FORKING

Sudden Overfitting Under Replay

Shaoyang Guo (Sean)

Shaoyang Guo (Sean) | Read-aloud script | 3,272 spoken words

Allow about 30 minutes at 110-115 words per minute, with pauses; actual delivery tends to run longer than these estimates, so plan cuts from the Contingencies note.

Read the body text only; headings and timings are navigation cues.

01 Introduction

TARGET CHECKPOINT 00:00 - 03:02

Thank you, Cecile, for the introduction, and thank you for having me. Hi, everyone.

My name is Shaoyang Guo, and you can also call me Sean. I'm a fourth-year undergraduate student at the School of Physics at Peking University, and I'll soon be starting my PhD with Ziming Liu at LiuLab. The work I'll share today was funded by MetaCircle, Ziming's research company. Also, please feel free to jump in and ask questions if I am not clear enough, or if you have anything you'd like to say.

Previously, I completed a one-year internship at ByteDance Seed, where I worked on post-training for vision-language models. I did pretraining, SFT, RL, and agent training on cutting-edge models. That experience gave me a much more complete picture of how large models are developed and trained, and it greatly satisfied my curiosity about how these systems work.

My research interests include self-evolving AI and the science of AI. Much of my motivation comes from wanting to understand how these systems are built and why they work. Actually, my ambition is to establish the science of AI, and we call it SAI: definite derivations between conclusions. OK, that's the grand ambition, and it's completely unimportant for the next thirty minutes. So let's get to our main topic.

Today, I'd like to share our recent work, "Forking: Sudden Overfitting Under Replay."

This work focuses on over-encoding, which is essentially an n-gram memory module. I'll explain how it works in a moment.

What we found is that, in our experiments, combining this module with repeated passes through the same data produces a distinctive pattern. At the start of a new epoch, training loss drops sharply, while validation loss can turn upward. As we repeat the data, the gap grows in a staircase pattern. The two curves split like a fork, so we call this phenomenon Forking.

Our main question is: why does adding this n-gram module make overfitting under replay so strong?

02 What is over-encoding?

TARGET CHECKPOINT 03:02 - 05:52

So let's look at how this n-gram module works.

The basic idea is simple. A standard language model has a token embedding table, which maps each token to a vector. Each Transformer block then adds an update to the residual stream.

Over-encoding adds another learned representation, this time for a short sequence of consecutive tokens, called an n-gram. A bigram contains two tokens, and a trigram contains three. The module looks up a vector for that local sequence and adds it to the residual stream.

To make the examples easy to follow, let's temporarily treat each word as one token. Take the phrase "there is a dog." Here, "there is a" is a three-token context, a trigram, and the token the model needs to predict is "dog." And already, with just this context, we know at least one thing: the next token should be a noun. And DeepSeek's Engram uses tables of up to 5-grams.

So the model receives both the ordinary token representation and an additional representation of the local phrase.

This is attractive because lookup is computationally cheap compared with applying a large dense neural network. The lookup can be done with CPU only, and the table can be stored in host memory, or even on SSD.

So this looks like a new scaling axis, an axis for scaling memory.

And I want to emphasize that our paper is not arguing that over-encoding does not work. It can work well. It can improve prediction and provide useful additional capacity.

What we want to point out is a failure mode that appears when over-encoding is combined with multi-epoch training. At the model sizes commonly used in research, it is very common. With large amounts of data, it becomes less obvious. But production-scale massive data still runs into this problem once it meets multi-epoch training.

03 The phenomenon: why we call it forking

TARGET CHECKPOINT 05:52 - 07:15

So what does this failure mode actually look like in training? Let me describe the central observation.

We found this phenomenon in the nanoGPT speedrun. We adopted an architecture with an n-gram module and trained it on a multi-epoch dataset, with the same backbone without the module as a control. At the beginning of the second epoch, the training loss drops like a staircase, and the validation loss diverges and goes up, like a fork. So we call it forking. In the third epoch, the staircase goes up and down, and the gap gets larger again.

This sounds like ordinary overfitting. But usually, LLMs do not overfit. The data is simply too massive, and the patterns in natural language are too complex, so LLMs usually sit in an underfitting regime. These days we do see overfitting even on extremely large MoEs. Over-encoding is similar, except that it looks more violent.

04 Three empirical patterns

TARGET CHECKPOINT 07:15 - 08:44

Why does this happen? We ran three experiments and found three patterns. First, the gap grows as the context frequency drops, approximately following a power law over the range we measured. Second, the gap grows as the n-gram table gets larger. Third, the gap shrinks as the dataset gets larger.

The mechanism behind this can be understood as simple overfitting. If a context appears only a few times, say "dog" follows "there is" just twice and "cat" never appears, then the model overfits to "dog." If the context appeared a thousand times, the observed counts would be much closer to the prior distribution, say dog to cat one to one.

When the table is smaller, different contexts stack onto the same entries, so the over-encoding effect weakens. It is equivalent to each vector being accessed by more contexts, so it effectively sees more samples. And when the dataset is larger, rare contexts become fewer, and the gap gets smaller.

05 Novel continuations and probability crowding-out

TARGET CHECKPOINT 08:44 - 12:02

So those are the three patterns. But a pattern is not yet a mechanism. The next clue comes from what we call novel continuations.

A novel continuation is not necessarily a new word. It is a continuation that never followed a particular context in the training corpus, even though the context itself appeared there.

Let's use a deliberately simplified example. Suppose "there is a" appears twice in our training sample, and both times it is followed by "dog."

Now imagine that, in the underlying distribution for this example, "dog" and "cat" are equally likely. Our small sample simply happened to contain two dogs and no cats.

The model may have seen the word "cat" many times elsewhere. What it has not seen is this particular context followed by "cat."

After the first pass, it may still assign some probability to "cat," using knowledge learned from other contexts.

But then we replay the same two examples. Both targets are still "dog." The training objective keeps rewarding the model for assigning more probability to "dog."

The probabilities of all possible next tokens must sum to one. As probability becomes increasingly concentrated on the observed continuation, alternatives can lose probability. We call this crowding-out.

When a held-out example contains "cat," the model now predicts it less well. Training loss improves, but validation loss gets worse.

The key point is that replay does not fill in the missing evidence. It reinforces the evidence already present.

So the model can become more confident without becoming better informed.

This is more specific than saying the dataset is biased. A finite sample can miss valid alternatives by chance, even when it comes from the right underlying distribution. Repeated optimization can then turn that sampling limitation into increasingly confident predictions.

And this is where the power law from earlier comes back. That frequency dependence is part of why this is not simple overfitting. Plain overfitting would not naturally produce this kind of power law; crowding-out over novel continuations does.

Our aggregate measurements support this account: under replay, the memory model becomes worse at predicting novel continuations, while the no-memory control improves on them.

06 Why the frequency dependence is not simply one over square root N

TARGET CHECKPOINT 12:02 - 14:55

Now there is one more thing to explain: the power law itself. If this comes from limited samples, we might expect the gap to decrease as one over the square root of the sample count. But in our earlier experiments, the fitted decay exponents were about zero point two five for bigrams and zero point three three for trigrams.

The exact numbers are not the main point. They were a clue that triggered our investigation, not universal constants or decisive evidence for the mechanism.

The relevant quantity is how much probability belongs to continuations that have not yet appeared in the sample. This is called missing mass.

At first, I suspected this power law might come from the training dynamics somehow. But I never found a good model for that. Later, it occurred to me that it might be an intrinsic power law of natural language itself.

So I explored this together with AI, and that is how I found the Good-Turing distribution. Good-Turing is essentially a power law that comes from the natural-language distribution, and it is exactly about novel continuations. It says: given that an n-gram context has already been seen f times, the probability that the continuation is novel at the next sample is proportional to f to the minus alpha.

And we found this alpha is approximately 0.25 for bigrams and 0.38 for trigrams, close to the gap exponents we measured. To be honest, the precisely measured coefficients do not match the gap's power law exactly, because the gap also carries higher-order contributions. Not only novel continuations have this bias; rare continuations have it as well. That needs a more careful treatment, especially for higher-order n, where the effects become more complex. We will not expand on that here. But the broad picture holds: the frequency dependence of the gap roughly follows that of novel continuations.

07 Why does the memory module amplify the problem?

TARGET CHECKPOINT 14:55 - 19:42

So incomplete coverage, plus crowding-out under replay, explains the patterns we measured. But there is still an important question. Both models see the same incomplete data. Why does adding an n-gram table make the problem much stronger?

Our explanation is that the table makes learning more local to individual contexts.

Consider "there is a" and "that is a." Both end with the token "a." Without the extra memory, their initial token embedding for that final token is shared, although the Transformer can still distinguish the phrases through attention.

With over-encoding, the two phrases receive different learned memory vectors from the very beginning. You can picture those vectors as context-specific offsets in the model's representation space.

And here I want to give you a sense of the magnitude. In our experiments, you only need to add the trigram table. Recall the earlier picture: the whole Transformer is really just the initial token embedding, plus the residuals added by many blocks. Now we add one layer of over-encoding, and we project the over-encoding vector onto the total representation, meaning over-encoding plus everything the Transformer adds. The over-encoding component already accounts for about 77 percent of the norm, roughly three quarters. This explains why over-encoding can work, and also how strong it is: the over-encoding only needs a small angle, and the starting point of the whole Transformer computation becomes completely different.

What does that mean? Consider "there is a dog" and "that is a dog." Without over-encoding, the Transformer's starting point for predicting what follows the final "a" is the same in both cases. So the model can learn one shared prior for what comes next, say dog to cat about one to one, and use it in both contexts. But with over-encoding, the starting points for "there is a" and "that is a" are already different. Then the vectors used to predict "dog" after "there is a" and after "that is a" point far apart, and they enter different distributions. So they are very likely to be independent subspaces. That is our picture, and in a moment we will verify this independence.

This means that when we train on "there is a" followed by dog or cat, even if we have seen many dog and cat samples, and even if we know the distribution after "that is a" should be similar, the knowledge does not generalize across. So the memory module introduces a failure mode here: unlike the traditional Transformer, which generalizes so well, its generalization is clearly very weak.

That is what we mean by weakly interacting context-specific subspaces.

We do not mean that the network literally breaks into disconnected pieces. The backbone is still shared. And different representation vectors, by themselves, do not prove that learning is independent.

This geometric picture is an intuition. To test the mechanism, we need an intervention on training itself.

The central prediction is straightforward: if most of the extra gap is generated by direct training on each context, then stopping those local updates should largely stop its additional gap growth, even while the rest of the model continues learning.

08 Testing local learning by masking the loss

TARGET CHECKPOINT 19:42 - 22:55

That brings us to one of the most decisive experiments.

We randomly selected two percent of the training trigram contexts. Starting from a replay boundary, we stopped applying the next-token loss at positions belonging to those contexts.

The text remained in the input, and all model parameters remained trainable. We did not remove the selected phrases from everything the model could see. We only removed the loss terms that directly trained those prediction positions.

Meanwhile, training continued on the other contexts.

For evaluation, we measured the loss distribution of the masked and the unmasked contexts on the validation set, and we also measured the probability mass on seen and unseen continuations. Everything was measured on fixed probes, so we could compare the same examples as the model changed.

The result was striking. The masked contexts' average additional gap growth almost stopped. The unmasked contexts continued to develop a larger gap at replay boundaries.

In one paired comparison, the selected contexts gained about one nat of gap under normal training, but only 0.007 nats when their local loss terms were masked. A nat is simply the unit of this loss; the important comparison is about one versus almost zero.

So learning on the other ninety-eight percent of contexts did not reproduce the gap growth caused by direct learning on the selected contexts themselves.

For unmasked contexts in the memory model, replay increased the concentration on seen continuations. For the masked contexts, that additional separation was largely arrested.

There is a useful distinction here. Individual predictions can still move because the shared network is changing. What nearly stops is the average growth of the training-validation gap. The gap that already existed does not disappear.

The no-memory control gives an informative contrast: its selected contexts could still gain support for continuations through learning on other contexts. That is consistent with stronger sharing of useful information, rather than treating every local context as a nearly separate problem.

This gives us direct intervention evidence for local generation of the gap, rather than just a correlation between rare contexts and poor generalization.

09 Why this matters beyond a small experiment

TARGET CHECKPOINT 22:55 - 25:35

Now that the mechanism is supported by direct intervention evidence, let me turn to the practical relevance.

Why might an architecture look successful during large-scale pretraining, yet reveal this problem in a replay experiment?

There is no contradiction. A large and diverse stream of mostly fresh data can provide much better coverage. Repeatedly optimizing a small fixed corpus is a different regime.

Even within one training run, a smaller subset might be replayed more heavily than the rest. A gap concentrated in that subset or in rare contexts could be diluted in an overall average. This is one reason to inspect more than a single aggregate training curve.

The issue becomes particularly relevant when we adapt a model to a smaller dataset. And I would say this failure mode may have a big impact on mid-training and SFT, because both of them replay limited data.

In our continued-training experiments, we started from a checkpoint with a relatively small gap, then trained further on smaller corpora for five passes. These included same-distribution text, code, and mathematics. With trainable memory, the gap reopened on all three.

In post-training, literally SFT and RL, we also performed experiments. For the RL side, we used a batch of RL trajectories and did SFT on them, as a substitute for online RL. Concretely, we tested a heavily repeated mathematics dataset associated with reinforcement learning with verifiable rewards. Here I want to be precise: this was a static data-reuse proxy, not an on-policy reinforcement-learning run.

Repeating a prompt while generating genuinely new responses is not the same as replaying an identical context-continuation pair. So these results motivate studying real RL pipelines; they do not establish that every PPO or GRPO run will fork.

10 What can we do about it?

TARGET CHECKPOINT 25:35 - 28:40

Finally, how can we reduce the harm?

One intervention follows directly from the frequency result: suppress memory retrieval for rare contexts.

This is different from the loss-masking experiment. Here, we keep the training examples and their loss terms. We only set the retrieved n-gram vector to zero when the context frequency falls below a chosen threshold.

The model still learns from those tokens through the shared backbone. It simply cannot rely on that particular local memory entry. We apply the same retrieval rule during training and evaluation.

This is not an argument for throwing away rare training examples. Those examples may contain valuable information. The intervention changes how the model represents them, while preserving the opportunity to learn through shared parameters.

In our experiments, this substantially reduced the gap.

For example, in the compact DeepSeek-style replication, masking reads for contexts seen at most one hundred times reduced the final gap from about 3.32 to 0.82 nats. Validation loss itself improved by about 1.60 nats.

That second measurement matters. A smaller gap alone could come from making training loss worse. Here, held-out prediction also improved.

Another natural idea is to freeze the memory table. That can help, but it does not guarantee a solution.

Freezing stops the entries from changing. It does not remove the context-specific representations already stored in them. The backbone can continue learning how to exploit those representations and fit the replayed examples.

In our continued-training tests, freezing often reduced the gap but did not eliminate it.

More broadly, we should consider memory capacity, n-gram order, and data coverage together. Increasing the n-gram order means using a longer local context, which can make each exact context more sparsely observed. A larger table can make those contexts more independent.

The design challenge is to preserve useful local memory without making sparsely observed contexts too easy to memorize.

Low-frequency retrieval masking is a promising intervention, not a complete solution. We still need stronger downstream evaluations and tests under realistic production training conditions.

11 Closing

TARGET CHECKPOINT 28:40 - 30:21

Let me close with three points.

First, over-encoding is a useful architectural idea, but under repeated training on limited data, it can amplify a sharp separation between training and validation performance. That is the phenomenon we call forking.

Second, our explanation combines incomplete continuation coverage with local, replay-driven sharpening. The model repeatedly reinforces the continuations it has seen, while plausible alternatives lose probability. Context-specific memory makes those small prediction problems more independent.

Third, reducing access to rare-context memory can help, but freezing the table alone is not a general guarantee. We need to evaluate memory mechanisms together with the data regimes in which they will actually be used.

For me, this is part of the science of AI: not just finding a method that improves a score, but understanding why it works, when it fails, and which experiments can distinguish between explanations.

There is still a great deal to investigate, and I plan to keep working on this problem.

Thank you to my co-authors, and thank you, Cecile, for hosting. Thank you everyone for listening. I'm looking forward to the discussion.

Questions for Cecile

OPTIONAL HOST GUIDE | Not part of the 30-minute presentation

Ten main questions plus three backups; suggested answers in Forking_QA_Sheet_for_Cecile.md, speaker prep in Forking_Focused_QA_Draft.md. Suggested thread if time is tight: 1-6. The speaker especially welcomes 9.

Main questions

1. When did you first realize that forking was more than just ordinary overfitting?

Follow-up: Which observation was hardest to explain?

2. Could you explain the dog-and-cat example again in simple terms?

Follow-up: How can a familiar word still be a "novel continuation"?

3. Does this happen only with your particular training settings, or more generally?

4. What does this mean in practice, for fine-tuning and reinforcement learning?

5. Does this really have an impact on production-scale models?

6. What is the next experiment you most want to run?

Follow-up: How will you know whether a fix really works, without losing the benefits of the memory?

7. You said your ambition is to establish the science of AI. What exactly does that mean?

Follow-up: How do you want to develop this idea over the next few years?

8. What is your view on innovation in language-model architectures?

9. Do we need a complete theory of AI before we can improve it?

10. Could AI itself help us discover the science of AI?

Backup questions

11. How do you know when an explanation is convincing?

12. Can researchers with limited compute still do useful AI research?

13. What makes a research problem worth pursuing for you?

Speaker notes

REFERENCE ONLY | Do not read aloud or forward as part of the host question list.

Timing

The spoken script is approximately 3,272 words. At 110-115 words per minute, plus brief pauses, allow roughly 29-31 minutes; the checkpoints run to about 30 minutes. The speaker's actual delivery runs about 1.3 times these estimates (a passage scripted for 12 minutes took about 16), so treat the checkpoints as lower bounds; paragraphs can be cut freely using the cut lines in Contingencies. Timings are rehearsal checkpoints, not a guarantee. Headings, source notes and this appendix are not spoken.

Name and title

Use Cecile, as confirmed by the host. The published paper title is Forking: Sudden Overfitting Under Replay. The speaker biography follows the details confirmed during this rehearsal.

Frequency exponents

The spoken script (Section 06) quotes earlier-setting measurements confirmed by the author: gap decay exponents of about 0.25 (bigram) and 0.33 (trigram), and Good-Turing novel-continuation exponents alpha of about 0.25 (bigram) and 0.38 (trigram). These are not the published v1 values (bigram gap about 0.25 and novelty about 0.23; Figure 5 trigram gap fit about 0.47, from the later setting). Present the exponents as the clue that triggered the mechanistic investigation, not as universal constants or decisive evidence. Higher-order contributions and the rare-continuation bias are now mentioned briefly in the script but must not be expanded orally; do not narrate the setting-change history unless asked. If a listener spots the mismatch with the public paper, use the prepared answer in Contingencies.

Masking experiment

The published selective loss-masking cohort is 2% of training trigram contexts, not 1% of tokens. Its additional fixed-probe gap growth changes from 1.018 to 0.007 nats. Inputs and all parameters remain trainable. Do not describe this as freezing the selected contexts or proving complete independence.

Representation diagnostic

Section 07 now states the projection diagnostic per the speaker's round-11 instruction: with the trigram table alone, projecting the over-encoding vector onto over-encoding plus the Transformer's contribution, the over-encoding component accounts for about 77 percent

of the norm, roughly three quarters. This is the authors' own measurement and was not found in the checked public v1; if asked for a source, present it that way. Do not claim a fractal geometry of Transformer representations.

Contingencies

Prepared 2026-10-21 with the speaker. Reference only; not spoken.

Bio: the full introduction is read even when the host has already introduced the speaker. Compressed fallback if time runs short: "As Cecile mentioned, I'm Sean, a fourth-year undergraduate at the School of Physics at Peking University, soon starting my PhD with Ziming Liu at LiuLab. This work was funded by MetaCircle, Ziming's research company." Then jump to the "Today, I'd like to share..." paragraph.

Greeting: the opening now includes "thank you for having me". If the host's introduction was very detailed, acknowledge it in one sentence, then continue with the full bio as planned.

Exponent mismatch: if a listener points out that the public v1 paper reports different fits (for example, a trigram gap fit of about 0.47), answer honestly and briefly: the quoted 0.25/0.33 and 0.25/0.38 come from earlier experiments; the setting was later changed and the published fits differ, with larger deviations from a pure power law; the qualitative story, that the gap's frequency dependence tracks novel-continuation coverage, is unchanged. Do not volunteer this unless asked.

Table learning rate: if asked whether the high table learning rate drives the effect, answer that it was chosen to expose the phenomenon within a short budget; the phenomenon was robust across the settings tested (host question sheet, main-list Q3), while the rate changes how fast and how strongly it develops.

Timing: checkpoints run to about 30 minutes at the rehearsal pace, but actual delivery runs about 1.3 times slower (a 12-minute passage took about 16), so plan cuts in advance. If behind schedule at Section 09, the compressible blocks are the RLVR-proxy precision paragraph in 09, the freeze discussion in 10, and the third point in 11; protect Sections 03, 06, 07 and 08, which carry the mechanism narrative.

Anonymity: never reveal or confirm which companies approached the speaker about collaboration, including whether it was DeepSeek; frame practitioner observations as "my understanding"; share no IP-sensitive details.

Evidence and scope

REFERENCE ONLY | Checked against the public paper and the rehearsal.

Hardware and commercial-model claims

Host-memory storage and prefetching concern the lookup pathway during inference; the whole module is not necessarily CPU-only. The main script does not claim to know why a commercial report did not discuss forking. The production-scale answer (main-list Q5) offers the speaker's speculation (single-epoch trainable phases, table-scaling headroom, effects diluted in aggregate averages) plus personal-communication context; frame it as speculation, keep it unattributed, and share no IP-sensitive details.

Experimental scope

The main controlled setting uses an eight-layer Transformer, a 768-dimensional hidden state, separate bigram and trigram tables, and table learning rates 128 times the backbone rate, with different optimizers. The setting was selected to expose the phenomenon within a short budget, not optimize production validation loss. The DeepSeek-style replication is a compact from-scratch model; its no-memory comparison did not complete stably under the same settings.

Mitigation and RL

Low-frequency retrieval masking is distinct from loss masking. In the DeepSeek-style replication, masking contexts with count at most 100 reduces the gap from 3.316 to 0.817 nats and improves validation loss by 1.600 nats. Freezing can mitigate but does not guarantee eliminating forking. The RLVR-like experiment uses a static, all-token language-model objective, not on-policy PPO or GRPO.

Source map

Primary paper: Shanbin Yu, Shaoyang Guo, Haoran Zhao, Danni Yu, and Ziming Liu. Forking: Sudden Overfitting Under Replay. arXiv:2610.00394v1 (2026).

`https://arxiv.org/html/2610.00394v1`

Script 3: Sections 2.1-2.2 and Appendices B.2/C.2. Script 4: Section 3.1. Scripts 5-7: Sections 4.1-4.3 and Appendix E. Script 8: Figure 9 and Appendix F.2. Script 9: Section 5 and Appendices G/H. Script 10: Section 3.2 and Appendix G.4. Script 11 synthesizes these findings.

Architecture reference: Xin Cheng et al. Conditional Memory via Scalable Lookup: A New Axis of Sparsity for Large Language Models. arXiv:2601.07372v1 (2026), Sections 2.1-2.5. Supports the distinction between sparse lookup, host-memory inference, and neural fusion.

https://arxiv.org/html/2601.07372v1