

Lecture 4: Data Repetition and Scaling

Allen Liu | CSCI-GA.3033-141

Fall 2026

1 Why does data repetition matter?

We want to train on a lot of data, but common data sources have many issues. One key issue is *duplication*. Conventional wisdom says that learning from many independent examples generally helps us train a better model, but *repeated examples do not give us new independent observations*. Crucially, the most common source of data, web scraping, makes repetition particularly common. Articles are republished across sites, pages are collected in successive crawls, and many pages share templates, navigation text, or legal notices. Different URLs therefore need not provide different training content. Small edits can also create near-duplicates that survive exact matching.

These copies consume training compute and give frequently reproduced material disproportionate weight. A large token count can consequently overstate the diversity of the training data. There have now been many papers finding substantial duplication even in processed language-model corpora; and showing that deduplication reduces memorized outputs and the training steps needed to reach comparable or better accuracy. See, for example, Lee et al. [1].

In this lecture, we'll focus on intuitively understanding the effects of repeated data on model performance. We will see that findings and scaling laws in simple mathematical settings allow us to make predictions about the effects of repeated data in more complex LLM training settings. Concretely, we study: *at a fixed training budget, how does the concentration of repeated data affect prediction on new data?*

2 An intermediate repeat count can be the most damaging

As mentioned, repeated data can damage language model performance. Naively, we expect that as data is repeated more times, the degradation gets worse. While this is true in many regimes, we can actually study the effect of data repetition in controlled settings and uncover some surprising and perhaps counterintuitive principles.

Chudnovsky et al. [2] train language models with 90% of tokens from unrepeated documents and 10% from a pool replayed r times. Each curve fixes model size and total compute. Increasing r shrinks that pool. Held-out loss peaks at an intermediate r , then partially recovers; the peak can be broad.

The comparison changes two things together: each repeated document receives more weight, while fewer distinct documents share that weight. These competing changes are the key to the model below.

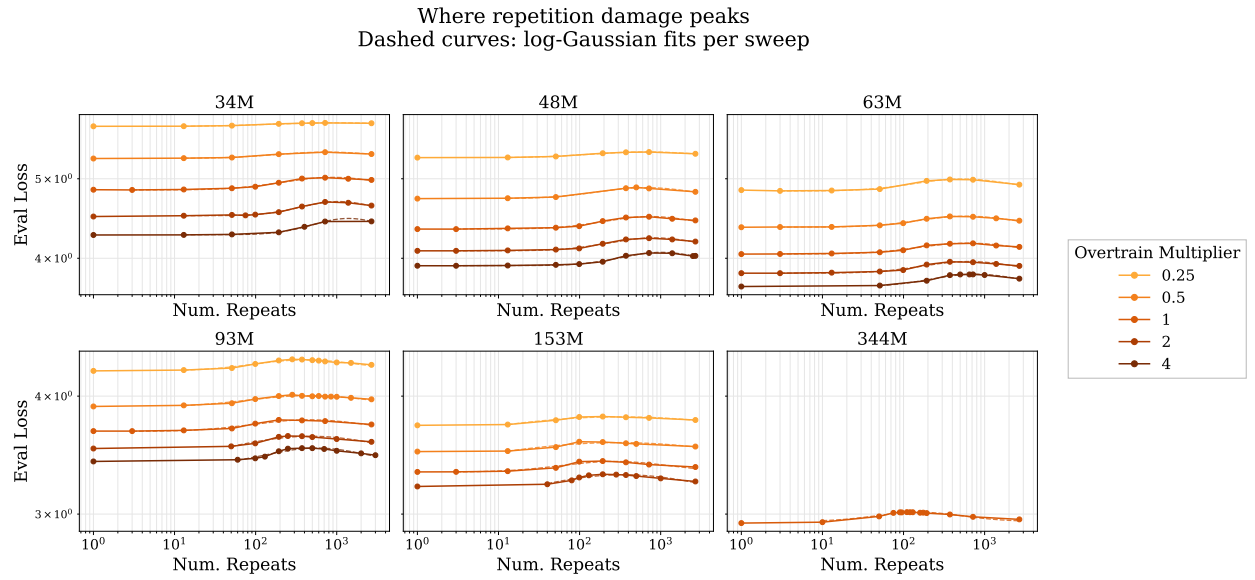


Figure 1: Reproduced from Figure 2 of [Chudnovsky et al., *Internal Data Repetition Destroys Language Models* \(2026\)](#), under [CC BY 4.0](#). Qwen3-style models on FineWeb-Edu-Dedup. Points are measurements; dashed curves are fits. Colors specify training-token budgets: $T = 20 \text{ OT } N$, where N is parameter count. Compare points along one curve.

3 From loss curves to practical predictions

The paper fits $r_{\text{peak}} \propto N^{-0.96}$ over its measured range: larger models peak at fewer repeats of larger pools. This empirical fit identifies potentially damaging configurations. Its regression analysis supplies a mechanism, rather than deriving this language-model exponent [2, Section 4.2].

Expressing a loss penalty in compute units

Suppose runs without repetition obey a fitted curve

$$L_0(C) = L_\infty + AC^{-\gamma}, \quad A, \gamma > 0.$$

Solving for the compute needed to reach a measured loss $L > L_\infty$ gives

$$C_{\text{eq}}(L) = \left(\frac{A}{L - L_\infty} \right)^{1/\gamma}.$$

Compare this with the actual compute used by the repeated-data run:

$$\text{compute-equivalent gain} = \frac{C_{\text{eq}}(L)}{C_{\text{actual}}}.$$

A ratio below one means that the reference curve reaches the same loss with less compute. This converts a prediction-error difference into a resource comparison, using the scaling methodology from lecture 3.

344M model, OT = 1

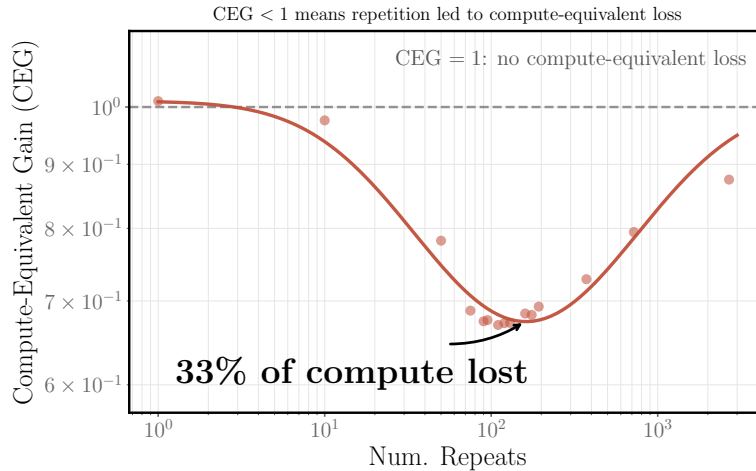


Figure 2: Reproduced from Chudnovsky et al. [2], Figure 1, under CC BY 4.0. The minimum ratio is about 0.67: equivalent loss at roughly two-thirds of the compute on the fitted reference curve. This is a fit-based estimate.

We now turn from measuring the penalty to explaining its shape. The model keeps exact copies and a limited learner, so we can track how repeated examples change the fitted prediction rule.

4 A regression model with exact copies

It turns out that the repetition effects discussed above are not specific to LLMs. We can actually develop intuitions about this phenomena through a simple mathematical model.

Give the learner d observed features, while allowing the target to depend on an additional feature it cannot use. This is the misspecified regression model of Chudnovsky et al. [2, Section 4.4 and Appendix D], written with one hidden Gaussian coordinate. Draw

$$x \sim \mathcal{N}(0, I_d), \quad z \sim \mathcal{N}(0, 1), \quad y = x^\top \theta_* + \sigma z,$$

with x and z independent. The learner predicts $x^\top \theta$ using only x . Labels are determined by the full input (x, z) , but the model cannot use the hidden signal σz . Its best population coefficient is θ_* , with loss σ^2 .

Hold the exposure budget fixed

Draw n_u independent originals that appear once and a separate pool of n_r independent originals that each appear r times. Fix the total exposure budget T and the fraction $\rho \in (0, 1)$ allocated to the repeated pool:

$$n_u = (1 - \rho)T, \quad n_r = \frac{\rho T}{r}, \quad T = n_u + r n_r.$$

Use integer pool sizes. Increasing r shrinks the repeated pool. Here T counts example exposures. Copying an example copies its features, label, and hidden residual. Set $w_i = 1$ for unrepeated originals and $w_i = r$ for repeated ones; the pool assignment is independent of the examples' values. Stack the $n = n_u + n_r$ distinct feature vectors into $X \in \mathbb{R}^{n \times d}$ and write $y = X\theta_* + \varepsilon$. Conditional on X , the residual vector has covariance $\sigma^2 I_n$. An original has an independent residual; all its copies share that same residual. Define

$$W = \text{diag}(w_1, \dots, w_n), \quad G = X^\top W X.$$

Assume X has full column rank, so $n \geq d$. Training to the least-squares minimum on the copied rows is equivalent to minimizing

$$L_{\text{train}}(\theta) = \frac{1}{T}(y - X\theta)^\top W(y - X\theta).$$

Setting the gradient to zero gives

$$G\hat{\theta} = X^\top W y, \quad \hat{\theta} - \theta_* = G^{-1} X^\top W \varepsilon.$$

The model fits accidental correlations between observed features and the hidden signal. The weights determine how strongly each residual pulls on the fitted coefficients.

5 How repetition changes prediction error

The fitted coefficients contain some of the training residuals. To measure their effect on generalization, evaluate the fit on a fresh example. Averaging over its independent x and z gives

$$L_{\text{test}}(\hat{\theta}) = \sigma^2 + \|\hat{\theta} - \theta_*\|^2.$$

Condition on X . The coefficient error has mean zero, and its covariance is

$$\begin{aligned} \mathbb{E}[(\hat{\theta} - \theta_*)(\hat{\theta} - \theta_*)^\top \mid X] &= G^{-1}X^\top W \mathbb{E}[\varepsilon\varepsilon^\top \mid X] W X G^{-1} \\ &= \sigma^2 G^{-1}X^\top W^2 X G^{-1}. \end{aligned}$$

The squared coefficient error is the sum of its coordinate squares. Taking the trace therefore yields

$$\boxed{\mathbb{E}[L_{\text{test}} \mid X] - \sigma^2 = \sigma^2 \text{tr}\left(G^{-1}X^\top W^2 X G^{-1}\right).}$$

The W^2 is the cost of sharing a residual: r copies contribute $r\varepsilon_i$, whose variance is $r^2\sigma^2$. By comparison, the sum of r independent residuals has variance $r\sigma^2$.

For pool Gram matrices $G_u = X_u^\top X_u$ and $G_r = X_r^\top X_r$, this becomes

$$\boxed{\mathbb{E}[L_{\text{test}} \mid X] - \sigma^2 = \sigma^2 \text{tr}\left[(G_u + rG_r)^{-1}(G_u + r^2G_r)(G_u + rG_r)^{-1}\right].}$$

When both pools cover the feature directions

If $n_u, n_r \gg d$, the Gaussian Gram matrices concentrate around $G_u \approx n_u I_d$ and $G_r \approx n_r I_d$. Substituting gives

$$\boxed{\mathbb{E}[L_{\text{test}} \mid X] - \sigma^2 \approx \sigma^2 d \frac{n_u + r^2 n_r}{(n_u + r n_r)^2} = \frac{\sigma^2 d}{T} (1 - \rho + \rho r).}$$

At fixed r and ρ , this is a d/T estimation-error law with a repetition penalty. At fixed T , the approximation grows with r : heavier weighting of a shrinking pool increases estimation error.

As $n_r = \rho T/r$ shrinks toward d , the approximation $G_r \approx n_r I_d$ breaks down. For $n_r < d$, G_r even has rank at most n_r : the repeated pool constrains only a subspace. We next examine why this can change the shape of the curve.

6 Why can a smaller repeated pool be easier to fit?

Keep T , ρ , and d fixed. As r grows, each repeated residual gets more weight, but the repeated pool contains fewer distinct constraints. Each original asks for only one equation, $x_i^\top (\hat{\theta} - \theta_*) \approx \varepsilon_i$.

To see the geometry, consider one repeated example (x, ε) . Write $\delta = \hat{\theta} - \theta_*$. Approximate a large unrepeated pool's loss by its population value. After dropping a constant, the objective is

$$n_u \|\delta\|^2 + r(\varepsilon - x^\top \delta)^2.$$

When $r\|x\|^2 \gg n_u$, its solution approaches the smallest change that fits the repeated example:

$$\min_{\delta} \|\delta\|^2 \quad \text{subject to} \quad x^\top \delta = \varepsilon.$$

Only the component parallel to x helps satisfy the constraint, so

$$\delta = \frac{\varepsilon x}{\|x\|^2}, \quad \|\delta\|^2 = \frac{\varepsilon^2}{\|x\|^2}.$$

For a high-dimensional standard Gaussian x , $\|x\|^2$ is typically about d . Fitting one residual then costs about ε^2/d in excess prediction loss. The fit uses the direction of this one example, leaving orthogonal directions unchanged.

A moderately sized repeated pool can disturb many directions. A small pool can be fitted in fewer directions, and already-fitted examples stop pulling harder. This competition can produce a peak and partial recovery in the fixed-budget experiment.

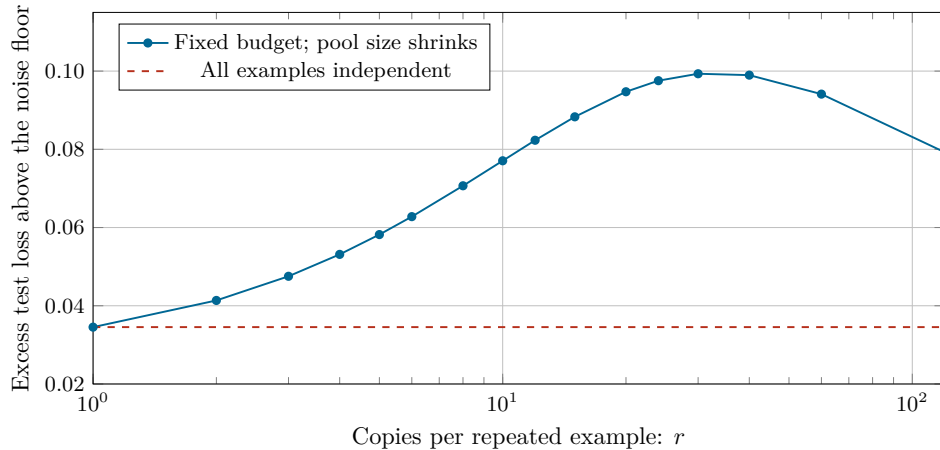


Figure 3: Course simulation of the regression model, using the exact conditional-risk formula from Section 5. $T = 600$, $\rho = 1/5$, $d = 20$, and $\sigma^2 = 1$. The repeated pool shrinks as r grows. Each point averages 512 Gaussian feature designs; bars show one Monte Carlo standard error. The dashed line uses T independent examples. This figure is distinct from the language-model measurements in Figure 1.

The exact risk formula lets us check whether this competition produces a peak for a given budget, dimension, and pool size. Here the loss partly recovers but remains above the independent-data baseline.

A Supporting calculations for repeated regression

The formula in Section 5 gives two useful checks. If $\sigma = 0$, the full-rank learner recovers θ_* exactly. If every weight is multiplied by the same positive constant, the factors cancel and the fit is unchanged.

Coverage with independent examples

Set $W = I_n$, so all examples are independent originals. Then

$$\mathbb{E}[L_{\text{test}} \mid X] - \sigma^2 = \sigma^2 \text{tr}((X^\top X)^{-1}) = \sigma^2 \sum_{j=1}^d \frac{1}{\lambda_j(X^\top X)}.$$

Here λ_j are the positive eigenvalues of the full-rank Gram matrix. A small eigenvalue identifies a direction with little training information and a large estimation cost under our isotropic test distribution.

Training loss

At the optimum, expanding the quadratic and using the normal equations gives

$$TL_{\text{train}}(\hat{\theta}) = \varepsilon^\top W \varepsilon - \varepsilon^\top W X G^{-1} X^\top W \varepsilon.$$

For any fixed matrix A , $\mathbb{E}[\varepsilon^\top A \varepsilon \mid X] = \sigma^2 \text{tr}(A)$. Therefore

$$\boxed{\mathbb{E}[L_{\text{train}} \mid X] = \frac{\sigma^2}{T} \left[T - \text{tr}(G^{-1} X^\top W^2 X) \right].}$$

Fitting a shared residual reduces its contribution to training loss while possibly increasing prediction error.

One repeated example at finite weight

In the population approximation from Section 6, differentiate $n_u \|\delta\|^2 + r(\varepsilon - x^\top \delta)^2$. The normal equation and its solution are

$$(n_u I + r x x^\top) \delta = r \varepsilon x, \quad \delta = \frac{r \varepsilon}{n_u + r \|x\|^2} x.$$

Thus

$$\|\delta\|^2 = \frac{r^2 \varepsilon^2 \|x\|^2}{(n_u + r \|x\|^2)^2} \longrightarrow \frac{\varepsilon^2}{\|x\|^2}.$$

For this fixed example and fixed n_u , its cost grows and saturates. The recovery mechanism in the main lecture compares shrinking pools. The accompanying tutorial gives the fixed-pool monotonicity proof and the full Gaussian-regression simulation.

References

- [1] Katherine Lee et al. *Deduplicating Training Data Makes Language Models Better*. Proceedings of ACL, 2022, pp. 8424–8445.
- [2] Jessica Chudnovsky et al. *Internal Data Repetition Destroys Language Models*. arXiv:2606.24998v1, 2026.