INFLUENCE DYNAMICS AND STAGEWISE DATA ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

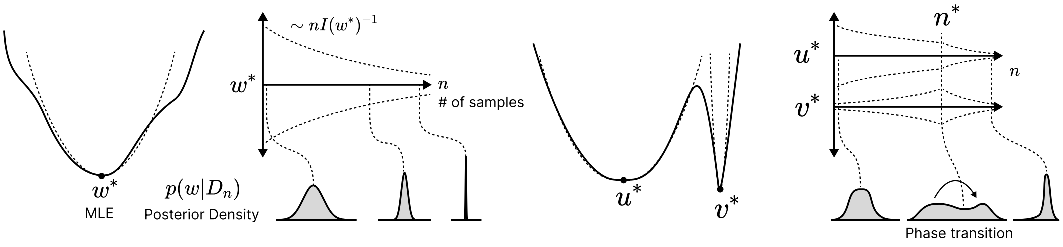
Current training data attribution (TDA) methods treat the influence one sample has on another as static, but neural networks learn in distinct stages that exhibit changing patterns of influence. In this work, we introduce a framework for stagewise data attribution grounded in singular learning theory. We predict that influence can change non-monotonically, including sign flips and sharp peaks at developmental transitions. We first validate these predictions analytically and empirically in a toy model, showing that dynamic shifts in influence directly map to the model's progressive learning of a semantic hierarchy. Finally, we demonstrate these phenomena at scale in language models, where token-level influence changes align with known developmental stages. This is a REBUTTAL VERSION

1 INTRODUCTION

Training data attribution (TDA) studies how training data shapes model behavior, a central problem in AI interpretability and safety (Cheng et al., 2025; Lehalleur et al., 2025). Understanding attribution requires accounting for the role of stagewise development and learning dynamics: *when* a model encounters a given sample affects *how* the model learns from that sample. What helps the model learn "dog" in early training may actively harm its ability to distinguish "poodle" from "terrier" later.

Currently, however, most approaches to TDA still ignore the role of development. In particular, influence functions (IFs) assume that data ordering has no effect on influence, which implies that influence is static and global over the course of training (Cook, 1977; Cook & Weisberg, 1980). This perspective, inherited from the analysis of regular statistical models, breaks down catastrophically for deep neural networks (see Section 2.1).

The cause of this breakdown is degeneracy: neural networks have degenerate loss landscapes with non-isolated critical points and non-invertible Hessians. Singular learning theory (SLT) predicts that this degeneracy gives rise to stagewise development, where models undergo phase transitions marked by changes in degeneracy and Hessian rank (Watanabe, 2009; 2018). Taken together, this suggests



(a) **Regular Development** (Bernstein von-Mises)

(b) **Stagewise Development** (Watanabe)

Figure 1: **Stagewise learning requires stagewise data attribution.** (a) In regular models, development is a uniform, monotonic process of posterior concentration around a single solution (Bernstein–von Mises). (b) In singular models, development is a stagewise process where the posterior undergoes phase transitions (Watanabe's singular learning theory). This stagewise development means the influence one sample has on another can profoundly change over the course of the learning process.

that data attribution should take account of stagewise development, which motivates our work to connect influence functions and developmental phase transitions.

Contributions. We put forth three contributions:

1. **A developmental framework for influence**: We introduce a theoretical framework, grounded in Singular Learning Theory (SLT), that connects influence functions to stagewise development. This predicts that influence is not static but can change non-monotonically (including sign flips and sharp peaks) at the phase transitions that define stagewise learning. This motivates a shift to **stagewise data attribution** that studies the dynamics of influence over time.
2. **Validation in a controlled system**: We analytically derive and empirically confirm our predictions in a hierarchical feature-learning model (Saxe et al., 2019a). We provide a concrete proof-of-concept showing that dynamic shifts in influence directly correspond to the model's sequential learning of the data's hierarchical structure.
3. **Application at scale in language models**: We demonstrate that these developmental phenomena are observable at scale in language models. In particular, we show that influence functions for tokens with key structural roles (e.g., delimiters, induction patterns) undergo non-monotonic, sudden changes that align with known transitions.

Our findings challenge the foundational assumptions of the static influence paradigm, providing a new framework for understanding not just which data points matter, but *when* and *why* they matter during the learning process.

2 THEORY

This section develops our theoretical framework for stagewise data attribution. In Section 2.1, we review the theory of stagewise development according to singular learning theory (SLT). We then, in Section 2.2, re-evaluate influence functions through this developmental lens, which motivates a shift from the global, point-wise approach of classical influence functions to a local, distributional variant. Finally, in Section 2.3, we derive our central predictions: influence is not a fixed property but changes non-monotonically, peaking at the phase transitions that define stagewise learning.

2.1 FROM UNIFORM TO STAGEWISE DEVELOPMENT

Developmental Interpretability: from SGD to Bayes and back again. While stochastic optimizers (such as SGD and Adam) are the de facto approach to training deep learning systems, their complex dynamics make direct theoretical analysis difficult. To make progress, we follow the recipe of Developmental Interpretability (Lehalleur et al., 2025; Wang et al., 2025b): we model the optimizer's learning trajectory with an idealized Bayesian learning process, then apply singular learning theory (SLT; Watanabe 2009) to make predictions about stagewise development, and finally test those predictions empirically in real networks trained via stochastic optimization.

The regular learning process (Bernstein–von Mises). In *regular* statistical models (with a unique MLE and invertible Fisher information matrix, FIM), the Bernstein–von Mises (BvM) theorem predicts a smooth, monotonic learning process where the posterior narrows around a single solution (Van der Vaart, 2000). More precisely, as the number of samples n increases, the Bayesian posterior converges to a Gaussian centered at the minimum $\boldsymbol{w}^*$ of the population loss with covariance $(n\boldsymbol{I}(\boldsymbol{w}^*))^{-1}$, where $\boldsymbol{I}(\boldsymbol{w}^*)$ is the FIM.

The singular learning process (Watanabe). Neural networks violate the regularity assumptions required for the Bernstein–von Mises theorem to hold. Not only do they have no unique minimum, but the loss landscape is *degenerate*: the Fisher information matrix is not everywhere invertible. SLT provides a framework for studying these *singular* models. Watanabe (2009) showed that degeneracy can give rise to stagewise learning, where neural networks undergo a succession of phase transitions between qualitatively distinct solutions, see Figure 1.

In this framework, development is driven by a competition between data fit (or empirical loss, $L_n(\boldsymbol{w}) = \sum_i \ell_i(\boldsymbol{w})$ over a dataset $\mathcal{D}$ of n samples) and model complexity (as measured via a measure of degeneracy known as the local learning coefficient, $\lambda(\boldsymbol{w})$). This evolving tradeoff can lead to first-order phase transitions, where the model abruptly shifts from concentrating in one region to another, and which can change the model's generalization behavior (see Appendix A for a formal treatment). As we will show in Section 2.3, these transitions are also responsible for the non-monotonic dynamics of influence functions throughout the learning process.

2.2 From Static to Developmental Influence Functions

The stagewise development of singular models requires a corresponding shift in our tools for data attribution, moving from global, point-wise measures to local, distributional ones.

Classical influence functions: a static view. Classical influence functions (IFs) are a standard technique for training data attribution, quantifying how an infinitesimal upweighting of a training point $\mathbf{z}_i$ affects an observable ϕ evaluated at the final model parameters $\boldsymbol{w}^*$ (Cook, 1977). The influence is given by:

$$\mathrm{IF}(\mathbf{z}_i, \phi) = \frac{\partial}{\partial \beta_i} \phi(\boldsymbol{w}^*(\boldsymbol{\beta}))\bigg|_{\boldsymbol{\beta}=\mathbf{1}} = -\nabla_{\boldsymbol{w}} \phi(\boldsymbol{w}^*)^\top \boldsymbol{H}^{-1}(\boldsymbol{w}^*) \nabla_{\boldsymbol{w}} \ell_i(\boldsymbol{w}^*), \tag{1}$$

where $\boldsymbol{H}(\boldsymbol{w}^*)$ is the Hessian of the total loss evaluated at the solution $\boldsymbol{w}^*$.

Crucially, the classical IF relies on the same regularity assumptions required for the Bernstein–von Mises theorem: the existence of a single, stable local minimum $\boldsymbol{w}^*$, and an invertible loss Hessian $\boldsymbol{H}(\boldsymbol{w}^*)$ at that minimum. As discussed, singular models like neural networks violate these conditions. Their loss landscapes are degenerate, featuring non-isolated minima and rank-deficient Hessians. This renders the classical IF theoretically ill-defined and practically unstable, which requires a dampening factor (see Appendix C.4), especially at intermediate checkpoints that are unconverged and away from minima.

Bayesian influence functions: a developmental tool. The Bayesian Influence Function (BIF) provides a principled alternative that is well-suited to the dynamics of singular models (Giordano et al., 2017; Kreer et al., 2025). Instead of measuring the change in a point estimate $\phi(\boldsymbol{w}^*)$, the BIF measures how the *posterior expectation* of an observable $\mathbb{E}[\phi(\boldsymbol{w})]$ changes. This derivative is equivalent to the negative covariance between the observable and the sample's loss:

$$\mathrm{BIF}(\mathbf{z}_i, \phi) = \frac{\partial}{\partial \beta_i} \mathbb{E}_{p_{\boldsymbol{\beta}}(\boldsymbol{w}|\mathcal{D})}[\phi(\boldsymbol{w})]\bigg|_{\boldsymbol{\beta}=\mathbf{1}} = -\mathrm{Cov}_{p(\boldsymbol{w}|\mathcal{D})}(\ell_i(\boldsymbol{w}), \phi(\boldsymbol{w})). \tag{2}$$

This formulation is ideal for studying development for three main reasons:

- It is **distributional**. Its definition in terms of expectations over posteriors makes it a natural tool for the Bayesian learning framework of SLT.
- It is inherently **Hessian-free**. By replacing the problematic Hessian inverse with a covariance estimation, it remains well-defined even on degenerate loss landscapes.
- It is **well-defined at any point** in the training trajectory, not just at stable local minima. This is essential for studying influence as a dynamic quantity that evolves over time.

Moreover, when the regularity assumptions hold, the BIF asymptotically recovers the classical IF in the large-data limit (Kreer et al., 2025); that is, the BIF is a natural higher-order generalization of the classical influence function. For these reasons, we adopt the BIF as our primary tool for measuring influence.

Estimating (local) Bayesian influence functions. To measure the BIF in practice, we use an estimator based on stochastic-gradient MCMC introduced in Kreer et al. (2025). This also introduces a dampening term that enables localizing the BIF to individual model checkpoints. For more details, see Appendix B, where we also discuss practical scaling advantages of the BIF compared to other popular IF methods, such as EK-FAC (Grosse et al., 2023).

2.3 Stagewise Data Attribution

Influence and susceptibility. In the language of statistical physics, the BIF is an example of a generalized *susceptibility* that measures a system's response (in this case, a model's loss on sample j) to a perturbation (in this case, change in importance of sample i). In physical systems, susceptibilities diverge at phase transitions, which makes them macroscopically the most legible signal that a phase transition is taking place. This suggests using (Bayesian) influence functions to *discover* transitions during the learning process.

Setup: a bimodal posterior. A first-order phase transition is characterized by the posterior distribution $p(\boldsymbol{w} \mid \mathcal{D})$ having significant mass in two distinct neighborhoods, which we label $\mathcal{U}$ and $\mathcal{V}$. We can model this as a mixture distribution:

$$p(\boldsymbol{w} \mid \mathcal{D}) = \pi_{\mathcal{U}} p(\boldsymbol{w} \mid \mathcal{U}) + \pi_{\mathcal{V}} p(\boldsymbol{w} \mid \mathcal{V})$$

where $\pi_{\mathcal{U}}$ and $\pi_{\mathcal{V}}$ are the posterior probabilities of being in phase $\mathcal{U}$ or $\mathcal{V}$ respectively, with $\pi_{\mathcal{U}} + \pi_{\mathcal{V}} = 1$. At the peak of a phase transition, $\pi_{\mathcal{U}} \approx \pi_{\mathcal{V}} \approx 0.5$. Away from the transition, one of the weights is close to 1 and the other is close to 0.

Decomposing influence with the law of total covariance. The BIF between samples i and j is defined as $\text{BIF}(\mathbf{z}_i, \ell_j) = -\text{Cov}_{p(\boldsymbol{w}|\mathcal{D})}(\ell_i(\boldsymbol{w}), \ell_j(\boldsymbol{w}))$. We can decompose this total covariance using the Law of Total Covariance, conditioning on the phase ($Z \in \{\mathcal{U}, \mathcal{V}\}$):

$$\text{Cov}(\ell_i, \ell_j) = \underbrace{\mathbb{E}[\text{Cov}(\ell_i, \ell_j \mid Z)]}_{\text{Average Within-Phase Influence}} + \underbrace{\text{Cov}(\mathbb{E}[\ell_i \mid Z], \mathbb{E}[\ell_j \mid Z])}_{\text{Between-Phase Influence}}$$

Let's analyze each term:

1. **Average Within-Phase Influence:** This term is the weighted average of the influence calculated strictly within each phase:
$$\mathbb{E}[\text{Cov}(\ell_i, \ell_j \mid Z)] = \pi_{\mathcal{U}} \text{Cov}_{\mathcal{U}}(\ell_i, \ell_j) + \pi_{\mathcal{V}} \text{Cov}_{\mathcal{V}}(\ell_i, \ell_j)$$
This represents the "baseline" influence. If there were no phase transition (e.g., $\pi_{\mathcal{U}} = 1$), this is the only term that would exist.
2. **Between-Phase Influence:** This term captures the covariance that arises because the expected losses themselves change as the model switches phases. Let $\mu_{i,\mathcal{U}} = \mathbb{E}[\ell_i \mid \mathcal{U}]$ be the expected loss of sample i in phase $\mathcal{U}$, and likewise for j and $\mathcal{V}$. The term expands to:
$$\text{Cov}(\mathbb{E}[\ell_i \mid Z], \mathbb{E}[\ell_j \mid Z]) = \pi_{\mathcal{U}} \pi_{\mathcal{V}} (\mu_{i,\mathcal{U}} - \mu_{i,\mathcal{V}})(\mu_{j,\mathcal{U}} - \mu_{j,\mathcal{V}})$$

Predicting stagewise changes in influence. This decomposition predicts dynamic changing influence patterns over learning. The departure from the classical view arises because influence is phase-dependent: the baseline "within-phase" influence may differ significantly across the transition ($\text{Cov}_{\mathcal{U}} \neq \text{Cov}_{\mathcal{V}}$), and the "between-phase" term introduces an additional effect during the transition. In particular, we derive two predictions that diverge from the classical view:

- **Influence Can Change Sign:** If the within-phase influences of the two phases have significantly different values or if the between-phase term is large enough to dominate the average baseline influence during the transition, then transitions can cause a large change in magnitude or even a change in sign.
- **Influence Peaks at Transitions:** The between-phase influence term is maximized when the posterior mass is evenly split ($\pi_{\mathcal{U}} \approx \pi_{\mathcal{V}} \approx 0.5$), causing a sharp peak in total influence at the critical point of a transition. The magnitude of the between-phase term is proportional to $(\mu_{i,\mathcal{U}} - \mu_{i,\mathcal{V}})$, which means the influence spike is largest for the samples on which the two phases disagree the most: peaks in influence identify the specific samples that *characterize* a given transition.

Towards stagewise data attribution. These theoretical predictions call for a shift from the classical static view of training data attribution to what we term *stagewise data attribution*: the analysis of influence as a dynamic trajectory over the entire learning process. The goal is to attribute learned behaviors not only to the data that influenced it, but also to the specific period of time in which that data had its effect. In the rest of this paper, we turn to testing the basic predictions above in order to validate this developmental framework.

3 Toy Model of Development

The acquisition of semantic knowledge involves the dynamic development of hierarchical structure in neural representations. Often, broader categorical distinctions are learned prior to finer-grained distinctions, and abrupt conceptual reorganization marks the non-static nature of knowledge acquisition, studied in both psychology and deep learning literature (Keil, 1979; Inhelder & Piaget, 1958; Hinton, 1986; Rumelhart & Todd, 1993; McClelland, 1995; Rogers & McClelland, 2004). For a principled understanding of the dynamical aspect of influence over training, we study a toy model of hierarchical feature learning introduced by Saxe et al. (2019a), for which ground truth and analytical tools are accessible.

We confirm our theoretical predictions in Section 2.3 by finding that the dynamics of influence coincide with the stagewise development of hierarchical structure. Sign flips occur as the model shifts to learning progressively finer levels of hierarchical distinctions, and peaks occur when the model begins learning a new level of distinction.

Setup: a hierarchical semantic dataset. We train a 2-layer deep linear network with MSE loss on a hierarchical semantic dataset from Saxe et al. (2019a). The dataset consists of one-hot input vectors representing objects, and each input maps to an output vector representing a collection of features that the object possesses (Figure 2). Importantly, this toy model mathematically shows that deep neural network architecture develops neural representation that reflects hierarchical differentiation in a progressive manner: learning animal vs. plants first, then mammals vs. birds, and then dogs vs. cats. See Appendix C for a detailed description of the toy model and its analytical treatment.

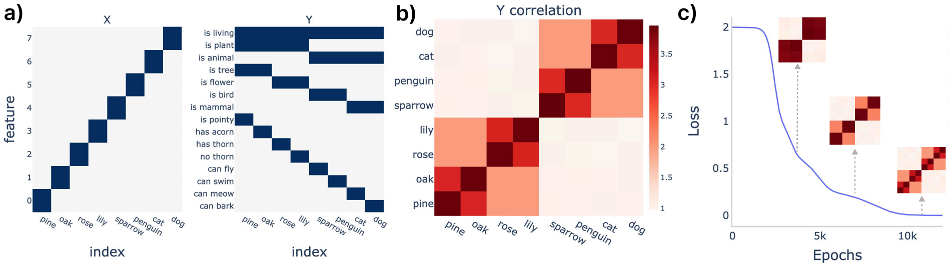


Figure 2: **A toy model of hierarchical semantic knowledge acquisition.** (a) A toy dataset adopted from Saxe et al. (2019a). Each object maps to a feature vector that describes the hierarchical structure of semantic knowledge, e.g., 'penguin' and `cat` are all living animals, but 'penguin' is a bird and `cat` is a mammal. (b) The correlation matrix of the feature output shows a hierarchical structure. (c) The hierarchical structure is acquired progressively during the training of the deep linear neural network.

Measuring influence *dynamics*. First, we probe the local BIF (Kreer et al., 2025) on the toy model over the entire learning trajectory. We use RMSProp-preconditioned SGLD sampler (described in Appendix B) to estimate a posterior from each checkpoint $\boldsymbol{w}_t^*$ at training time t. See Appendix B for details of the local BIF implementation and Appendix C.2 for the hyperparameter sweep. Furthermore, we derive the dynamics of the influence function analytically, leveraging the mathematical tractability of the toy model (see Appendix C.6).

Leave-one-out (LOO) verification. To confirm our observation from the BIF and analytical treatment, we conduct retraining experiments. Specifically, we consider the Leave-One-Out (LOO) setting, where we ablate one data point and measure the loss difference of other data points compared to the baseline loss without ablation. We measure the loss difference *over training time* t:

$$\Delta\ell_{j,t}^{\setminus i} = \ell_{j,t}^{\mathcal{D}} - \ell_{j,t}^{\mathcal{D}_{\setminus i}}, \tag{3}$$

measured at time t where $\mathcal{D}$ is the full dataset, $\mathcal{D}_{\setminus i}$ is the dataset with data index i ablated, and j is the index of the data point that we are querying. In Figure 3, we observe that the loss difference from LOO results in a similar pattern to what we see in the BIF and the analytical derivation, and

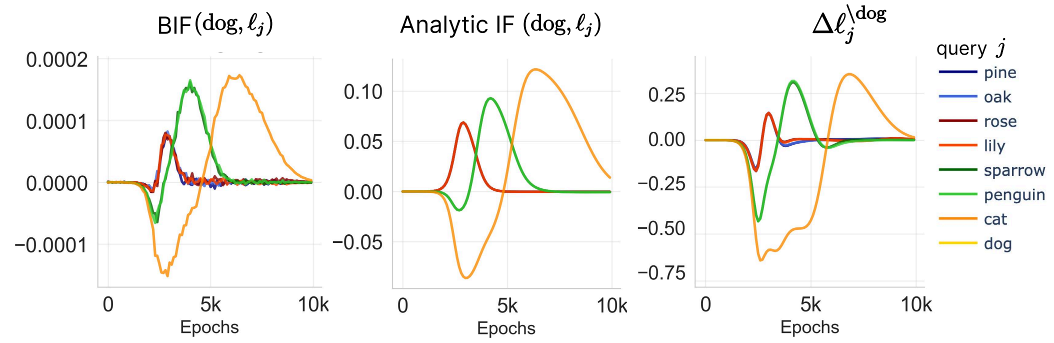


Figure 3: **Influence over time on a hierarchical semantic dataset.** We measure the influence of `dog` sample on other query samples j with the following: (Left) BIF ($\beta = 1000, \epsilon = 1e\text{-}3, \gamma = 5e + 3$). (Center) Analytical IF (see full derivation in Appendix C.6). (Right) Loss difference from Leave-One-Out (LOO) retraining experiment. All three measures agree that the influence one sample has on another can vary non-monotonically over the course of training as discussed in Section 3. See Appendix C.2 for additional pairs of samples and experimental details.

validates their use. The additional results on perturbing other data points show consistent trend in Appendix C.2 and are also visible in the dampened classical IF Appendix C.4.

With these results, we confirm the predictions from Section 2: (1) influence changes over time non-monotonically and can change sign, and (2) peaks in influence are correlated with key developmental transitions in model behavior. In the rest of the section, we will establish how these observations are reflected in the progressive learning of the hierarchical structure in our toy model.

“When” matters for measuring influence. We clearly observe that influence goes through non-monotonic change over the course of training, strengthening our argument that a static interpretation of influence is fallacious. In Appendix C.6, we analytically derive that influence is a function of singular mode strength of data input-output covariance learned by the network, which is a time-dependent variable and thus justifies studying data attribution from a dynamic perspective. Over the course of training, each data point induces a non-static influence on other data points. The influence can flip sign—the same data can be either helpful or harmful, depending on when it is presented. It can also mark clear peaks at a specific time point on different query data points. Furthermore, the influence over time from one data point to another point is specific to those points. The influence from `dog` to `cat` might be the same as `dog` to ‘penguin’ early on, but they are distinguished later in the learning process.

To confirm that these influence dynamics represent a causal effect on model performance, we perform an additional retraining experiment in Appendix C.5. Instead of removing a sample for the full training run, we ablate specific data points only during short temporal windows. We find that ablating data precisely during the stage where the BIF assigns it peak influence induces the largest loss difference compared to ablation at other times. This demonstrates that our measure correctly identifies the critical window in which a specific sample drives the learning process.

Change of influence reflects stagewise learning. During the phase where one hierarchical distinction (e.g., `animal` vs. `plant`) is being learned, upweighting a data point in the same class (`dog`) is helpful for learning a query data point (`sparrow`) as indicated by a negative influence (positive covariance). In contrast, upweighting a data point that belongs to a different class harms learning that query data point (`pine`), reflected in a positive influence. A data point `dog` is helpful (negative influence) to learning `sparrow` early on in the learning, while learning to distinguish `animal` vs. `plant`, but it is harmful (positive) later on when learning to distinguish `mammal` vs. `bird`. In Figure 4 b), we show Multi-Dimensional Scaling (MDS, Torgerson 1952; Cox & Cox 2008) of hidden representation of each data over time as in Saxe et al. (2019a) where learning of each hierarchical level is reflected on the branching node (numbered). We observe that the time point of the branching node matches the peaks in influence. That is, at the transition, where the model learns to distinguish mammal vs bird within the animals and form a new hierarchy level, the influence between mammal and bird is the highest.

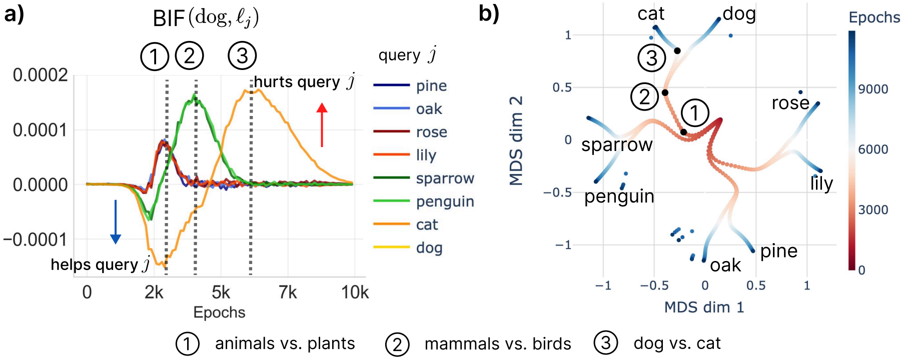


Figure 4: **BIF captures developmental influence.** (a) The peak positive influence from dog to different data points is noted with ①, ②, and ③. (b) MDS of the hidden representations in the network over the course of learning. The peaks in influence match the branching points of the MDS trajectory, where each hierarchical category develops (black points).

4 Language Models

To investigate the dynamics of influence in a real-world setting, we study the acquisition of token-level syntactic knowledge using language models from the Pythia Scaling Suite (Biderman et al., 2023).

We confirm our theoretical predictions in Section 2.3 by finding non-monotonic influence trajectories, with large changes in magnitude, sign flips, and peaks that correspond to known developmental changes like the formation of the induction circuit.

Per-token influence functions. One clear benefit of the BIF in the language-modeling setting is that computing influence at the level of individual tokens rather than sequences incurs no additional computational cost. Loss is computed on a per-token basis during RMSprop-SGLD, which is already necessary when considering the autoregressive losses that represent the standard for LLM pretraining:

$$\ell_i(\boldsymbol{w}) = \sum_k \ell(x_{i,k} \mid x_{i,0} \dots x_{i,k-1}, \boldsymbol{w}),$$

where $x_{i,j}$ is the jth token in the ith text sequence. These per-token losses can then be stored individually and used to estimate the per-token BIF matrix for the relevant dataset.

Classifying tokens into syntactic classes. Following the experimental setup of Baker et al. (2025), we classify individual tokens according to how they are used to give structure to text. These classes include strictly syntactic tokens (left delimiters, right delimiters, and formatting tokens—such as newlines), morphological roles (parts of words and word endings), and a broader structural class—tokens that have been used in conjunction earlier in the context, forming an inductive pattern. We note that this classification is not exhaustive, nor is it exclusive—not all tokens have a class, and some may occupy multiple classes at once. A full classification is provided in Appendix D.1.1.

Calculating group influence. To estimate patterns of influence between tokens across classes, we make use of the following procedure:

1. Using a subset of The Pile (Gao et al., 2021), we compute the *normalized* BIF (Kreer et al., 2025) between all pairs of tokens, sampled from the SGLD-estimated model posterior, as described in Appendix B.
2. Each token is classified in accordance with the listed structural classes based on pattern-matching, following Baker et al. (2025).

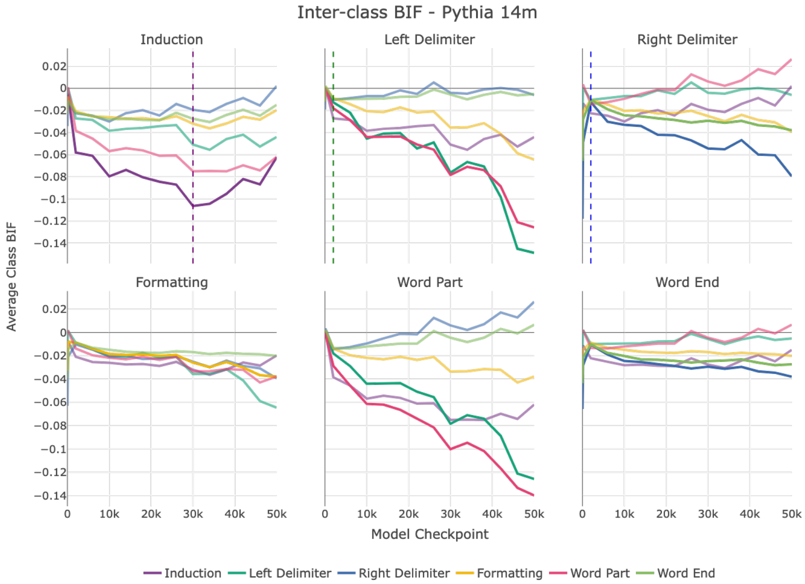


Figure 5: **Token–class relationships.** BIF between structural classes through training. We observe structural relationships between tokens reflected in influence patterns between classes, including strong intra-class relationships, development of induction, and relationships between word elements and corresponding delimiter token classes. Dashed lines indicate major inflection points in the BIF. Opacity captures strongly influencing pairs of tokens. For additional resolution into the early stages of training, see Appendix D.1.4

3. For every possible pair of classes, we compute the average (or "group") influence between tokens in these classes, excluding influences between the same token in different classes.

These inter-class influences can then be used to provide insight into how much influence tokens from one class have tokens from other classes.

Learning induction. In Figure 5, we plot the inter-class influence for each query class across all class pairings at several model checkpoints. From these plots, some clear dynamics emerge. We see that the formation of model structures capturing induction relationships in the BIF starts as early as 1000 steps into learning, and continues to strengthen for the next 30k training steps before appearing to peak and fall, corresponding with the results of Tigges et al. (2024), which finds that Pythia models begin to learn the induction circuit at this time, and exhibit an apex at 30k training steps before diminishing.

In Appendix D.3, we study a small language model trained from scratch and track the influence on a targeted set of synthetic samples to automatically identify the induction bump. We find that the influence of induction-pattern tokens sharply rises as the induction circuit is being learned. We further validate the practical utility of these insights with a stagewise intervention experiment in Appendix D.4. We find that upweighting induction patterns specifically when the induction circuit begins to form accelerates the formation of induction heads significantly more than upweighting them before this window. This confirms the prediction derived from the influence dynamics: influence cannot be fully understood from a static analysis at the end of training.

Learning where to end. Another notable pattern occurs in the influence traces of left delimiters and right delimiters. Both classes appear among high-influence tokens for the other class very early in training, but this relationship quickly inverts, with tokens from the opposite delimiter class progressively becoming increasingly negative as the model learns to distinguish between these classes. We also see increasing influence between left and right delimiters and related classes that may perform similar roles in structuring text. Left delimiters share high influence with parts of words but not the ends of words, while right delimiters share uplifted influence with ends of words but not word parts. These relationships are also visible in the reverse direction (from word parts / word ends to left / right delimiters). These relationships develop over time, with a particularly notable spike in the relationship between word parts and left delimiters later in model training. For additional more granular influence trajectories between individual pairs of tokens, see Appendix D.2

Dynamics of influence. Taken together, these results demonstrate that influence between different classes of tokens is not a static property but a dynamic one that evolves throughout training. We observe changes in both the magnitude and sign of influence, with the timing of these shifts varying depending on the specific structural capability being learned. For instance, the influence related to induction patterns exhibits a non-monotonic peak that aligns with the known developmental phase transition for this circuit. Similarly, the relationship between left and right delimiters undergoes a sign flip, indicating a qualitative shift in how the model processes scope and pairing. These findings confirm the predictions of our stagewise framework: different data becomes influential at different times, reflecting the model's progressive, stage-by-stage acquisition of syntactic and structural knowledge.

5 Discussion & Conclusion

Stagewise data attribution and developmental interpretability. This work places data attribution within the context of Developmental Interpretability. We follow the established methodology of using singular learning theory (SLT) as a theoretical lens to study the dynamics of models trained with stochastic optimizers. Prior work has successfully used this pipeline to explain phenomena like phase transitions in toy models of superposition (Chen et al., 2023), algorithm selection in transformers (Carroll et al., 2025), stagewise learning in toy transformers (Urdshals & Urdshals, 2025), and the stagewise emergence of structure (e.g., n-grams, induction, parenthesis-matching, space-counting) in language models (Hoogland et al., 2024; Wang et al., 2025b; Baker et al., 2025; Wang et al., 2025a). Our contribution is to use the SLT account of phase transitions to predict and subsequently verify that a training sample's influence is not static but a dynamic quantity that evolves with the model's development.

Stagewise vs. unrolling-based attribution. Our work complements another important line of research that moves beyond static influence functions: trajectory-based or "unrolling" methods like TracIn (Pruthi et al., 2020), HyDRA (Chen et al., 2021), and SOURCE (Bae et al., 2024). These techniques approximate the total influence of a sample by integrating its contributions (such as gradient updates or local influence scores) across numerous checkpoints along the full training path. This approach provides a more faithful account of the path-dependent nature of SGD and can offer more accurate attribution scores than single-point estimates.

However, our findings regarding influence dynamics raise the possibility of cancellation effects in these unrolling-based measures. Since influence can flip signs during training, opposing contributions from different stages could potentially offset each other when integrated. While such a score may still accurately reflect the expected cumulative impact, it risks obscuring the data point's role in driving specific developmental stages. Fundamentally, the goal of these methods differs from ours: unrolling techniques compute a single, cumulative summary of a sample's total impact, treating the learning process as a black box. In contrast, our framework treats the learning trajectory as an object of study in itself, aiming to understand *when* data matters during development.

A mechanism for implicit curricula. The concept of a curriculum—learning from easier to harder data—has been argued consistently for more efficient neural learning (Bengio et al., 2009; Wang et al., 2021; Lee et al., 2024). However, the efficacy of this technique has been shown to be limited in practice (Wu et al., 2020; Mannelli et al., 2024). One prevalent explanation is that an *implicit curriculum* is adopted by learning through gradient descent in neural networks (Graves et al., 2017; Rahaman et al., 2019; Saxe et al., 2019a; Valle-Perez et al., 2018). Our findings provide a new, more granular mechanism for understanding this phenomenon. The stagewise evolution of influence demonstrates how different data points become "important" at different moments, effectively creating a dynamic, self-organizing curriculum.

Limitations and future work. Our framework points toward several avenues for future research. The primary theoretical gap remains the link between the Bayesian learning process of SLT and the non-equilibrium dynamics of SGD. On the empirical front, a key direction is to move from a behavioral to a mechanistic account of influence. Ultimately, model generalization is grounded in the circuits and internal structure a model acquires over training. A more complete science of interpretability therefore understands not just *which samples* are most influential (data attribution), but *when* they are most influential (stagewise attribution), and *how* they shape model internals such as features and circuits (mechanistic interpretability).

From pointwise to stagewise data attribution. Ultimately, this paper argues for a shift in how we approach training data attribution. The static perspective, which assigns a single, global influence score to each data point, only at the end of training, offers an incomplete and, at times, misleading picture. By demonstrating that influence is inseparable from development, we advocate for moving from point-wise to stagewise data attribution.

This developmental lens is essential for tackling a key scientific challenge: understanding the correspondence between data structure, loss landscape geometry, learning dynamics, and model internals (Wang et al., 2025b). Stagewise data attribution, by tracking influence *dynamics*, provide a concrete tool to map these connections, opening new possibilities for interpreting, debugging, and ultimately steering how models learn.

Reproducibility Statement

We describe our primary method, the Bayesian Influence Function (BIF) estimated via RMSPropSGLD, in Appendix B. We first validate our predictions in a controlled toy model (Section 3), providing analytical derivations, Leave-One-Out (LOO) verification, and all hyperparameters in Appendix C. We then demonstrate our findings at scale using public Pythia language models on The Pile dataset (Section 4), with all experimental specifics, token classification schemes, and hyperparameters detailed in Appendix D.

LLM Usage Statement

We used large language models (LLMs) to assist in the preparation of this manuscript. Their role included proofreading, correcting grammatical errors, and rephrasing sentences to improve clarity and flow. Beyond surface-level edits, we utilized LLMs as a collaborative tool for brainstorming options for narrative structures and receiving feedback on the clarity of our arguments. LLMs also served as coding assistants in implementing our experiments and generating visualizations. Additionally, we used LLMs to aid in the research process, for instance, by suggesting relevant theoretical concepts such as the Law of Total Covariance for the decomposition in Section 2. In all instances, the final content, including all code, theoretical claims, and text, was reviewed, validated, and is the sole responsibility of the authors.

References

Maxwell Adam, Zach Furman, Wilson Wu, Philipp Alexander Kreer, and Jesse Hoogland. The loss kernel: A geometric probe for deep learning interpretability, 2025.

Miki Aoyagi. Consideration on the learning efficiency of multiple-layered neural networks with linear units. *Neural Networks*, 172:106132, 2024.

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If Influence Functions are the Answer, Then What is the Question? 2022. CitationKey: pbrf-if.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger B. Grosse. Training data attribution via approximate unrolling. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15,* *2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/7af60ccb99c7a434a0d9d9c1fb00ca94-Abstract-Conference.html.

Garrett Baker, George Wang, Jesse Hoogland, and Daniel Murfet. Structural Inference: Studying Small Language Models with Susceptibilities, April 2025. URL http://arxiv.org/abs/2504.18274. arXiv:2504.18274 [cs].

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Liam Carroll, Jesse Hoogland, Matthew Farrugia-Roberts, and Daniel Murfet. Dynamics of Transient Structure in In-Context Linear Regression Transformers, January 2025. URL http://arxiv.org/abs/2501.17745. arXiv:2501.17745 [cs].

Yuanyuan Chen, Boyang Li, Han Yu, Pengcheng Wu, and Chunyan Miao. HyDRA: Hypergradient Data Relevance Analysis for Interpreting Deep Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7081–7089, May 2021. ISSN 2374-3468. doi: 10.1609/aaai.v35i8.16871. URL https://ojs.aaai.org/index.php/AAAI/article/view/16871. Number: 8 CitationKey: HyDRA.

Zhongtian Chen, Edmund Lau, Jake Mendel, Susan Wei, and Daniel Murfet. Dynamical versus bayesian phase transitions in a toy model of superposition. *arXiv preprint arXiv:2310.06301*, 2023.

Deric Cheng, Juhan Bae, Justin Bullock, and David Kristofferson. Training Data Attribution (TDA): Examining Its Adoption & Use Cases, January 2025. URL http://arxiv.org/abs/2501.12642. arXiv:2501.12642 [cs].

R. Dennis Cook. Detection of influential observation in linear regression. *Technometrics : a journal of statistics for the physical, chemical, and engineering sciences*, February 1977. ISSN 0040-1706. URL https://www.tandfonline.com/doi/abs/10.1080/00401706.1977.10489493. tex.copyright: Copyright Taylor and Francis Group, LLC.

R Dennis Cook and Sanford Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508, 1980.

R. Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. Monographs on statistics and applied probability. Chapman and Hall, New York, 1982. ISBN 0-412-24280-0. URL https://hdl.handle.net/11299/37076.

Michael AA Cox and Trevor F Cox. Multidimensional scaling. In *Handbook of data visualization*, pp. 315–347. Springer, 2008.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027, 2021. URL https://arxiv.org/abs/2101.00027.

Ryan Giordano, Tamara Broderick, and Michael I. Jordan. Covariances, robustness, and variational Bayes. *Journal of Machine Learning Research*, 19:51:1–51:49, 2017. URL https://api.semanticscholar.org/CorpusID:53238793.

Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. In *international conference on machine learning*, pp. 1311–1320. Pmlr, 2017.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamilė Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying Large Language Model Generalization with Influence Functions, August 2023. URL `http://arxiv.org/abs/2308.03296`. arXiv:2308.03296 [cs].

Geoffrey E Hinton. Learning distributed representations of concepts. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 8, 1986.

Rohan Hitchcock, Gary W Delaney, Jonathan H Manton, Richard Scalzo, and Jingge Zhu. Emergence of computational structure in a neural network physics simulator. *arXiv preprint arXiv:2504.11830*, 2025.

Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. The developmental landscape of in-context learning. *arXiv preprint arXiv:2402.02364*, 2024.

Jesse Hoogland, George Wang, Matthew Farrugia-Roberts, Liam Carroll, Susan Wei, and Daniel Murfet. Loss landscape degeneracy and stagewise development in transformers. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL `https://openreview.net/forum?id=45qJyBG8Oj`.

Bärbel Inhelder and Jean Piaget. *The growth of logical thinking from childhood to adolescence: An essay on the construction of formal operational structures*. Routledge, 1958.

Frank C Keil. *Semantic and conceptual development: An ontological perspective*. Harvard University Press, 1979.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions, December 2020. URL `http://arxiv.org/abs/1703.04730`. arXiv:1703.04730 [stat] CitationKey: deep-influence-functions.

Philipp Alexander Kreer, Wilson Wu, Maxwell Adam, Zach Furman, and Jesse Hoogland. Bayesian influence functions for hessian-free data attribution, 2025.

Edmund Lau, Zach Furman, George Wang, Daniel Murfet, and Susan Wei. The local learning coefficient: a singularity-aware complexity measure. In *The 28th international conference on artificial intelligence and statistics*, 2025. URL `https://openreview.net/forum?id=1av51Z1suL`.

Jin Hwa Lee, Stefano Sarao Mannelli, and Andrew Saxe. Why do animals need shaping? a theory of task composition and curriculum learning. *arXiv preprint arXiv:2402.18361*, 2024.

Simon Pepin Lehalleur, Jesse Hoogland, Matthew Farrugia-Roberts, Susan Wei, Alexander Gietelink Oldenziel, George Wang, Liam Carroll, and Daniel Murfet. You Are What You Eat – AI Alignment Requires Understanding How Data Shapes Structure and Generalisation, February 2025. URL `http://arxiv.org/abs/2502.05475`. arXiv:2502.05475 [cs].

Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Stefano Sarao Mannelli, Yaraslau Ivashynka, Andrew Saxe, and Luca Saglietti. Tilting the odds at the lottery: the interplay of overparameterisation and curricula in neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2024(11):114001, 2024.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2408–2417, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/martens15.html`.

JL McClelland. A connectionist perspective on knowledge and development. In *Developing cognitive competence*, pp. 157–204. Psychology Press, 1995.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating Training Data Influence by Tracing Gradient Descent. In *Advances in Neural Information Processing Systems*, volume 33, pp. 19920–19930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/e6385d39ec9394f2f3a354d9d2b88eec-Abstract.html. CitationKey: TracIn.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International conference on machine learning*, pp. 5301–5310. PMLR, 2019.

Timothy T Rogers and James L McClelland. *Semantic cognition: A parallel distributed processing approach*. MIT press, 2004.

David E Rumelhart and Peter M Todd. 1 learning and connectionist. *Attention and performance XIV: Synergies in experimental psychology, artificial intelligence, and cognitive neuroscience*, 14: 3, 1993.

Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23): 11537–11546, 2019a. doi: 10.1073/pnas.1820226116. URL https://www.pnas.org/doi/abs/10.1073/pnas.1820226116.

Andrew M. Saxe, James L. McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23): 11537–11546, June 2019b. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1820226116. URL https://pnas.org/doi/full/10.1073/pnas.1820226116.

Curt Tigges, Michael Hanna, Qinan Yu, and Stella Biderman. LLM Circuit Analyses Are Consistent Across Training and Scale. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, November 2024.

Warren S Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

Einar Urdshals and Jasmina Urdshals. Structure development in list-sorting transformers. *arXiv preprint arXiv:2501.18666*, 2025.

Guillermo Valle-Perez, Chico Q Camargo, and Ard A Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

George Wang, Garrett Baker, Andrew Gordon, and Daniel Murfet. Embryology of a Language Model, August 2025a. URL http://arxiv.org/abs/2508.00331. arXiv:2508.00331 [cs].

George Wang, Jesse Hoogland, Stan van Wingerden, Zach Furman, and Daniel Murfet. Differentiation and Specialization of Attention Heads via the Refined Local Learning Coefficient. In *Proceedings of The 13th International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=SUc1UOWndp&noteId=MCoFYhi7ZE.

Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):4555–4576, 2021.

Sumio Watanabe. *Algebraic Geometry and Statistical Learning Theory*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2009.

Sumio Watanabe. *Mathematical theory of Bayesian statistics*. Chapman and Hall, 2018.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pp. 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? *arXiv preprint arXiv:2012.03107*, 2020.

Appendix

1. **Appendix A: Phase Transitions.** This section provides a formal outline of how phase transitions arise in singular models and how they impact influence functions.
2. **Appendix B: Bayesian Influence Functions.** This section provides additional experimental details on estimating the local BIF.
3. **Appendix C: Toy Model Details and Additional Results.** This subsection describes the hierarchical dataset, model architecture, and Leave-One-Out (LOO) training procedure. It also contains additional LOO results and the full analytical derivation of the influence dynamics in the deep linear network model.
4. **Appendix D: Language Model Details and Additional Results.** This subsection describes the structural token classification scheme, provides RMSPropSGLD hyperparameters for the Pythia experiments, and details additional validation experiments.

A Phase Transitions

In this appendix, we provide a theoretical justification for the claims made in Section 2 regarding phase transitions and their effect on model predictions and influence, grounded in statistical physics and the theory of Bayesian statistics. This theoretical treatment motivates our empirical investigation into the interaction between stagewise development and influence functions for models trained via standard stochastic optimizers.

A.1 BIF as Susceptibility

Setup. We begin with the model-truth-prior triplet:

1. The **model** $p(\mathbf{z} \mid \boldsymbol{w})$ assigns a probability to a sample $\mathbf{z} \in \mathcal{Z}$ for a given choice of weights $\boldsymbol{w} \in \mathcal{W} \subseteq \mathbb{R}^d$.
2. The **truth** or data-generating distribution $q(\mathbf{z})$ generates a dataset of i.i.d. samples $= \{\mathbf{z}_i\}_{i=1}^n$.
3. The **prior** $\varphi(\boldsymbol{w})$ assigns an initial probability distribution to each choice of weights.

From this triplet, we obtain a posterior through (the repeated application of) Bayes' rules:

$$p(\boldsymbol{w} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{w})\varphi(\boldsymbol{w})}{p(\mathcal{D})}, \tag{4}$$

where $p(\mathcal{D} \mid \boldsymbol{w}) = \prod_i p(\mathbf{z}_i \mid \boldsymbol{w})$ is the likelihood, and $p(\mathcal{D}) = \int_{\mathcal{W}} p(\mathcal{D} \mid \boldsymbol{w})\varphi(\boldsymbol{w})d\boldsymbol{w}$ is the marginal likelihood. Often, we're interested in a *tempered* Bayesian posterior:

$$p_{\boldsymbol{\beta}}(\boldsymbol{w} \mid \mathcal{D}) = \frac{p_{\boldsymbol{\beta}}(\mathcal{D} \mid \boldsymbol{w})\varphi(\boldsymbol{w})}{p_{\boldsymbol{\beta}}(\mathcal{D})} = \frac{e^{-\boldsymbol{\beta}\cdot\boldsymbol{\ell}(\boldsymbol{w})}\varphi(\boldsymbol{w})}{p_{\boldsymbol{\beta}}(\mathcal{D})}, \tag{5}$$

where $\boldsymbol{\beta}$ is a vector of per-sample importance weights, and $\boldsymbol{\ell}(\boldsymbol{w})$ is the vector of per-sample losses for a given weight $\boldsymbol{w}$.

The (global) free energy formula. The (global) free energy is the negative log marginal likelihood, $F_n^{\boldsymbol{\beta}} = -\log p_{\boldsymbol{\beta}}(\mathcal{D})$. A central result of singular learning theory (SLT; Watanabe 2009) is an asymptotic (in the limit of infinite data) expression for this quantity. For a critical point $\boldsymbol{w}^*$ in a neighborhood $\mathcal{W}$, the free energy asymptotically expands as

$$F_n := F_n^{\boldsymbol{\beta}}\Big|_{\boldsymbol{\beta}=\mathbf{1}} = nL_n(\boldsymbol{w}^*) + \lambda(\boldsymbol{w}^*)\log n + O_p(\log\log n), \tag{6}$$

where $\lambda(\boldsymbol{w}^*)$ is the learning coefficient, a degeneracy-aware measure of complexity (that coincides with half the effective parameter count for minimally singular models).

Free energy as moment-generating function. In statistical physics and thermodynamics, the free energy is the central object of study. This is because, if one manages to obtain a closed-form expression for the free energy, it is possible to calculate arbitrary expectation values simply by differentiating the free energy. Expectation values are of primary interest because they correspond to the things we can measure in an experimental setting.

For example, in the learning theory setting, the expected (per-sample) loss can be expressed as

$$\mathbb{E}[\ell_i(\boldsymbol{w})] = \frac{\partial F_n^{\boldsymbol{\beta}}}{\partial \beta_i}\bigg|_{\boldsymbol{\beta}=\mathbf{1}}. \tag{7}$$

Combining Equation (7) with Equation (2), lets us express the BIF as a 2nd-order derivative of the free energy, assuming both samples are in the training dataset:

$$\mathrm{BIF}(\mathbf{z}_i, \mathbf{z}_j) = \frac{\partial \mathbb{E}_{\boldsymbol{\beta}}[\ell_j(\boldsymbol{w})]}{\partial \beta_i}\bigg|_{\boldsymbol{\beta}=\mathbf{1}} = \frac{\partial^2 F_n^{\boldsymbol{\beta}}}{\partial \beta_i \partial \beta_j}\bigg|_{\boldsymbol{\beta}=\mathbf{1}}. \tag{8}$$

Susceptibilities are defined as second-order derivatives of the free energy, which makes the BIF an example of a generalized susceptibility.

A.2 Internal Model Selection

The (local) free energy formula. The local free energy formula is defined analogously to the global free energy, but with the domain of integration restricted to a particular region of parameter space or "phase" $W \subseteq \mathcal{W}$:

$$F_n^{\boldsymbol{\beta}}(W) = -\log \int_W p_{\boldsymbol{\beta}}(\mathcal{D} \mid \boldsymbol{w})\varphi(\boldsymbol{w})d\boldsymbol{w}. \tag{9}$$

This admits an analogous asymptotic form to Equation (6):

$$F_n(W) := F_n^{\boldsymbol{\beta}}(W)\bigg|_{\boldsymbol{\beta}=\mathbf{1}} = nL_n(\boldsymbol{w}^*) + \lambda(\boldsymbol{w}^*)\log n + O_p(\log\log n), \tag{10}$$

but where now $\boldsymbol{w}^*$ is a local minimum within W, and λ is the local learning coefficient associated with that local minimum (Lau et al., 2025).

Coarse-graining. Given a partitioning of parameter space into multiple disjoint "phases" $\mathcal{W} = \cup_i W_i$, the global free energy can be computed from the local per-phase free energies as follows:

$$\begin{aligned}
F_n &= -\log \int_{\mathcal{W}} e^{-nL_n(\mathcal{D}|\boldsymbol{w})}\varphi(\boldsymbol{w})d\boldsymbol{w} && (11)\\
&= -\log \sum_i \int_{W_i} e^{-nL_n(\mathcal{D}|\boldsymbol{w})}\varphi(\boldsymbol{w})d\boldsymbol{w} && (12)\\
&= -\log \sum_i e^{-F_n(W_i)} && (13)\\
&\approx \min_i F_n(W_i). && (14)
\end{aligned}$$

The last line follows from the well-known use of a log-sum exponential as a smooth approximation for the max function. This is to say that globally, the free energy is determined primarily by the phase with the lowest free energy, with exponentially suppressed contributions from all other regions of phase space.

Competition between phases. Assume the posterior distribution is concentrated in just two distinct neighborhoods, which we label $\mathcal{U}$ and $\mathcal{V}$, and is vanishing everywhere else. That is, these two phases constitute a degenerate set of minimizers of the free energy. We can then write the posterior as a mixture distribution:

$$p(\boldsymbol{w} \mid \mathcal{D}) = p(\boldsymbol{w} \mid \mathcal{U})p(\mathcal{U} \mid \mathcal{D}) + p(\boldsymbol{w} \mid \mathcal{V})p(\mathcal{V} \mid \mathcal{D})$$

where $p(\boldsymbol{w} \mid \mathcal{U})$ is the posterior distribution conditioned on the model being in phase $\mathcal{U}$, and $p(\mathcal{U} \mid \mathcal{D})$ is the total posterior probability of this phase, with $\pi_{\mathcal{U}} + \pi_{\mathcal{V}} \approx 1$.

First-order phase transitions. This free energy formula predicts the existence of first-order "Bayesian" phase transitions. When two solutions compete, e.g., u with neighborhood $\mathcal{U}$ and v with neighborhood $\mathcal{V}$, then the posterior log-odds evolve as $\log \frac{p_n(\mathcal{U})}{p_n(\mathcal{V})} = n\Delta L_n + \log(n)\Delta\lambda + O_p(\log\log n)$, where $\Delta L_n = L_n(v) - L_n(u)$ and $\Delta\lambda = \lambda(v) - \lambda(u)$. If u has higher loss but lower complexity ($\Delta L_n < 0$, $\Delta\lambda > 0$), the posterior initially prefers the simple solution $\mathcal{U}$ but switches to the complex solution $\mathcal{V}$ when $\log(n)/n < -\Delta L_n/\Delta\lambda$. This is a "Type-A" transition (Carroll et al., 2025; Hoogland et al., 2025), see Figure 1.

Alternatively, if the two phases agree on the linear term (they have the same minimum loss), then the trade-off will be pushed down into lower-order terms, and there can be a "Type-B" transition, in which complexity (as measured by the LLC) *decreases*, in exchange for an increase in lower-order terms.

B Bayesian Influence Functions

Estimating the BIF with SGMCMC. To estimate the BIF in practice, we use an RMSProp-preconditioned SGLD sampler (Welling & Teh, 2011; Li et al., 2016) to estimate a posterior from each checkpoint $\boldsymbol{w}^*$ This approximates Langevin dynamics with loss gradients and locality regularization from initial $\boldsymbol{w}^*$ modulated by localization strength γ,

$$\boldsymbol{w}_{s+1} = \boldsymbol{w}_s - \frac{\hat{\epsilon_s}}{2}\left(\sum_{i\in|\mathcal{D}|} \nabla_{\boldsymbol{w}}\ell_i(\boldsymbol{w}_s) + \gamma(\boldsymbol{w}_s - \boldsymbol{w}^*)\right) + \mathcal{N}(0, \hat{\epsilon}_s). \tag{15}$$

The update rule takes advantage of an adaptive learning rate $\hat{\epsilon}_s$ compared to vanilla SGLD and is more robust to varying step size (Hitchcock et al., 2025). The full algorithm is described in Appendix B.

Practical considerations with the BIF. As mentioned in the main body and following the discussion at length in Kreer et al. (2025), we consider several practical modification of the local BIF. First, we use a normalized BIF for the language-modeling experiments, which involves computing the Pearson correlation instead of the covariance over losses. Empirically, we find that this behaves more stably than the raw covariance and thus is easier to track over time. Second, we consider a per-token BIF, which can be trivially obtained by avoiding the loss accumulation over token indices and saving all per-token losses at each SGLD draw. Finally, to avoid potentially spuriously high covariances, we drop same-token influence scores when computing aggregate group influence scores, as in Adam et al. (2025).

Addressing influence from unseen training samples. A potential objection to our developmental analysis is that at early checkpoints, the model's optimizer has not yet encountered every training sample. It might therefore seem paradoxical to measure the "influence" of a sample the model has not yet "seen."

This concern can be straightforwardly resolved. The BIF is defined as the sensitivity of an observable to a sample's contribution weight, β_i, to the total loss. For a sample $\mathbf{z}_i$ that has already been processed, we measure influence by considering perturbations around its baseline weight of $\beta_i = 1$. For a sample the optimizer has not yet encountered, we can simply treat its current weight as $\beta_i = 0$ and evaluate the same derivative at that point.

The resulting quantity remains well-defined, with a slightly different interpretation: it measures the model's sensitivity to the *initial introduction* of a new sample, rather than the *re-weighting* of an existing one. Our practical implementation naturally accommodates this, as we use independent data sources for the SGLD gradient updates and the forward passes used to compute the loss covariance.

Algorithm 1 RMSPropSGLD for Bayesian influence

```
**Input:** Initial model parameters $\boldsymbol{w}^* \in \boldsymbol{w}$, training dataset $\mathcal{D} = (\boldsymbol{z}_i)_{i=1}^{n=8}$, loss functions $\ell_i := \ell(\boldsymbol{z}_i; -) \colon \boldsymbol{w} \to \mathbb{R}$ for each $i \in [n]$, observables $\phi_j \colon \boldsymbol{w} \to \mathbb{R}$ for each $j \in [n]$, RMSPropSGLD hyperparameters $\epsilon$ (step size), $\beta$ (inverse temperature), $\gamma$ (localization), $m$ (sampling batch size), $C$ (number of chains), $T$ (chain length), $b$ (decay rate), $\alpha$ (stability constant).
**Output:** $\boldsymbol{B} = (\mathrm{BIF}(\boldsymbol{z}_i, \phi_j))_{1 \le i \le n, 1 \le j \le m} \in \mathbb{R}^{n \times m}$
$\boldsymbol{L} \leftarrow \mathbf{0}_{n \times CT}, \boldsymbol{\Phi} \leftarrow \mathbf{0}_{m \times CT}$
**for** $1 \le c \le C$ **do**
    $\boldsymbol{w} \leftarrow \boldsymbol{w}^*$
    **for** $1 \le t \le T$ **do**
        **for** $1 \le i \le n$ **do**
            $\boldsymbol{L}_{i,(c-1)C+t} \leftarrow \ell_i(\boldsymbol{w})$                ▷ Compute train losses
        **end for**
        **for** $1 \le j \le m$ **do**
            $\boldsymbol{\Phi}_{j,(c-1)C+t} \leftarrow \phi_j(\boldsymbol{w})$                ▷ Compute observables
        **end for**
        Sample full batch $\mathcal{B} = \mathcal{D}$ of size $m = n$ for small toy dataset.
        $\boldsymbol{V}_t \leftarrow b\boldsymbol{V}_{t-1}[i] + (1-b)\nabla_w \ell^2$
        $\hat{\boldsymbol{V}}_t \leftarrow \frac{1}{1-b^t}\boldsymbol{V}_t$
        $\hat{\epsilon}_t \leftarrow \frac{\epsilon}{\sqrt{\hat{\boldsymbol{V}}_t + \alpha}}$
        $\eta_t \leftarrow \eta \sim \mathcal{N}(0,1)$
        $\boldsymbol{w} \leftarrow \boldsymbol{w} - \frac{\hat{\epsilon}_t}{2}\left(\frac{\beta n}{m}\sum_{k \in \mathcal{B}_t} \nabla_{\boldsymbol{w}} \ell_k(\boldsymbol{w}) + \gamma(\boldsymbol{w} - \boldsymbol{w}^*)\right) + \sqrt{\hat{\epsilon}_t}\eta_t$                ▷ SGLD update
    **end for**
**end for**
$\boldsymbol{B} \leftarrow \frac{1}{CT-1}\boldsymbol{L}\left(\boldsymbol{I}_{CT} - \frac{1}{CT}\mathbf{1}_{CT}\mathbf{1}_{CT}^\top\right)^2 \boldsymbol{\Phi}^\top$                ▷ Covariance between $\boldsymbol{L}$ and $\boldsymbol{\Phi}$
**Return** $\boldsymbol{B}$
```

Scalability of BIF. Besides the theoretical advantages discussed in Section 2.2, the BIF also has several practical advantages over classical IF approximations, as discussed in Kreer et al. (2025).

First, classical approximations like EK-FAC (Grosse et al., 2023) incur a substantial up-front computational cost to "fit" the inverse Hessian estimate. The BIF, in contrast, has no upfront fitting cost. This comes at the cost of having a higher compute cost per individual query. This means that the BIF is more computationally efficient for smaller datasets, while the classical IF is more efficient for larger datasets, where it can amortize the upfront costs over many queries. For the settings we considered, where the focus is studying development, the priority was coverage over many checkpoints rather than coverage over the entire dataset. These tradeoffs favored the BIF as the more tractable choice.

Furthermore, our focus on *fine-grained* structural attribution places us in the specific regime where BIF is most efficient: computing dense, per-token influence matrices for targeted subsets of data. Unlike Hessian-based methods, which typically require sequential scoring passes to isolate each individual token contribution, the BIF estimator computes the full set of token-by-token influences in a single batched process without additional memory overhead for backpropagating individual per-token gradients.

C Toy Model of Development

In this section, we provide details of the toy model experimental details and analytical investigation of deep linear neural network learning dynamics under the data perturbation introduced in Section 3. It includes a training setup of toy data with a 2-layer deep linear neural network and BIF hyperparameter sweeps, and extended results of the Leave-One-Out (LOO) experiments and BIF measurements.

C.1 Toy Dataset

We use the semantic hierarchical dataset introduced in Saxe et al. (2019a). We set the number of data points $N = 8$ with hierarchy level $H = 3$, that is, the data can be organized by 4 levels of hierarchy, e.g., `dog` belongs to the lowest hierarchy level 1 of living organism, and level 2 of animals (vs. plants), and level 3 of mammals (vs. birds). The input is a data index given by an identity matrix of size $N \times N$. The model needs to learn to associate with the feature of each data point, which forms the above hierarchical structure, the output size of $N \times O$ with $O = 15$.

We train a 2-layer bias-free linear neural network with 50 hidden neurons, which is overparameterized for our rank 8 dataset. We initialize the network with small weights by sampling from a Gaussian distribution $\mathcal{N} \sim (0, \sigma^2),\ \sigma = 1e^{-3}$. We train the network on mean squared error (MSE) loss with a small learning rate $5e^{-3}$ and train for $10K$ epochs with full batch unless mentioned otherwise.

C.2 BIF Additional Results and Hyperparameters Sweep

For a principled decision of hyperparameters for the SGLD sampler, we perform a preliminary hyperparameter sweep on the SGLD sampler using the same deep linear neural network architecture and the data to estimate LLC. Since ground truth LLC is known for deep linear neural networks (Aoyagi, 2024), we can select a range of hyperparameters by choosing ones that have a well-matched LLC estimate to the ground truth. Based on this procedure, we validate the superiority of RMSPropSGLD to vanilla SGLD and also the insensitivity of the sampler quality to decay rate b and stability constant α in a certain range and fixed the value to reduce the grid search dimension. We also narrowed down the localization strength range.

The range of the conducted hyperparameter sweeps for BIF measurement is summarized in Table 1. The values given in the range were sampled on a logarithmic scale. RMSPropSGLD sampling procedure is shown in B.

Table 1: Summary of hyperparameter sweep range for BIF experiments on toy model.

Hyperparameter	Range
ϵ (step size)	[1e-7 - 1e-2]
β (inverse temperature)	[1e+1 - 1e+4]
γ (localization strength)	[1e-2 - 1e+6]
m (batch size)	8 (full batch)
C (number of chains)	[2,4,8]
T (chain length)	[200, 400, 800, 1000]
b (decay rate)	[0.8, 0.9, 0.95, 0.99]
α (stability constant)	[1e-4 - 1.0]

We present the Pearson correlation coefficient between Leave-One-Out (LOO) loss difference trajectory and BIF trajectory over learning over varying hyperparameter choices of inverse temperature β, localization strength γ and step size ϵ in Figure 6. In Figure 6, we see that the correlation coefficient changes smoothly as we vary the hyperparameters. In the main Section 3 Figure 3, we showed the BIF measured with the hyperparameters that have the highest correlation with LOO ($\beta = 1000, \epsilon = 1e\text{-}3, \gamma = 5e + 3$).

C.3 Leave-One-Out (LOO) Experiment

For the Leave-One-Out (LOO) experiment, which is also referred to as retraining, we mask the corresponding data index to 0 for both input and output. We use the same hyperparameter as above and we report the loss difference of data j after ablating data i at each time, $\Delta\ell_{j,t} = \ell_{j,t}^{\mathcal{D}} - \ell_{j,t}^{\mathcal{D}\setminus i}$.

C.4 Classical Influence Function

We measure classical influence over the training trajectory of the toy model

$$\mathrm{IF}_t(\mathbf{z}_i, \phi) = -\nabla_{\boldsymbol{w}_t}\phi(\boldsymbol{w}_t)^\top \boldsymbol{H}^{-1}(\boldsymbol{w}_t)\nabla_{\boldsymbol{w}_t}\ell_i(\boldsymbol{w}_t), \tag{16}$$

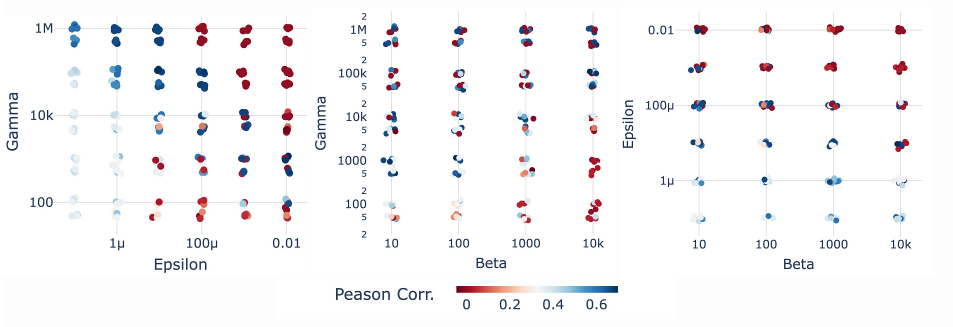


Figure 6: **BIF–LOO trace correlation with varying inverse temperature β, localization strength γ and step size ϵ.** On a single grid point between two hyperparameters (e.g. γ, ϵ in the leftmost), multiple points refer to different values of the remaining hyperparameter (β in the leftmost). In general, high γ and low ϵ gave the highest correlation with the gold standard LOO experiment, and β was less significant in the toy model.

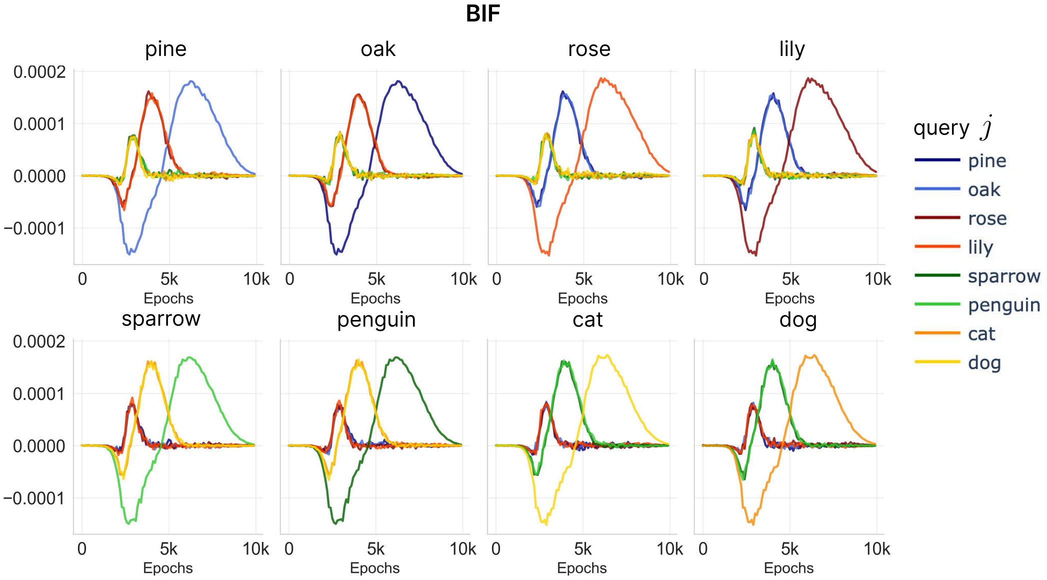


Figure 7: **BIF for perturbing each data point.** The subtitle indicates the perturbed data point.

at time t. Note that we do not use the global minimum $\boldsymbol{w}^*$ but the minimum at that specific measurement time t. Since the Hessian $\boldsymbol{H}$ of the overparameterized neural network is non-invertible due to multiple zero / near-zero eigenvalues, we take the approximation methods as in Koh & Liang (2020); Grosse et al. (2023). First is to add a constant dampening term γ to the Hessian,

$$\tilde{\boldsymbol{H}} = \boldsymbol{H} + \gamma^* \mathbf{I}. \tag{17}$$

Suitable γ enforces $\tilde{\boldsymbol{H}}$ to have positive eigenvalues. Grosse et al. (2023); Martens & Grosse (2015); Bae et al. (2022) used damped Gauss-Newton-Hessian (GNH), an approximation to $\boldsymbol{H}$ which linearizes the network's parameter-output mapping around the current parameters,

$$\boldsymbol{G} = \mathbb{E}[\mathbf{J}^\top \boldsymbol{H} \mathbf{J}] \tag{18}$$

$$\bar{\boldsymbol{H}} = \boldsymbol{G} + \lambda^* \mathbf{I}. \tag{19}$$

Using approximated damped-Hessian ($\tilde{\boldsymbol{H}}$) and damped GNH ($\bar{\boldsymbol{H}}$), we measure classical influence function over time (Eq. 16) with varying dampening constant γ.

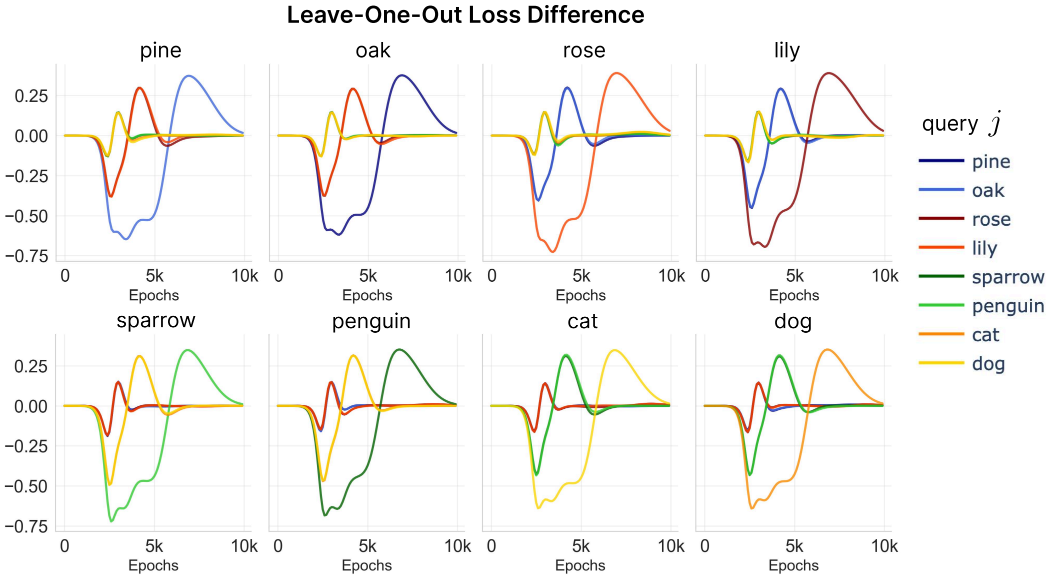


Figure 8: **Leave-One-Out (LOO) loss difference for perturbing each data point.** The title indicates the ablated data point.

The dampening constant γ^* is scaled with α, the maximum absolute value of all eigenvalues of the $\boldsymbol{H}$

$$\gamma^* = \gamma\alpha, \tag{20}$$

and we vary γ in range of $[1e\text{-}3, 10]$ in logarithmic scale.

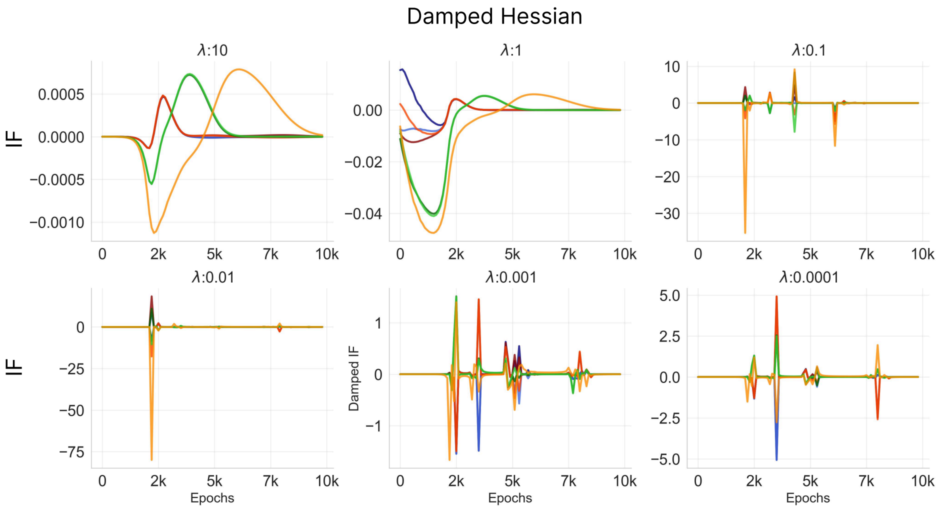


Figure 9: **Influence with damped Hessian approximation.** We measure the classical influence of the `dog` sample with damped Hessian approximation $\tilde{\boldsymbol{H}}$ with varying dampening constant γ.

C.5 Time-Specific Ablation Experiment

Our observation that influence is a time-dependent function proposes another perspective – perturbing the same data point but at different training times would lead to non-identical influence. That is, the exact moment the network interacts with each data point matters. Here we probe this hypothesis with a simple retraining experiment in the toy model. We hypothesize that the measured influence

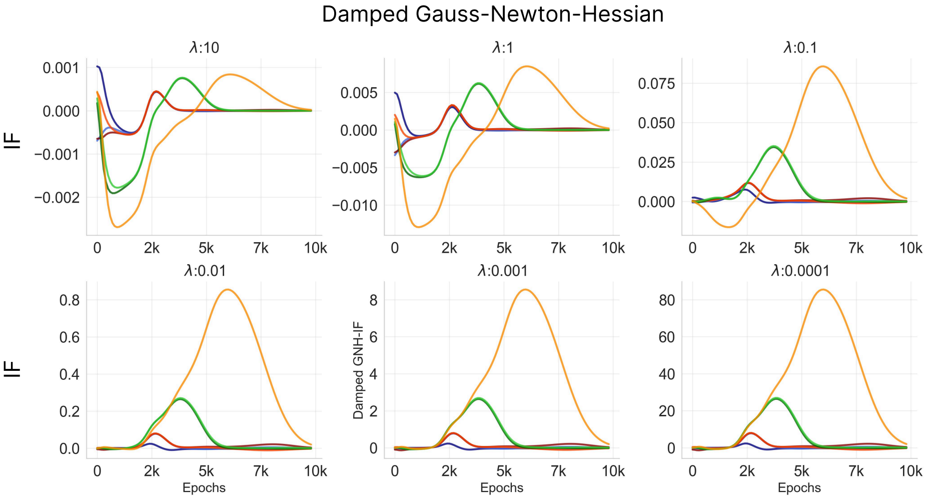


Figure 10: **Influence with damped Gauss-Newton-Hessian approximation.** We measure the classical influence of the dog sample with damped Gauss-Newton-Hessian approximation $\tilde{\boldsymbol{H}}$ with varying dampening constant γ.

trajectory tells us which time point would be most critical for a model to learning each sample. For example, the influence of dog on cat makes a peak around $t = 6000$, which implies ablating dog around that time would be most helpful in learning cat. We probe this hypothesis with a retraining experiment with a brief ablation at different stages of learning.

In Figure 11, we ablate a data point dog during the duration of D epochs from t and show the integrated loss difference compared to the baseline (no ablation) during D. As we expected, the loss difference is the biggest around $t = 6000$, when the highest influence occurs, indicating ablating dog was helpful the most at this stage. Similarly, we observe the negative peak of integrated loss difference around $t = 3000$, which also matches the negative minimum in the influence measurement, implying that learning of the cat was the most detrimental at this time point when dog is ablated. We show that the most important stage changes with the query data that belongs to a different hierarchy, e.g., sparrow. Collectively, we strengthen our claim that influence over time shows us how a sample interacts with “what” data “when”, and that it is often correlated with the structure of the data, such as hierarchy.

C.6 Analytical Investigation of Toy Model

In this section, we analytically derive the dynamics of the influence function in a deep linear neural network. First, we introduce deep linear neural network learning dynamics studied in Saxe et al. (2013; 2019b) and then we show the time dependency of the influence function and its relation to progressive learning of the hierarchical structure.

Deep Linear Neural Network Learning Dynamics

Following the singular modes learning formulation in Saxe et al. (2019a; 2013), we can describe each data point i's loss into the singular modes basis. We assume whitened input $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, $\boldsymbol{X}^T\boldsymbol{X} = I$ and output $\boldsymbol{Y} \in \mathbb{R}^{N \times O}$. The input-output correlation and its singular value decomposition (SVD) become

$$\mathbf{C} := \Sigma_{yx} = \boldsymbol{Y}^T\boldsymbol{X} \tag{21}$$

$$SVD(\Sigma_{yx}) = \mathbf{U}\mathbf{S}\mathbf{V}^\top. \tag{22}$$

With squared error loss, individual data point at data i is $\ell_i = \frac{1}{2}\|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2$. We project data $\mathbf{x}_i$ to $\boldsymbol{V}$ space (object analyzer) and its component becomes $\varepsilon_{ik} = \mathbf{x}_i\boldsymbol{V}_k$, where $\boldsymbol{V}$ is a column vector of $\boldsymbol{V}$. Similarly, we project $\eta_{ik} = \mathbf{y}_i\boldsymbol{U}_k$ and with rank $r = \text{rank}(\mathbf{C})$,

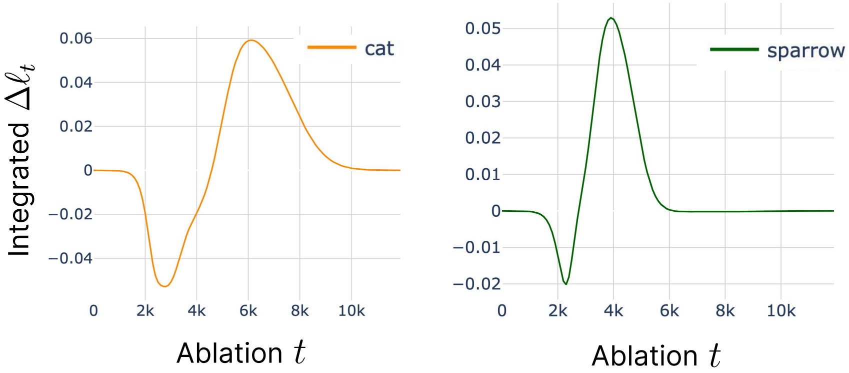


Figure 11: **Brief ablation of `dog`.** For each experiment, we ablate `dog` for duration $D = 100$ epochs starting from timepoint t. Starting time t is sampled uniformly with an interval of 200 epochs. For each experiment, we integrate the loss difference to the baseline during D. We show the integrated loss of query (a) `cat` and (b) `sparrow` over different ablation windows. The most critical period of influence from the `dog` on the two samples is different. The higher level hierarchy (animals) sharing sample `sparrow` has an earlier critical point than the lower level hierarchy (mammals) sharing sample `cat`.

$$\ell_i = \frac{1}{2}\sum^{r}(\eta_{ir} - g_r\varepsilon_{ir})^2, \tag{23}$$

g_k is a singular mode strength of mode k of the network obtained from

$$\boldsymbol{W}(t) = \boldsymbol{W}_2(t)\boldsymbol{W}_1(t) = \boldsymbol{U}\boldsymbol{G}(t)\boldsymbol{V}^\top, \tag{24}$$

assuming the network and the data shares the aligned right and left singular vectors $\boldsymbol{U}, \boldsymbol{V}$. This assumption holds with small initialization and small learning rate.

Now, we can compute the loss gradient with respect to the parameter space and decoupled mode space. In parameter space,

$$\nabla L_{\boldsymbol{W}_1} = \boldsymbol{W}_2^\top(\Sigma_{yx} - \boldsymbol{W}_2\boldsymbol{W}_1) \tag{25}$$

$$\nabla L_{\boldsymbol{W}_2} = (\Sigma_{yx} - \boldsymbol{W}_2\boldsymbol{W}_1)\boldsymbol{W}_1^\top. \tag{26}$$

We can substitute the terms in singular decomposed form,

$$\nabla L_{\boldsymbol{W}_1} = \boldsymbol{U}(\boldsymbol{S} - \boldsymbol{G})\boldsymbol{A}_1 \tag{27}$$

$$\nabla L_{\boldsymbol{W}_2} = \boldsymbol{A}_2(\boldsymbol{S} - \boldsymbol{G})\boldsymbol{V}^\top \tag{28}$$

$$\boldsymbol{W}_1 = \mathbf{R}\boldsymbol{A}_1\boldsymbol{V}^T, \boldsymbol{W}_2 = \boldsymbol{U}\boldsymbol{A}_2\mathbf{R}^T, \boldsymbol{A}_2\boldsymbol{A}_1 = \boldsymbol{G}. \tag{29}$$

The effective singular values from each layer $\boldsymbol{W}_{1,2}$ are $\boldsymbol{A}_{1,2}$ and $\mathbf{R}^T\mathbf{R} = I$, which eventually aligns the effective weights' singular vectors to that of the data correlation Σ_{yx}. In general, we cannot assume $a_{1k} = a_{2k}$ throughout learning, but the equality holds with the strictly balanced initialization. With the small initialization assumption, the first-order approximation of equality holds, which eventually allows us to write down the effective singular value trajectory throughout learning as in Saxe et al. (2019a).

Importantly, $\boldsymbol{G}(t)$ is a time-varying variable, evolving throughout the training that could be written in closed form with s_k being corresponding diagonal singular value of $\boldsymbol{S}$ from data covaraince,

$$\boldsymbol{G}_{kk}(t) = \frac{s_k e^{2s_k t/\tau}}{e^{2s_k t/\tau} - 1 + s_k/\boldsymbol{G}^0_{kk}}, \; \boldsymbol{G}^0_{kk} = \boldsymbol{G}_{kk}(t=0), \tag{30}$$

and the model that learned the training data recovers

$$\boldsymbol{W}^* = \mathbf{UGV} = \mathbf{C}. \tag{31}$$

For convenience, we will drop the notation of time dependence in the rest of this section. For an end-to-end linear map $\boldsymbol{W} = \boldsymbol{W}_2\boldsymbol{W}_1 \in \mathbb{R}^{O\times D}$ define its representation in the data singular basis and the singular mode strength g_k,

$$\boldsymbol{G} \;:=\; \boldsymbol{U}^\top \boldsymbol{W}\boldsymbol{V}, \qquad g_k \;:=\; \boldsymbol{G}_{kk} \;=\; u_k^\top \boldsymbol{W} v_k. \tag{32}$$

For a data point $p = (x_p, y_p)$ write the coordinates in the *original* basis

$$\gamma_p := \boldsymbol{V}^\top x_p \in \mathbb{R}^r, \qquad \eta_p := \boldsymbol{U}^\top y_p \in \mathbb{R}^r, \tag{33}$$

the per-sample residual in mode space

$$\phi_p := \boldsymbol{G}\,\gamma_p - \eta_p \tag{34}$$

and the per-sample loss gradient in mode space

$$\ell_p := \phi_p \gamma_p^\top. \tag{35}$$

INFLUENCE FUNCTION IN TOY MODEL USING SINGLE DATA POINT PERTURBATION

Following Cook & Weisberg (1982); Koh & Liang (2020), we will derive *influence function*, that is, the change in model prediction with a targeted perturbation of training data. In the following, we will derive the response of original $\boldsymbol{U}, \boldsymbol{V}$ and $\boldsymbol{S}$ as a function of a single data point perturbation to derive the influence function in singular basis. With those in hand, we can use the closed-form dynamics on Eq. 30 to get time-dependent dynamics.

First, we consider a perturbation by upweighting a single data point p with factor ε. The perturbed cross-covariance becomes

$$\mathbf{C}_p(\varepsilon) = \boldsymbol{U}_p(\varepsilon)\boldsymbol{S}_p(\varepsilon)\boldsymbol{V}_p(\varepsilon)^\top = \sum_{p\neq j} \mathbf{y}_j\mathbf{x}_j^\top + (1+\varepsilon)\mathbf{y}_p\mathbf{x}_p^\top = \mathbf{C} + \varepsilon \mathbf{y}_p\mathbf{x}_p^\top. \tag{36}$$

Let $\mathbf{C}'$ be the first-order derivative of the data covariance with ε-upweighting of a single data point p,

$$\mathbf{C}'_p := \left.\frac{\partial \mathbf{C}_p(\varepsilon)}{\partial \varepsilon}\right|_{\varepsilon=0}. \tag{37}$$

With the perturbed dataset, the learned 2-layer linear network mapping at time t becomes

$$\boldsymbol{W}_p(\varepsilon, t) = \boldsymbol{U}_p(\varepsilon)\boldsymbol{G}_p(\boldsymbol{S}_p(\varepsilon), t)\boldsymbol{V}_p^\top(\varepsilon), \tag{38}$$

$$\boldsymbol{G}_t(\boldsymbol{S}, t) = \text{diag}(g_t(s_1), ...g_t(s_r)) \tag{39}$$

where s is a singular value from diagonals of $\boldsymbol{S}$ and $g_t(\cdot)$ follows a closed form dynamics as given in Eq. 30. We assume that the network will have aligned singular basis with the perturbed data covariance, given in $\boldsymbol{U}(\varepsilon)$ and $\boldsymbol{V}(\varepsilon)$. For convenience, we will move the parameter ε to a subscript and drop the perturbed data index p.

We apply product rule to $\mathbf{C}'_\varepsilon$,

$$\mathbf{C}'_\varepsilon = \boldsymbol{U}'_\varepsilon \boldsymbol{S}\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{S}'_\varepsilon\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}_\varepsilon'^\top. \tag{40}$$

We project the above perturbation to the singular mode coordinates,

$$\boldsymbol{Q}_\varepsilon := \boldsymbol{U}^\top \mathbf{C}'_\varepsilon \boldsymbol{V} = \boldsymbol{U}^\top\Big(\boldsymbol{U}'_\varepsilon \boldsymbol{S}\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{S}'_\varepsilon\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}_\varepsilon'^\top\Big)\boldsymbol{V}. \tag{41}$$

Since $\boldsymbol{U}, \boldsymbol{V}$ are orthonormal matrices,

$$\boldsymbol{U}^\top\boldsymbol{U} = \mathbf{I},\ \boldsymbol{V}^\top\boldsymbol{V} = \mathbf{I} \tag{42}$$

and differentiating these orthonormality constraints with respect to the perturbation ε

$$\frac{\partial}{\partial\varepsilon}\Big(\boldsymbol{U}^\top\boldsymbol{U}\Big)\Big|_{\varepsilon=0} = \boldsymbol{U}_\varepsilon'^\top\boldsymbol{U} + \boldsymbol{U}^\top\boldsymbol{U}'_\varepsilon = 0. \tag{43}$$

$$\frac{\partial}{\partial\varepsilon}\Big(\boldsymbol{V}^\top\boldsymbol{V}\Big)\Big|_{\varepsilon=0} = \boldsymbol{V}_\varepsilon'^\top\boldsymbol{V} + \boldsymbol{V}^\top\boldsymbol{V}'_\varepsilon = 0. \tag{44}$$

Due to orthonormality, $\boldsymbol{U}_\varepsilon', \boldsymbol{V}_\varepsilon'^\top$ lie in the tangent space of $\boldsymbol{U}, \boldsymbol{V}^\top$. We can define

$$\boldsymbol{A} := \boldsymbol{U}^\top \mathrm{U}_\varepsilon{}', \; \boldsymbol{B} := \boldsymbol{V}_\varepsilon'^\top \boldsymbol{V}, \tag{45}$$

where the orthonormal constraints give

$$\boldsymbol{A}^\top + \boldsymbol{A} = 0, \;\; \boldsymbol{B} + \boldsymbol{B}^\top = 0, \tag{46}$$

and $\boldsymbol{A}$,$\boldsymbol{B}$ become skew-symmetric, $\boldsymbol{A} = -\boldsymbol{A}^\top, \; \boldsymbol{B} = -\boldsymbol{B}^\top$. Intuitively, $\boldsymbol{A}, \boldsymbol{B}$ is equivalent to the in-subspace angular velocity (rotation rate) of the columns of $\boldsymbol{U}$ and $\boldsymbol{V}^\top$ due to the perturbation, respectively.

With this, we can rewrite Eq. 41

$$\boldsymbol{Q} = \mathbf{AS} + \boldsymbol{S}_\varepsilon' + \mathbf{SB}, \tag{47}$$

where $\mathrm{diag}(\boldsymbol{A}) = \mathrm{diag}(\boldsymbol{B}) = 0$ since $\boldsymbol{A}, \boldsymbol{B}$ are both skew symmetric.

Now, we solve Eq. 47 for $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{S}_\varepsilon'$. First, we consider a case where the singular values $\mathrm{diag}(\boldsymbol{S})$ are non-degenerate. Due to the diagonal constraint of singular values, the response from the perturbation should also be diagonal for the singular values

$$\boldsymbol{S}_{\varepsilon,ii}' = \boldsymbol{Q}_{\varepsilon,ii} = \boldsymbol{U}_i^\top \mathbf{C}_\varepsilon' \boldsymbol{V}_i. \tag{48}$$

The response of the singular basis is obtained from Eq. 45,

$$\boldsymbol{U}_\varepsilon' = \mathbf{UA}, \;\; \boldsymbol{V}_\varepsilon'^\top = \mathbf{BV}^\top, \tag{49}$$

where

$$\boldsymbol{A}_{jk} = \begin{cases} \frac{\boldsymbol{U}_j^\top \mathbf{C}_\varepsilon' \boldsymbol{V}_k}{s_k - s_j}, & j \neq k, \\ 0, & j = k. \end{cases} \tag{50}$$

$$\boldsymbol{B}_{kj} = \begin{cases} \frac{\boldsymbol{U}_k^\top \mathbf{C}_\varepsilon' \boldsymbol{V}_j}{s_k - s_j}, & j \neq k, \\ 0, & j = k. \end{cases} \tag{51}$$

When $\boldsymbol{S}$ has degenerate singular values or a near-zero gap, above is ill-defined as the denominator $s_k - s_j = 0$. In this case, we apply block perturbation, treating the whole invariant subspace (sharing the degenerate singular values) all at once.

We take the sets of the degenerate singular values $\{s_b : s_j = s_b, \forall s_j \in \mathrm{diag}(\boldsymbol{S})\}$, define the block coupling by projecting the perturbation $\mathbf{C}'$ on this block,

$$\bar{\boldsymbol{Q}}_\varepsilon := \bar{\boldsymbol{U}}^\top \mathbf{C}_\varepsilon' \bar{\boldsymbol{V}}, \tag{52}$$

where $\bar{\boldsymbol{U}}$ and $\bar{\boldsymbol{V}}^\top$ are sets of the degenerate singular values corresponding to sets of columns of $\boldsymbol{U}$ and $\boldsymbol{V}$. It can be further decomposed into a symmetric and a symmetric-skewed part,

$$\bar{\mathbf{M}}_\varepsilon = \frac{1}{2}(\bar{\boldsymbol{Q}}_\varepsilon + \bar{\boldsymbol{Q}}_\varepsilon^\top), \;\; \bar{\mathbf{K}}_\varepsilon = \frac{1}{2}(\bar{\boldsymbol{Q}}_\varepsilon - \bar{\boldsymbol{Q}}_\varepsilon^\top), \;\; \bar{\mathbf{M}}_\varepsilon + \bar{\mathbf{K}}_\varepsilon = \bar{\boldsymbol{Q}}_\varepsilon. \tag{53}$$

Due to diagonality constraints of singular values and zero-diagonals in the skew-symmetric part $\bar{\mathbf{K}}_\varepsilon$, the first-order singular value consistency reduces to $\bar{\mathbf{M}}_\varepsilon = \bar{\boldsymbol{G}}_\varepsilon'$. To have diagonal singular value matrix $\bar{\boldsymbol{S}}_\varepsilon'$, we choose $\mathbf{R}$ such that

$$\bar{\mathbf{M}}_\varepsilon = \bar{\boldsymbol{S}}_\varepsilon' = \mathbf{R}^\top \Lambda \mathbf{R}, \tag{54}$$

where $\mathrm{diag}(\Lambda)$ becomes the first-order splits within the degenerate block with preferred direction defined by $\mathbf{R}$. Skewed part $\bar{\mathbf{K}}_\varepsilon$ sets the in-block angular velocity,

$$\bar{\boldsymbol{A}} = \bar{\boldsymbol{B}} = \frac{1}{2s_b} \bar{\mathbf{K}}_\varepsilon. \tag{55}$$

Then,

$$\bar{\boldsymbol{S}}_{\varepsilon,ii}' = \Lambda_{ii}, \;\; \bar{\boldsymbol{U}}_\varepsilon' = \bar{\boldsymbol{U}} \bar{\boldsymbol{A}}, \;\; \bar{\boldsymbol{V}}_\varepsilon' = \bar{\boldsymbol{V}} \bar{\boldsymbol{B}}. \tag{56}$$

Now we are equipped with $\boldsymbol{U}'_\varepsilon, \boldsymbol{V}'_\varepsilon, \boldsymbol{S}'_\varepsilon$ described by original we define influence on $\boldsymbol{W}(t,\varepsilon)$ from the perturbation on a data point p at fixed time t by applying derivative on Eq. 38,

$$\mathcal{I}_p := \frac{\partial \boldsymbol{W}}{\partial \varepsilon}\bigg|_{\varepsilon=0} = \boldsymbol{U}'_\varepsilon \boldsymbol{G}(\boldsymbol{S},t)\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{G}(\boldsymbol{S}'_\varepsilon,t)\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{G}(\boldsymbol{S},t)\boldsymbol{V}'^\top_\varepsilon \tag{57}$$

$$= \mathbf{UA}\boldsymbol{G}(\boldsymbol{S},t)\boldsymbol{V}^\top + \boldsymbol{U}(\text{diag}(g'_t(s)) \odot \text{diag}(\boldsymbol{S}'_\varepsilon)))\boldsymbol{V}^\top + \boldsymbol{U}\boldsymbol{G}(\boldsymbol{S},t)\mathbf{BV}^\top. \tag{58}$$

We can further derive influence on measurable ϕ_i in singular basis using the chain rule,

$$\mathcal{I}(z_p,\phi) = \left\langle r_i x_i^\top, \frac{\partial \boldsymbol{W}}{\partial \varepsilon} \right\rangle = r_i^\top \left(\frac{\partial \boldsymbol{W}}{\partial \varepsilon} x_i \right) = r_i^\top \boldsymbol{U}(\mathbf{A}\mathbf{G}_t + \boldsymbol{G}'_{\varepsilon,t} + \boldsymbol{G}_t \boldsymbol{B})\boldsymbol{V}^\top x_i, \tag{59}$$

where ϕ_i is a squared loss of data point i and z_p is the perturbed data point. This perturbation is expected to hold with a small perturbation $\varepsilon \approx 0$. We use down-weighting $\varepsilon = -0.1$ to approximate the ablation effect in Figure 3.

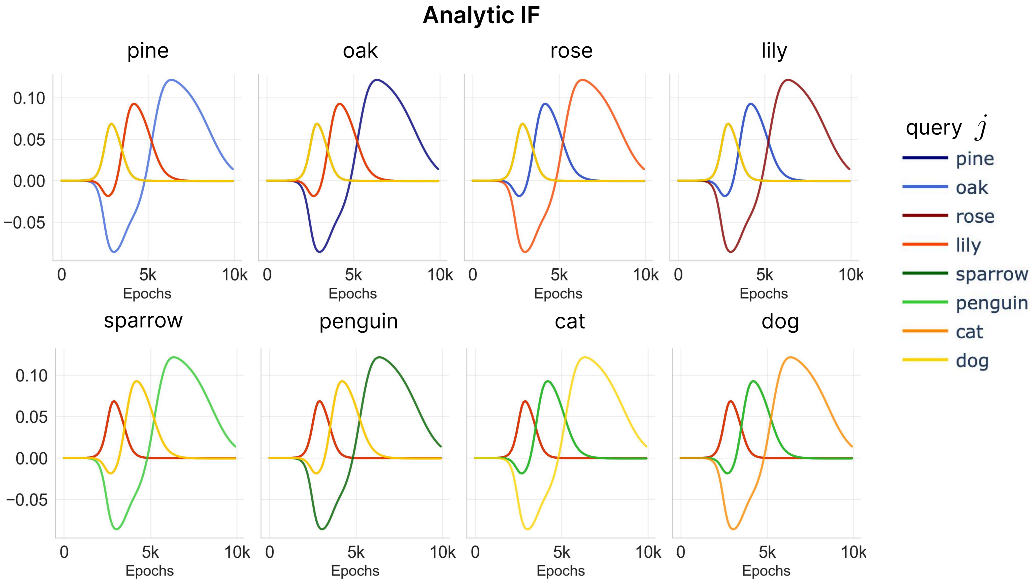


Figure 12: **Analytical loss difference from perturbing each data point.** The title indicates the perturbed data point ($\varepsilon = -0.1$).

D Language Models

This section discusses additional experimental details for the language model experiments.

D.1 Structural Token Classification

Here, we present additional details and further experiments conducted to investigate the development of influence with respect to how tokens structure text. These experiments were conducted using the 14 million parameter Pythia model. Tokens are generated using the same tokenizer as the Pythia models use in order to be able to use these tokens with the model suite.

D.1.1 Token Classes

Tokens are classified based on their role in structuring text. The classes we used are based on the classes used by Baker et al. (2025). Figure 13 demonstrates these classes graphically, with tokens outlined in bold indicating class membership, and distinct colors representing distinct classes.

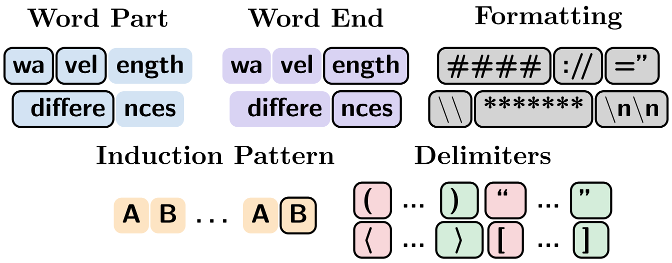


Figure 13: **Examples of structural classification of tokens.** Bold outline indicates class membership.

The classes are as follows:

Word Part Tokens that constitute a part of a word that is not the end of the word. This class excludes tokens that represent entire words or entire words with the addition of additional characters (e.g. a space and an entire word).

Word End Tokens that constitute the ending of a multi-token word. This class also excludes tokens that represent entire words and entire words with the addition of additional characters.

Formatting Tokens that are generally used to format text including newlines and repeated characters.

Induction Pattern Tokens that appear in a context in which the token is preceded by another particular token, and the same token has been preceded by the same other token earlier in the context.

Left Delimiters Tokens that are generally eventually followed by an equivalent right delimiter with interceding text generally considered to occupy a distinct scope.

Right Delimiters Tokens that close the scope opened by a paired left delimiter.

D.1.2 RMSPropSGLD Hyperparameters

Table 2: Summary of hyperparameter grid search for BIF experiments on Pythia 14M.

Hyperparameter	Range
n (number of sequences)	600
J (sequence length)	55
ϵ (step size)	$1\mathrm{e}-6$
$n\beta$ (inverse temperature)	$\{256, 1024\}$
γ (localization strength)	$\{500, 1000\}$
m (batch size)	64
C (number of chains)	4
T (chain length)	200
burn in steps	0
k (nearest neighbors)	30

We found that the overall dynamics expressed by the BIF did not vary substantially among hyperparameters chosen in this setting. Plots in the main paper were selected based on the parameters that optimized the average class recall on the KNN experiment which is discussed in section Appendix D.1.3

D.1.3 Same-class prediction with BIF K-Nearest Neighbors

One way to investigate the ability of the BIF to capture what sorts of text structure a model has learned is to use it to predict which structural class a token belongs to based on the classes that token shares high influence with.

We query the predictive capacity of token-level influences using a simple nearest-neighbors approach. The predictive model is as follows:

$$y_i(x_i) = \text{argmax}_a \frac{1}{r_a(x_i)} \sum_{x_j \in m_i} \mathbf{1}_a(x_j),$$

where $\mathbf{1}_a$ is the indicator function for elements of class a, and r_a is the token-dependent class frequency in the dataset:

$$r_a(x) = \frac{1}{\sum_{x'} \mathbf{1}_{\{x\}^c}(x')} \sum_{x'} \mathbf{1}_{\{x\}^c}(x')\mathbf{1}_a(x'),$$

with $\{x\}^c$ denoting the set of all tokens that are not x. Tokens are thus predicted to have the majority label of their corresponding top-influence set m_i, adjusted for class rates overall.

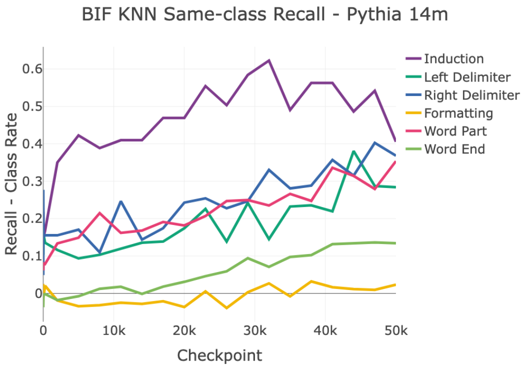


Figure 14: **KNN BIF predictions change over time.** Using the BIF in order to select the top 30 highest influence tokens and taking the majority class from among them (adjusted for class rates), we plot the recall minus the class rate for each class. This means that 0 represents the random baseline. We see that the BIF selects for in-class tokens above random for all categories by the end of training, indicating that it is sensitive to the class structure learned by the model.

Figure 14 shows the results of this experiment. We observe that the classes of high-influence other tokens act as a better than random predictor for all classes by the end of training, with recall generally improving as the model progresses through training.

D.1.4 Influence Dynamics Early in Training

Wang et al. (2025a) demonstrates that significant changes in model dynamics occur early in training for semantic-based token classes like the ones that we use in Section 4. In order to evaluate these effects, we estimate the BIF at logarithmically spaced model checkpoints, in contrast with the linear ones demonstrated in Figure 5. The results of this investigation are shown in Figure 15

We observe the inflection in influence between induction tokens noted previously (line at 32k training steps), but furthermore, observe additional changes in direction and even sign of the BIF occurring at early stages in training that were not observed in Figure 5. There is an additional inflection point in both the induction-induction BIF and formatting-formatting BIF at 128 training steps. We also more clearly see the early dynamics between right delimiters and tokens capturing the ends of words, with an inflection point at 2048 training steps. Furthermore, we observe clear changes in the overall sign of the BIF both between formatting-formatting tokens and between word parts and right delimiters. Taken together, these results indicate stagewise changes in the BIF, where, on average, tokens from a particular class can have differing effects on the loss of tokens both within the same class, and across classes, depending on when in training these effects are considered.

D.2 Per-Token Influence Dynamics

In Figure 16, we observe that different subgroups of tokens impose different magnitudes and signs of influence, which again can change over time. We see a few interpretable features. For example, we

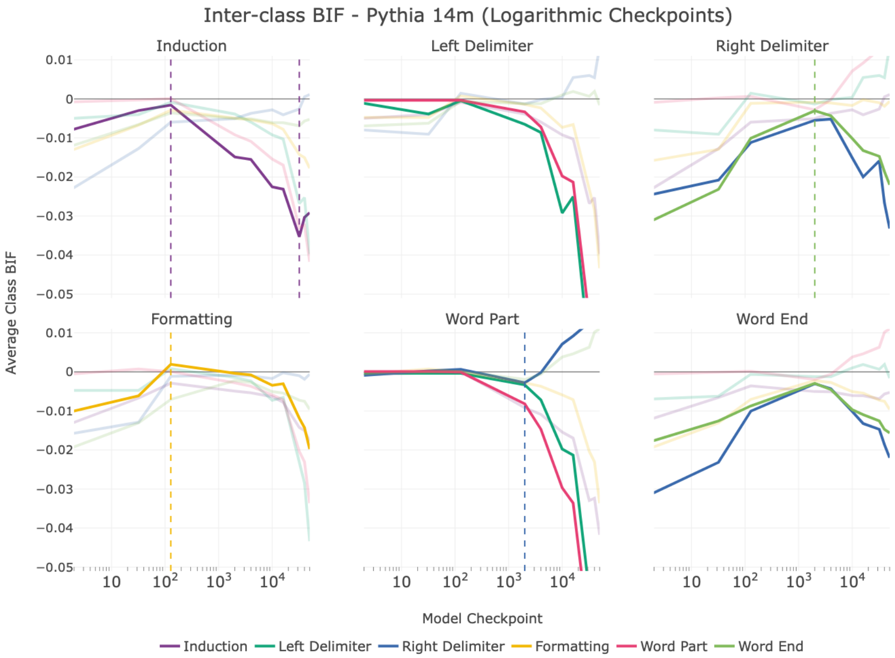


Figure 15: **Token-class relationships at logarithmic checkpoints.** In early stages of learning, we observe dynamic relationships between token classes, and non-monotonicity in the BIF. Dashed lines represent points at which the direction of changes to the BIF across training inverts.

see that the same subtypes of brackets can assist learning as training goes by, even if the directionality is different (observed in curly bracket, square bracket, and angle bracket but not in parenthesis), which might be due to the paired nature of the bracket signs.

D.3 Influence Dynamics of Induction Head Formation

Our initial experiments with the public Pythia checkpoints (Biderman et al., 2023) suggest a signal for the learning of induction patterns. However, the checkpoint frequency was too coarse in the vicinity of checkpoints 1k to 10k to observe the fine-grained dynamics of this developmental transition. To investigate this critical phase in more detail, we trained a Pythia-14M model from scratch, saving checkpoints every 100 steps for the first 20,000 training steps. Though we train on the same Pile data (Gao et al., 2021), it is in a different order and on different hardware, which means this training run is not the same as the original Pythia-14M training run.

We constructed a synthetic dataset with sequences where the second half repeats the first, e.g., *Sequence 1*: `A B`. . . `A B` and *Sequence 2*: `C D`. . . `C D`. Sequences were constructed as to not share tokens. The first half of each sequence is made up of random tokens, and so the only structure that can be used for prediction is the repetition of the second half of the context.

During SGLD we *collect* losses on this dataset, but *sample* using the Pile, meaning that loss on these synthetic samples does not affect our sampling trajectory. We compute the BIF matrices at each checkpoint of our homemade Pythia-14M and then look at the mean correlation between the different parts of our sequence (Namely repeated tokens with repeated tokens from other sequences, non-repeated tokens with non-repeated tokens from other sequences, and non-repeated tokens with repeated tokens from other sequences).

The results are shown in Figure 17. The influence between tokens in the *repeated* segments of each sequence (top panel, blue line) undergoes a sudden, large increase, peaking and then stabilizing. Meanwhile, the BIF between non-repeated segments or across non-repeated and repeated segments shows no such change.

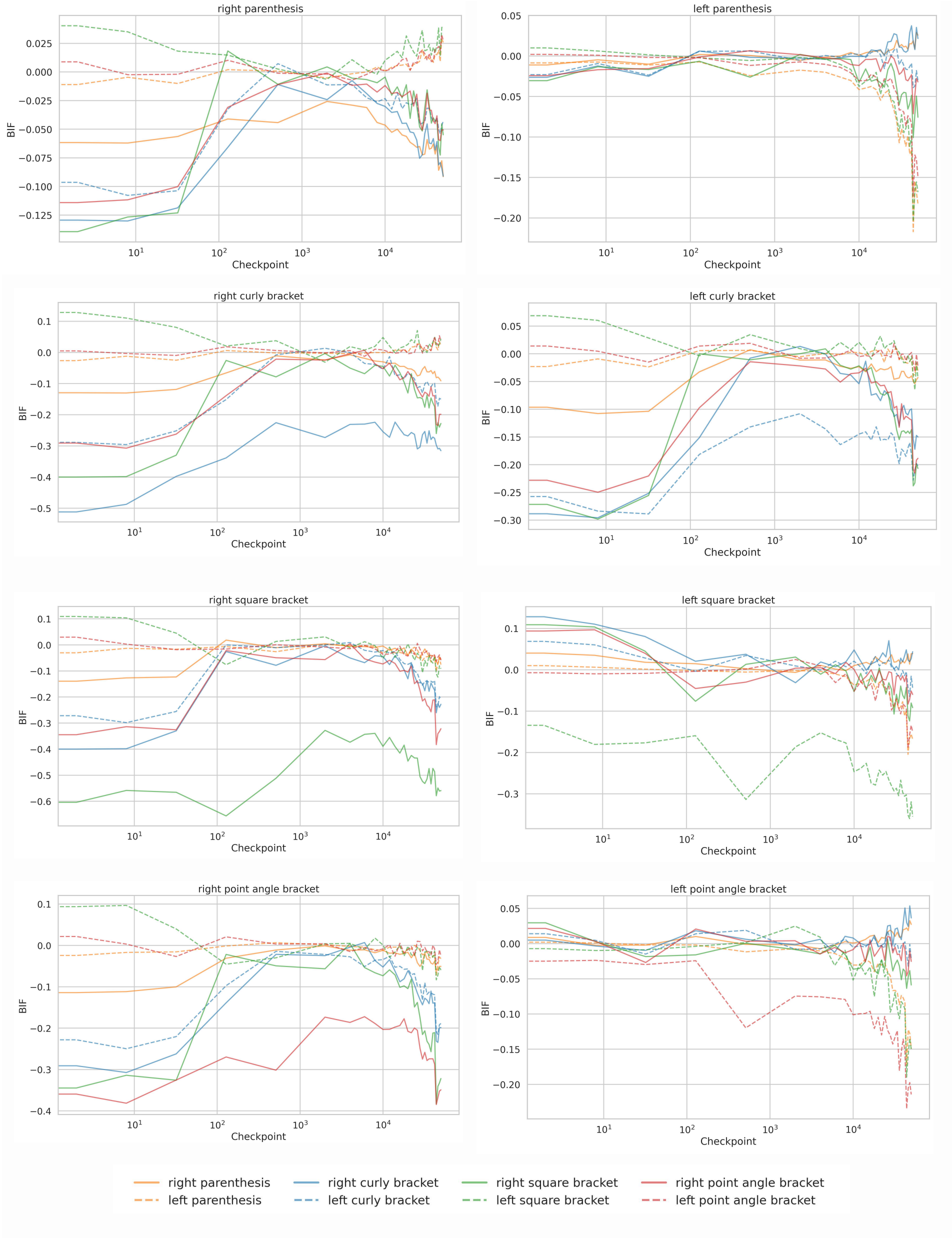


Figure 16: **Intra-classes (delimiters subtype) BIFs in Pythia-14M.** We plot BIFs between more fine-grained syntactic token classes of delimiters.

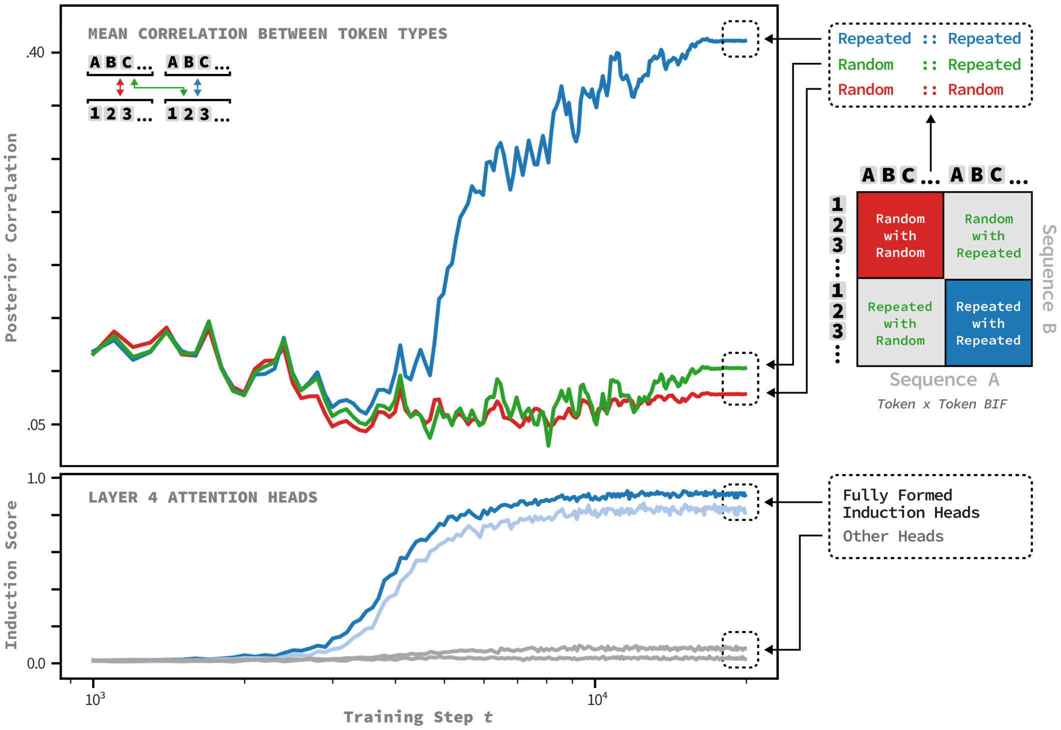


Figure 17: **Influence dynamics during induction head formation.** We trained a small transformer from scratch with high-frequency checkpointing to capture the formation of induction heads. (Top) The normalized BIF between corresponding tokens in repeated sequences (blue) shows a sharp increase and peak, while correlations between non-repeated samples (red) or between non-repeated and repeated segments (green) remain low. (Bottom) The induction score for the attention heads that become induction heads rises together with the posterior correlation, while other heads remain inactive.

Simultaneously, we measure the "induction score" of the model's attention heads—a standard metric from mechanistic interpretability that quantifies how strongly a head implements the induction algorithm (Olsson et al., 2022). As shown in the bottom panel, the induction score for the heads that become the "induction heads" begins to rise right before the BIF increases between the repeated groups.

D.4 Intervening in Induction Head Formation

To validate the causal implications of stagewise influence, we conducted an intervention experiment targeting the acquisition of induction (Olsson et al., 2022). Following the results from the previous section that show that the influence of induction patterns can vary significantly over training, we predict that upweighting the same subset of data should have different effects on model development depending on *when* that upweighting occurs. In particular, we expect upweighting induction samples will have a larger effect on model behavior if this occurs during or after the "induction bump."

Setup. We trained a 3 million parameter transformer on 13 million tokens from The Pile for a total of 30,000 steps. To isolate the effect of timing, we created five experimental runs where we upweighted "induction-heavy" tokens (defined as tokens that can be predicted from in-context bigram and multigram statistics) by a factor of 4× during specific 2500-step windows: 0–2.5k, 2.5k–5k, 5k–7.5k, 7.5k–10k, 10k–12.5k, and 12.5k–15k. We compared these against a control model trained with no upweighting. All models shared identical initialization and data ordering; thus, deviations in dynamics are strictly attributable to the timing of the intervention.

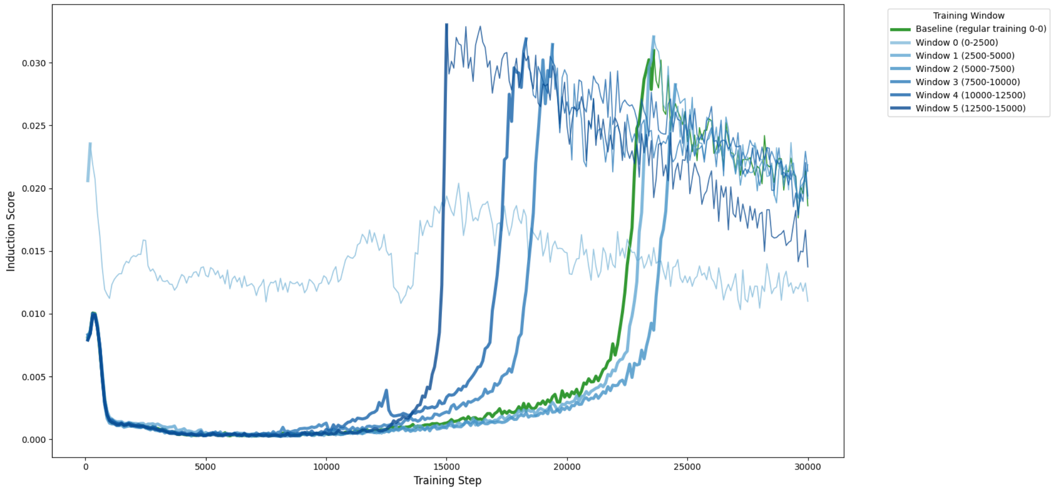


Figure 18: **Stagewise intervention accelerates induction head formation.** A comparison of the maximum induction score across Layer 0 heads for each intervention window. The baseline model (green) naturally forms induction heads at step ∼23k. Upweighting induction data during the critical developmental window (Windows 4 and 5, dark blue lines) significantly accelerates this capability, causing heads to form as early as step 15k. In contrast, interventions applied too early (e.g., Window 0) fail to produce stable induction heads, while intermediate windows have diminishing effects. This confirms the prediction that the model is most sensitive to induction data during specific developmental stages.

We track the formation of the induction circuit using the induction score metric described in Appendix D.3. We track the induction for each individual head in Figure 19 and compare the induction scores for each model's highest-scoring head in Figure 18.

Results. We find that the timing of the upweighting window significantly alters the developmental trajectory of the induction circuit, confirming that the influence of data is stage-dependent. In the control setting, the induction circuit emerges gradually, with the induction score peaking relatively late in the training process at timestep 23k. Intervening during these later stages (windows 10k–12.5k and 12.5k–15k) successfully accelerates this process, causing the induction capabilities to peak earlier than in the baseline. Conversely, upweighting the exact same patterns during intermediate phases (2.5k–5k and 5k–7.5k) results in induction being learned *later* than the control. Upweighting during steps 0-2500 appears to "stunt" the development of induction by causing induction to be learned to a weak degree very early on. Only much later does a second induction head start to form.

We note also that the heads that become induction heads (Olsson et al., 2022) differ only for the early window runs. Upweighting induction patterns later in training leads to layer 1 head 4 being the primary induction head, whereas upweighting earlier (windows 0 and 1) leads to various other heads specializing.

These results provide causal evidence for our stagewise framework. Upweighting a data pattern outside the phase where the model is receptive to it yields diminished returns or developmental delays, whereas upweighting it inside the critical window accelerates its acquisition.

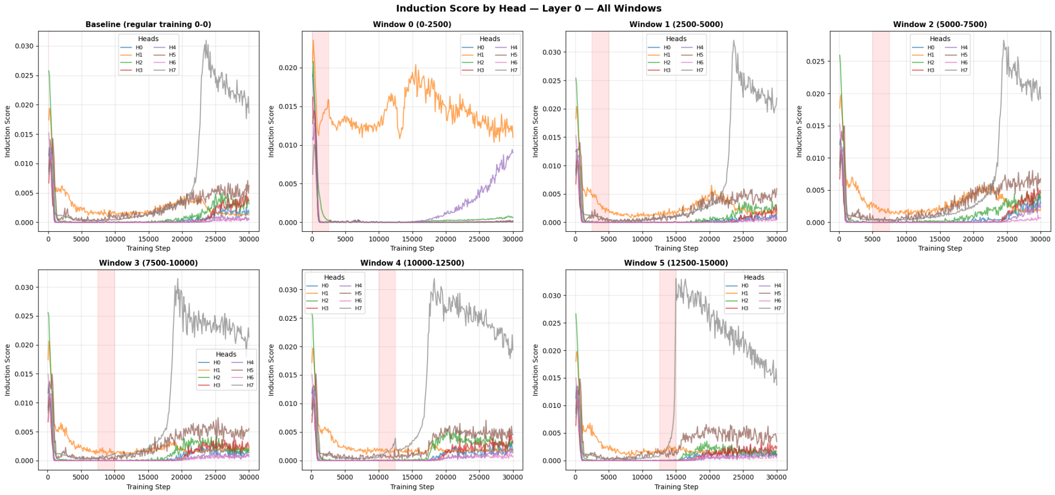


Figure 19: **Induction score trajectories for individual Layer 0 heads under different intervention schedules.** Each subplot represents a separate training run where induction-pattern tokens were upweighted during the indicated step interval (pink shaded region). The "Baseline" plot shows the natural emergence of induction heads (dominated by Head 7, grey line) occurring around step 23k. Interventions applied during the critical developmental windows identified by BIF (Window 4 and Window 5) successfully accelerate this formation, shifting the phase transition earlier to steps ∼17k and ∼15k respectively.