the embeddings produced after active forgetting. As such, a linear POS probe trained on these embeddings never achieves above random chance throughout training. The embedding distribution looks quite similar to the random initialization distribution, indicating that no information has been encoded in these embeddings.

Finally, the temporary forgetting setting reflects aspects of both vanilla training and active forgetting; that is, the head of the token distribution learns to partition nouns and adjectives whereas the tail of the distribution does not learn any structure. The tail embeddings much more closely resemble the initialization distribution with temporary forgetting than with vanilla training. This results in a unseen token generalization in addition to memorized information.

A.15 PRINCIPLE COMPONENT ANALYSIS OF EMBEDDINGS

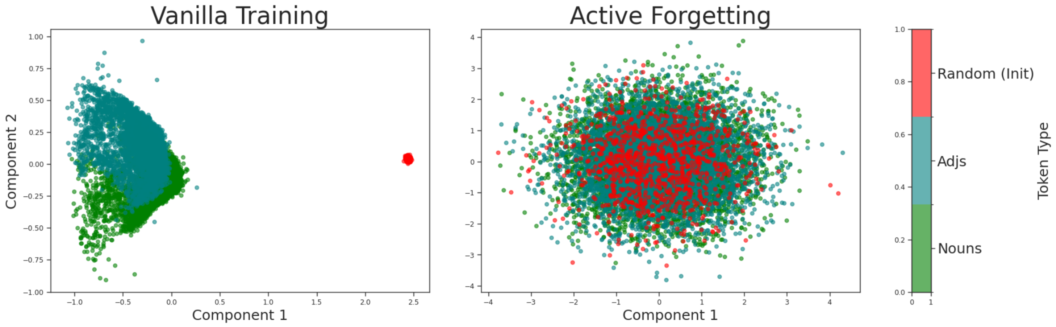


Figure 20: Vanilla training imposes structure on the adjectives and nouns such that randomly initialized (unseen) tokens are out-of-distribution whereas active forgetting embeddings resemble the initial distribution. Parameters used are $v = 10000, \alpha = 1.0001, \varepsilon = 0.10$.

We find that while vanilla training results in embeddings that lie on a manifold, active forgetting results in embeddings that look similar to the initial distribution. This helps motivate our use of temporary forgetting as we would like to preserve embedding structure. Moreover, note that in the above figure we use $\alpha = 1.0001$ and PCA whereas in Figure 1 (Bottom Right), we use $\alpha = 1.5$ and T-SNE. The tail tokens in the higher skew distribution see fewer gradient updates and thus resemble the randomly initialized (unseen) tokens more (in addition to T-SNE likely being a better visualization tool).

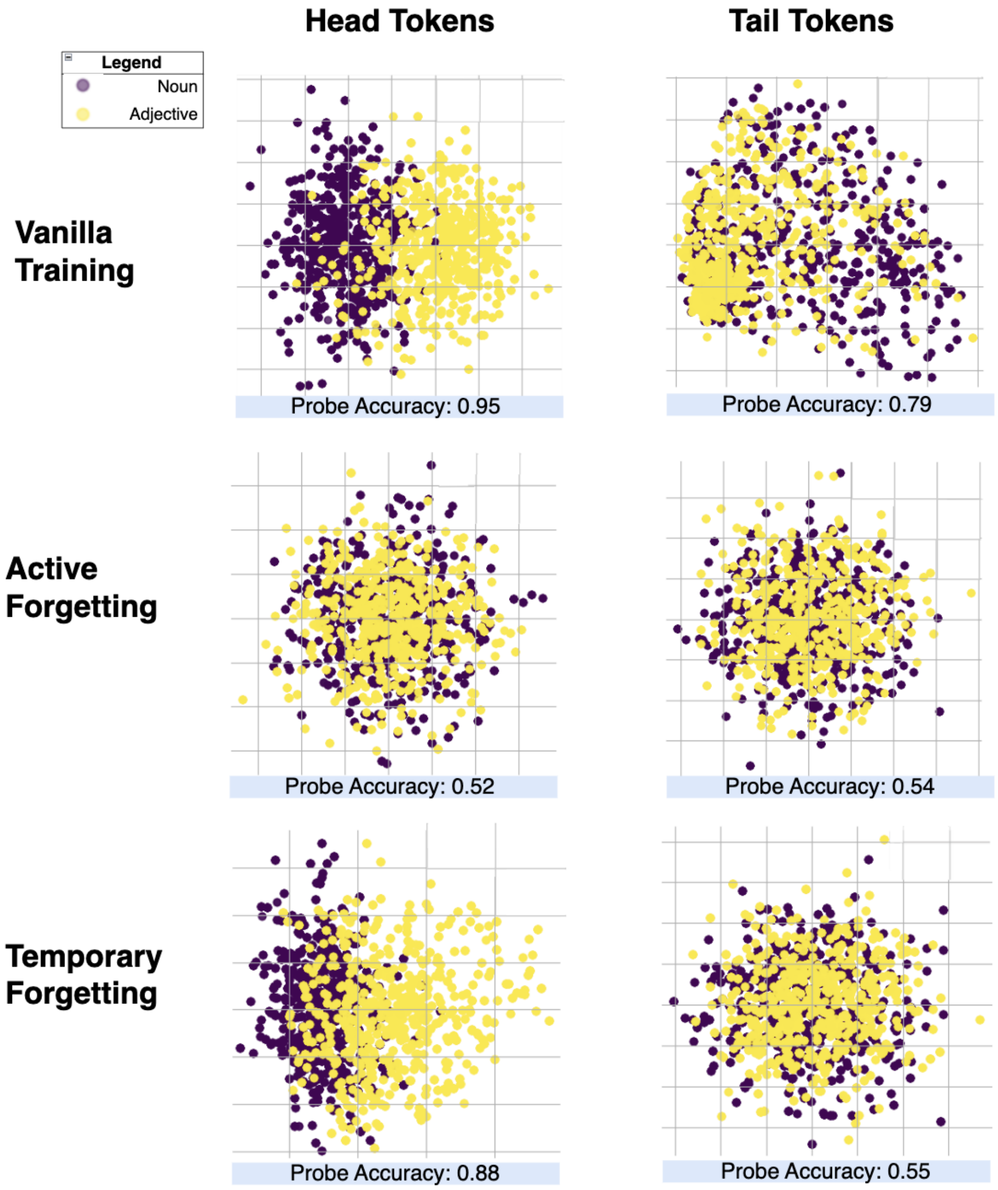


Figure 21: Vanilla training learns to partition noun and adjective embeddings in the head of the distribution, and some structure in the tail. Active forgetting learns no separation between noun and adjective embeddings. Temporary forgetting learns structure in the head of the distribution and no structure in the tail of the distribution. Parameters used are $v = 10000, \alpha = 1.2, \varepsilon = 0.10$.

A.16 OTHER RANDOM DISTRIBUTION GENERALIZATION

Note that while we define structural in-context learning as free from reliance on any *encoded semantic information*, it is important to note that this does not mean that structural in-context learning assumes *no* geometry of the space. In fact, this would be practically impossible to achieve because connectionist networks function in a geometric space and take advantage of orthogonality, translation, scaling, etc. If we cannot make assumptions about the distribution from which the data is sampled, then we deprive our networks of their toolbox. Still, we test on random sampling distributions for the embeddings other than our initialization distribution. Namely, we test on a uniform distribution from 0 to 1 and a large normal distribution with mean of 5 and standard deviation of 5.

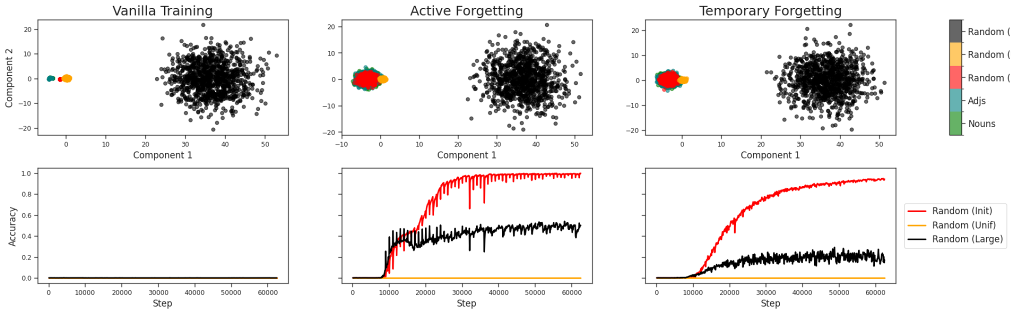


Figure 22: Vanilla training fails on all random tokens, whereas active/temporary forgetting succeed on the random distribution of initialization. Active and stop forgetting do not generalize to arbitrary random distributions, although show some generalization to normal distributions with large means and variances.

A.17 REQUIRED COMPUTE FOR EXPERIMENTS

We employed compute resources at a large academic institution. We scheduled jobs with SLURM. For our naturalistic experiments, each MultiBERT seed required 24 separate runs (one per tested checkpoint at a particular timestep), which totaled $\approx$ 100 hours on an RTX A5000 with 24 GB of GPU memory. Over 3 seeds, this was $\approx$ 300 hours of GPU usage. For our synthetic setting, the vanilla training required 64 separate runs (one per hyperparameter combination of vocab size, ambiguity, and sampling distribution), which totaled $\approx$ 250 hours of RTX A5000 usage. Likewise, our active forgetting and temporary forgetting interventions took a similar amount of GPU usage. Therefore, in total, our GPU usage for all synthetic experiments summed up to about 750 hours. We ran experiments mostly in parallel with SLURM to iterate quickly. Compute was a significant limitation for the development time and informed our development of training interventions in a synthetic setting. In total, our GPU usage was significantly higher than the reported number due to various failed/modified experiments. The total compute likely was around 20,000 GPU-hours on RTX A5000s, although this is a rough estimate.