INPUT RELATIONAL STRUCTURE SHAPES TRAINING DYNAMICS IN NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Input representations are known to influence machine learning models, yet their effects are typically evaluated only after training has converged. We ask whether the training trajectory of learned representations carries a signature of the relational structure encoded in the inputs, and whether this signature predicts robustness. Using molecular property prediction, we compare models operating on graph-, sequence-, and vector-based encodings, and track the linear extractability of 32 chemical properties from their representations across training checkpoints. All three encodings support substantial extraction already at initialization, but differ in how they reorganize it: sequence models largely preserve their initial extractability profile (Spearman $\rho = 0.90$ between initial and peak extractability), whereas graph and vector models reorganize substantially ($\rho = 0.62$ and 0.68), each on different property subsets. To determine whether these representational dynamics are related to robustness, we introduce controlled corruption across all input relational structures and repeat training. Although corruption has little effect on overall task performance, it alters representations in an encoding-specific way: disruption is diffuse and strongest for vectors, intermediate for sequences, and selective for graphs, concentrated on properties that depend on the removed components. Finally, the evolution of extractability during uncorrupted training correlates with corruption sensitivity ($\rho = 0.72$), whereas geometric summaries of the same trajectories do not, suggesting representation dynamics as a complementary lens on robustness.

1 INTRODUCTION

Inductive biases enter the modeling pipeline at multiple levels (Goyal and Bengio, 2022), with the encoding of input data, in particular, determining the *relational structure* available to subsequent components of the pipeline (Battaglia et al., 2018). For example, graphs make entities and pairwise relations explicit, whereas sequences impose an ordering over elements, and fixed-length vectors provide no explicit relational structure. Consequently, models based on different input relational structures differ in which functions they efficiently approximate (Bronstein et al., 2021) and in their associated complexity (Ortiz-Jiménez et al., 2021). While these differences are typically characterized through properties of trained models, much less is known about how input relational structure influences the emergence of learned representations during optimization.

Training dynamics have been shown to leave lasting effects on the resulting model, with early training choices propagating to the converged solution (Jastrzębski et al., 2020; Frankle et al., 2020), and simple summaries of these dynamics already predicting generalization (Vahedian et al., 2021; Jiang et al., 2024; Schiff et al., 2021). These findings raise the question of whether input relational structure shapes not only final representations, but also how they emerge during training in machine learning models.

In this work, we investigate whether training dynamics carry a signature of input relational structure, focusing on how representations are formed and reorganized during training. In particular, we ask whether such signatures distinguish graph-, sequence-, and vector-based models, and whether they relate to model robustness.

We study the extent to which a set of properties can be extracted from model representations, using the probing paradigm (Alain and Bengio, 2017; Belinkov, 2022). We apply these probes across training checkpoints rather than only at convergence, so that changes in extractable information

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

throughout training become the object of study (Saphra and Lopez, 2019). Our experiments focus on molecular property prediction, where a single underlying entity (a molecule) admits three input relational structures: graphs (Gilmer et al., 2017), sequences (Weininger, 1988), and fixed-length vectors (Rogers and Hahn, 2010).

Molecules are associated with a rich set of interpretable and practically relevant properties, which we use in this work to probe learned representations. These properties range from simple compositional quantities, such as atom counts, to more global structural properties, such as spectral radii of atomic graphs. The domain additionally supports controlled interventions that remove specific structural information from the input, allowing us to systematically vary the information available to the model. Because different probe properties depend on the removed information to different degrees, this setting enables us to study how relational structure influences the propagation of perturbation effects through learned representations.

We make the following contributions:

- We track the linear extractability of molecular properties over training for graph, sequence, and vector inputs. All three support substantial extraction at initialization: sequence models change little thereafter, whereas graph and vector models reorganize substantially, each on a different set of properties.
- We show that a structured input corruption early in training leaves task performance statistically unchanged but alters representations in an encoding-specific way: localized to the affected properties for graphs, diffuse for vectors, and limited but non-specific for sequences.
- We introduce volatility, the standard deviation of extractability changes over training, and show that, computed on uncorrupted runs alone, it correlates with representational sensitivity to corruption and improves robustness-aware model selection.

2 Related Work

Input representation as inductive bias. A long line of work has established that the prior carried by the input relational structure constrains what downstream model architectures can exploit. Battaglia et al. (2018) characterize these constraints as relational inductive biases, distinguishing between the entity-and-relation structure of graphs, the ordering of sequences, and the unstructured all-to-all coupling of fixed-length vectors, arguing that such biases are central to the ability to extrapolate from known building blocks to novel configurations. Bronstein et al. (2021) formalize this through the symmetry group of the input domain, arguing that a model's class of efficiently expressible functions is determined by the invariances and equivariances its input structure admits. Goyal and Bengio (2022) situate these priors within a broader hierarchy of biases that also enter through architecture and objective. Subsequent empirical work has tied these priors to sample efficiency and out-of-distribution generalization (Webb et al., 2024; Kerg et al., 2022), and to a separation between in-distribution accuracy and the geometry of the representations that support it (Davari et al., 2022; Zhang et al., 2022). Together, these studies establish that input relational structure influences both what models learn and how efficiently they learn it, but they provide limited insight into how representations emerge over the course of optimization. We build on this foundation by asking whether the input relational structure also governs the *temporal dynamics* of representation formation, and how these dynamics shape the disruption of representations under input perturbation.

Probing across training. Probing (Alain and Bengio, 2017) operationalizes the functional content of representations by training auxiliary predictors to recover information. Subsequent work has proposed variations including control tasks, MDL-based probes, and structural probes (Hewitt and Liang, 2019; Voita and Titov, 2020; Hewitt and Manning, 2019), with recent syntheses emphasizing that many of these formulations capture closely related notions of extractable information (Belinkov, 2022). Probing across training checkpoints has been studied primarily in language models, where it tracks when different types of information become linearly accessible (Saphra and Lopez, 2019; Chiang et al., 2020; Müller-Eberstein et al., 2023). In the molecular domain, probing has been applied only to converged representations (Akhondzadeh et al., 2023). We extend probing by tracking how representations induced by different input relational structures evolve during training, comparing their trajectories across input formats rather than analyzing each in isolation, and assessing the robustness of the information they encode.

Compressed training dynamics as a predictor. Several lines of work compress the training trajectory into low-dimensional summaries that predict downstream outcomes. For example, Jiang et al. (2024) derive scalar capacitance metrics from edge dynamics on a network's line graph that predict task generalization, Vahedian et al. (2021) predict task performance from graph-property summaries of the first few epochs, and Schiff et al. (2021) use perturbation-response curves to anticipate generalization gaps. Moreover, Birdal et al. (2021) take a trajectory-topological route, predicting generalization from the persistent homology of the optimization path. Motivated by such developments, we also summarize the trajectory shape into a scalar, but target representation-level *robustness* rather than task generalization.

Geometric similarity vs. representational content. A broad family of tools, including CKA (Kornblith et al., 2019) and measures based on effective rank (Roy and Vetterli, 2007), compare representations through their geometric similarity. A recent line of work asks how such geometric comparisons relate to the information that can be extracted from representations; Ding et al. (2021) show that representational geometry and probe performance can diverge, as representations can exhibit high CKA similarity despite large differences in probing performance. Similarly, Davari et al. (2023) demonstrate transformations that move CKA significantly while leaving model behavior essentially intact, and Nguyen et al. (2022) show that the structure CKA recovers can be driven by a small subset of influential datapoints reflecting simple input statistics. In §5.3 we contrast content-based and geometric summaries of the same trajectories as predictors of robustness.

3 METHODS

Input representations. To isolate the contribution of input relational structure to representation learning, we hold data and tasks fixed, and vary only how each molecule is presented to the model. We use three relational structures: a **graph**, with atoms as nodes and bonds as edges, a **sequence**, presented as a SMILES string (Weininger, 1988), which is a textual linearization of the molecular graph, and as a **fixed-length vector**, namely an extended connectivity fingerprint (Rogers and Hahn, 2010) (ECFP), encoding local atomic neighborhoods up to a chosen radius.

Featurizations. To ensure that the effects we observe are attributable to relational structure rather than to specific modeling choices, within each input type we consider three different featurizations that vary the amount of local context provided to the model. For **graphs (node features)**: atom type; atom type combined with features derived from the atom's immediate neighborhood (1-hop); and atom type combined with features derived from larger multi-hop neighborhoods. For **sequences**: atom-level tokenization; Atom-in-SMILES (Ucak et al., 2023), which augments each atom token with a description of its local environment; and SMILES Pair Encoding (Li and Fourches, 2021), which merges frequent substrings into single tokens. For **fixed-length vectors**: ECFP radius in $\{0, 1, 2\}$. Aggregating over featurizations within each input format ensures that observed effects can be attributed to the input format rather than a specific featurization choice. Exact specifications are given in Appendix D.

Probing. We probe for 32 molecular properties (Appendix G) at epochs in $\{0, 10, 20, \ldots, 200\}$, where epoch 0 corresponds to the randomly initialized model prior to any training. At each epoch, we extract the activations of the penultimate layer over the training set, fit a linear probe per property, and evaluate it on the validation set. To distinguish representation encoding from probe capacity, we follow (Hewitt and Liang, 2019) and report *selectivity*, i.e., the gap between probe r^2 on true labels and on a control task with shuffled labels. Overall, the selected probing properties can be organized into five high-level categories: *compositional* properties are additive over atoms; *substructure counts* record occurrences of local patterns; *basic topological* properties capture simple connectivity statistics; *physicochemical* properties depend jointly on composition and connectivity; and *higher-order topological* properties are global invariants derived from spectral or distance-matrix structure. The full list of properties and the precise definition of each category are given in Appendix G.

Input corruption. To assess whether input relational structure influences representational robustness, we apply a *controlled input corruption* for the first N training epochs and restore full inputs for the remainder. Specifically, we use the Bemis–Murcko decomposition (Bemis and Murcko, 1996), which partitions each molecule into a coarse structural core and the finer components attached to

it. Models are then trained using only the structural core as input for the first $N \in \{5, 20, 60, 120\}$ epochs, and from epoch $N + 1$ onward, training proceeds on the full, unmodified molecules. This induces a single switch from a restricted to a full-information regime during training, where corrupted and uncorrupted runs receive identical inputs from epoch $N + 1$. The resulting setup enables a controlled comparison of representation dynamics *after the switch*, since any subsequent differences reflect the influence of the earlier restriction on how representations were formed. Moreover, we partition probed properties into *directly affected* properties, whose computation depends primarily on components removed during the restricted phase, and *indirectly affected* properties, whose computation relies jointly on removed components and information retained during the restricted phase. This allows us to assess whether post-switch dynamics exhibit selective disruption across property classes according to how much they rely on information removed during the restricted phase. The partition is provided in Appendix G.2. Moreover, a second corruption, which preserves molecular topology while removing atom and bond identity, is used as a replication and is described in Appendix H.

Trajectory comparisons. For each corrupted training run, we measure how its probe trajectory deviates from that of the corresponding uncorrupted run, obtained by training the same model on uncorrupted data, using the dynamic time warping (DTW) divergence (Sakoe and Chiba, 1978) between the two r^2 probe trajectories. DTW is invariant to local temporal misalignments, so trajectories with similar shapes but shifted along the time axis yield small distances, whereas large distances reflect differences in the structure of representation learning rather than its timing.

4 Experiments

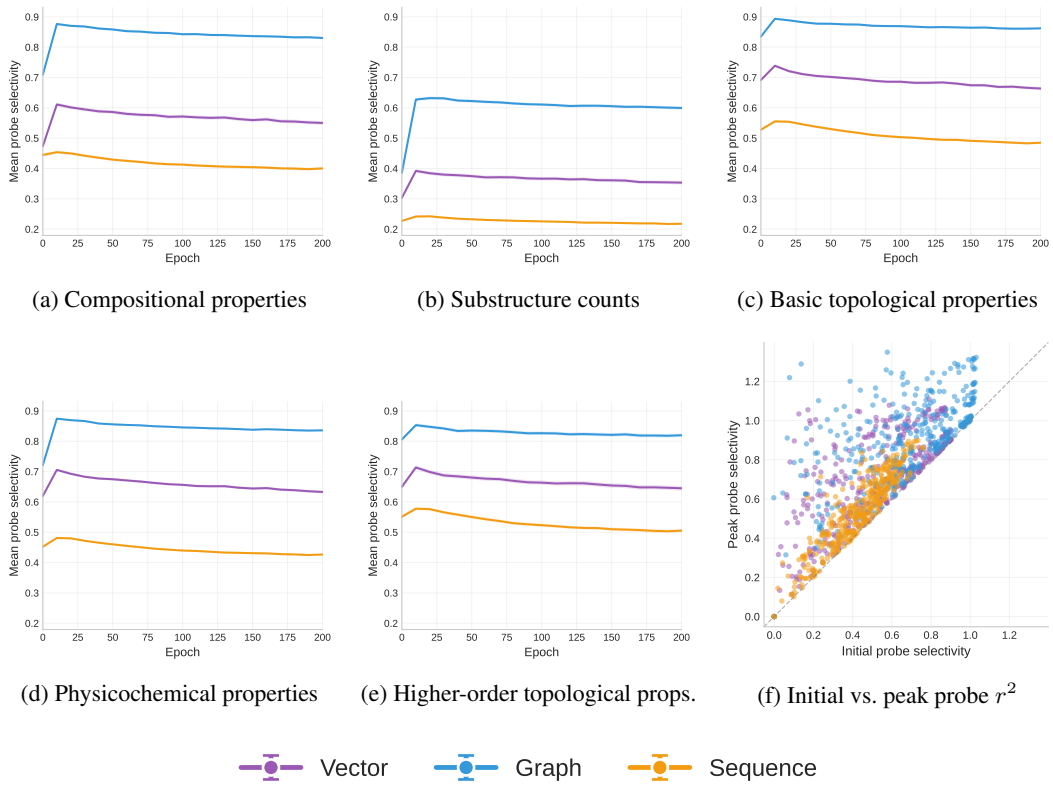
Architectures. To distinguish effects attributable to input relational structure from those arising from a particular model class, we evaluate multiple architectures for both graph- and sequence-based inputs. For graph inputs, we train a GIN (Xu et al., 2019) and a Graph Transformer (Dwivedi and Bresson, 2020), which differ in how relational information is propagated and aggregated across the molecular graph. For sequences, we use both a Transformer encoder (Vaswani et al., 2017) and a bidirectional LSTM (Schuster and Paliwal, 1997), providing two contrasting mechanisms for modeling token order and long-range dependencies. For fixed-length vectors, we employ an MLP, as this encoding does not define an explicit relational structure over input elements, making the fully connected MLP the natural architecture for operating on this representation. All models are trained for 200 epochs. Full architectural specifications and hyperparameters are reported in Appendix C.

Datasets and tasks. We evaluate all models on 12 molecular property prediction datasets drawn from MoleculeNet (Wu et al., 2018) and Therapeutics Data Commons (Huang et al., 2021). These comprise eight binary classification tasks (BBBP, BACE, HIV, AR, ER, PPAR, CYP2C19, CYP3A4) and four regression tasks (CACO2, ESOL, FreeSolv, Lipophilicity). All datasets are partitioned into random 80/10/10 training, validation, and test splits. Classification is evaluated with the Matthews Correlation Coefficient (Matthews, 1975), which is robust to class imbalance; regression is evaluated with mean squared error (MSE). Performance for all model-dataset pairs are reported in Appendix A.

Experimental scale. The study comprises 1260 baseline and 4320 corruption runs, each probed at 21 checkpoints on 32 properties with shuffled-label controls, resulting in approximately 7.50×10^6 probe fits (Appendix C).

5 Results

We organize our findings around three claims. First, the input format produces representational dynamics with distinct temporal signatures: sequence-based representations stabilize early and change little thereafter, while graph- and vector-based ones reorganize substantially, each on a different set of properties (§5.1). Second, an input corruption leaves task performance intact yet induces representation-level disruption whose *structure*—selective versus diffuse across properties—is determined by the input format (§5.2). Third, a single dynamic summary of uncorrupted training, i.e., the volatility of probe extractability, predicts representational vulnerability to corruption and substantially outperforms geometric summaries of the same representations (§5.3). Reported r^2 values are *selectivity* (Hewitt and Liang, 2019) scores throughout.



(a) Compositional properties (b) Substructure counts (c) Basic topological properties

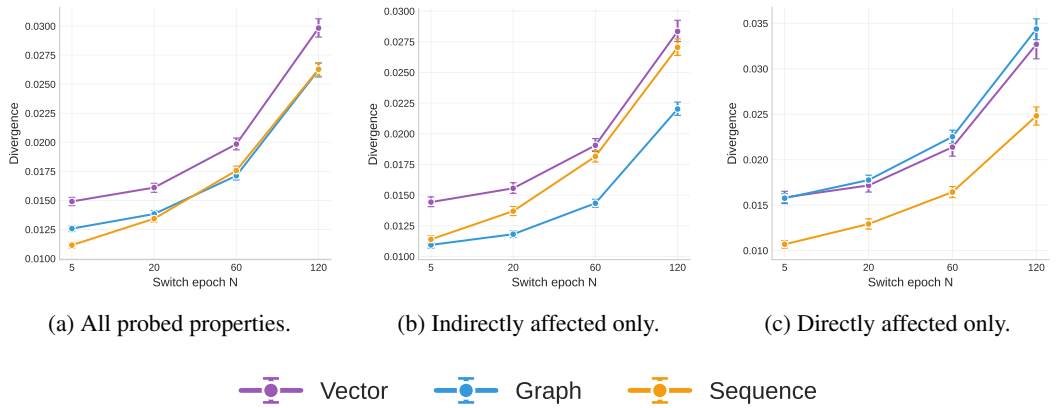
(d) Physicochemical properties (e) Higher-order topological props. (f) Initial vs. peak probe r^2

Figure 1: Probe r^2 trajectories by property category, across the three representation families (Vector, Graph, Sequence). Shaded bands: ± 1 SEM across runs. a compositional; b substructure; c basic topological; d physicochemical; e higher order topological. f initial and peak extractability.

5.1 Input relational structure produces distinct training dynamics

We begin by examining the mean probe r^2 trajectories across property categories, averaged within each input family across architectures, featurizations, datasets, and seeds (Figure 1). All three encodings exhibited a two-phase shape, with a sharp rise over the first 10 epochs, followed by a slow decay over the remaining 190, consistent with prior reports of early- and late-phase training regimes (Achille et al., 2017; Frankle et al., 2020). At initialization, mean probe selectivity ranged from 0.39 to 0.84 across categories for graph-based representations, from 0.30 to 0.69 for vector-based representations, and from 0.23 to 0.55 for sequence-based representations, with graphs highest and sequences lowest in every high-level property category. Graph-based representations gained the most in mean probe selectivity (ranging from 0.05 to 0.24 across categories), vector-based ones sat in between (0.05 to 0.14), and sequence-based representations gained the least (0.01 to 0.03). The subsequent decay in mean probe selectivity was small for all three families (0.02 to 0.08 compared to the respective peak values), indicating that the majority of representational reorganization occurs early in training. Thus, each family traversed a different magnitude of representational change while sharing the same coarse temporal shape.

To assess whether training generates information beyond what is present at initialization, we compare the property-level probe selectivity at initialization with the maximum value attained during training, across properties and datasets (Figure 1F). The three input families exhibited distinct relationships between these quantities, as reflected in their within-family Spearman correlations. Sequence-based representations showed a strong alignment ($\rho = 0.90$, $p < 10^{-10}$), indicating that the properties most extractable at initialization remain the most extractable ones after training. Graph- and vector-based representations were coupled to initialization considerably less strongly ($\rho = 0.62$ and $\rho = 0.68$, both $p < 10^{-10}$), indicating that training reshapes the relative ordering of property extractability rather than preserving the initial ordering of extractability across properties.



(a) All probed properties. (b) Indirectly affected only. (c) Directly affected only.

Figure 2: Length-normalized post-switch DTW divergence as a function of switch epoch N, broken down by representation family. Error bars: 95% bootstrap CIs over per-step DTW values. Panel a aggregates all properties; b restricts to indirectly affected properties; c to directly affected ones.

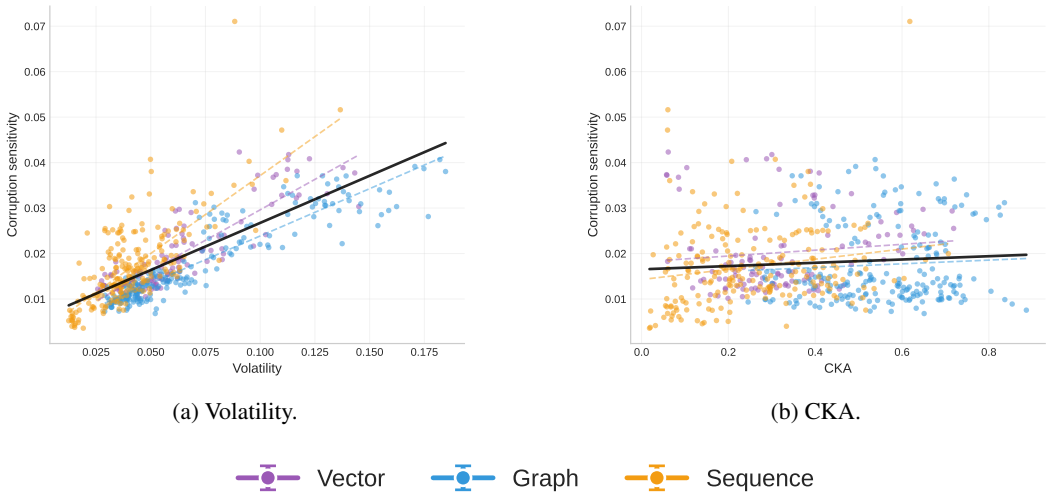
Moreover, the observed partial coupling was not uniform across property categories, and the pattern differed between the two families. For graph-based representations, the correlation was high for basic and higher-order topological properties ($\rho = 0.79$ and 0.92, respectively) and low for compositional and physicochemical properties ($\rho = 0.35$ and 0.37). For vector-based representations the correlation was highest for substructure counts ($\rho = 0.70$) and higher-order topology ($\rho = 0.68$), and lowest for physicochemical properties ($\rho = 0.41$). Sequence-based representations showed uniformly high correlations across all five categories ($\rho = 0.75$ to 0.91). In each case, the properties that remained coupled to initialization were those the input format encodes directly: the graph encoding specifies the full adjacency structure, from which topological invariants follow, whereas an ECFP records counts of local atomic neighborhoods, from which substructure counts follow. Properties not directly available in either format, such as the compositional and physicochemical ones, are acquired during training instead. This suggests that training primarily reorganizes information that is not already explicitly supported by the relational structure of the input. Sequences remained uniformly coupled across all five categories, so this account applied to the two families that reorganized substantially, while the sequence case was governed by the overall stability of their representations.

5.2 Input corruption reveals input-specific representational divergence

The results in §5.1 establish that different input relational structures induce different representational trajectories during training. We next ask whether the input relational structure actively shapes how representations adapt to perturbations during training. To this end, we corrupt inputs using the Bemis–Murcko decomposition described in §3, henceforth referred to as the Murcko corruption: each corrupted run sees only the structural core of every molecule for the first $N \in \{5, 20, 60, 120\}$ epochs, after which training resumes on the full molecule. Details can be found in Appendix G.1.

Task-level performance remains unchanged. Aggregating across architectures and switch epochs, corrupted runs perform 0.026 baseline standard deviations below their matched counterparts, with a 95% confidence interval of $[-0.066, +0.015]$ (Cohen's d $= -0.04$; details in Appendix E). Moreover, there is no dose-response relationship between corruption duration and performance deficit (Spearman $|\rho| \leq 0.05$, all $p > 0.26$). Taken together, these results indicate that corrupted runs are indistinguishable from their baseline counterparts under standard evaluation, and that longer corruption does not systematically increase or decrease the effect at the task-performance level.

Representational content diverges. While task-level performance differences are negligible, analyzing the post-switch phase reveals whether internal representations return to their unperturbed trajectories once inputs are restored or remain systematically altered in terms of their encoded content. We quantify post-switch divergence using the length-normalized DTW divergence between probe selectivity trajectories over epochs $N+1$ to 200, to ensure comparability across different values of N.



(a) Volatility.

(b) CKA.

Figure 3: Corruption sensitivity as a function of a measure calculated during uncorrupted training. Volatility is a predictive measure (Panel a), while CKA is not (Panel b).

Post-switch divergence increases monotonically with the duration of the initial corruption across all input families (Figure 2). Thus, even though all models subsequently received identical inputs, longer periods of restricted information left increasingly persistent traces in the learned representations. Moreover, in all cases, the divergence at $N = 120$ was at least double the value $N = 5$: from 0.015 to 0.03 for vector-based representations, 0.013 to 0.026 for graph-based representations, and 0.011 to 0.026 for sequence-based representations. These results indicate that divergence scales with corruption duration.

Structure of divergence depends on property class. When stratified by property class (Fig. 2b, 2c), the three input families exhibited distinct patterns of post-switch divergence, which we interpret in terms of their initialization–endpoint coupling in §5.1. Fixed-length vectors exhibited similar divergence across property classes (maximum difference 0.033 vs. 0.028 at $N = 120$, respectively). The reorganization they undergo during training (§5.1, $\rho = 0.68$) is therefore largely non-selective: in the absence of relational structure, perturbations propagated broadly across the representation space, resulting in uniform divergence across property classes. Sequence-based representations showed no separation between property classes, with indirectly affected properties diverging marginally more than directly affected ones (0.027 vs. 0.025 at $N = 120$). In combination with the strong initialization–endpoint coupling reported in §5.1 ($\rho = 0.90$), this indicates that representations remain tightly constrained by their initial structure, limiting the extent of post-switch deviation despite the apparently global propagation of perturbations. Graph-based representations, in contrast, exhibited clear separation between the directly and indirectly affected property classes at all N, with maximum differences from 0.005 at $N = 5$ to 0.012 at $N = 120$. Interestingly, although graph- and vector-based representations are coupled to their initialization to a comparable degree ($\rho = 0.62$ and $\rho = 0.68$, §5.1) and reach comparable overall divergence, corruption induces selective propagation of effects for graphs, in contrast to the diffuse propagation seen in vectors. Thus, the magnitude of representational reorganization alone does not determine robustness; the structure of the underlying relation also matters. This pattern replicates under a second corruption (Appendix H).

5.3 Trajectory volatility predicts corruption sensitivity

Sections 5.1 and 5.2 show that input relational structure induces distinct representational content dynamics and differential responses to perturbation. We next ask whether response to corruption can be anticipated from baseline, uncorrupted, training trajectories alone. To this end, we summarize each baseline training trajectory using a single scalar quantity, *volatility*, defined as the standard deviation of the differences in probe selectivity between consecutive checkpoints, computed per property and averaged over properties. Volatility thus captures the extent of representational change

during training, with higher values corresponding to more pronounced reorganization over time. Appendix B provides an ablation over alternative ways of summarizing trajectory fluctuation, all of which yield consistent rankings.

Volatility correlates with corruption sensitivity. We start by comparing the volatility of each baseline run against its mean corruption sensitivity, defined as the average divergence from its corresponding corrupted counterparts across all switch epochs $N \in \{5, 20, 60, 120\}$ (Figure 3). Across all runs, the correlation was $\rho = 0.72$, with the relationship remaining predictive within each input family ($\rho = 0.89$ for graphs, $\rho = 0.86$ for vectors, and $\rho = 0.67$ for sequences), suggesting that volatility carried a signal that is related to corruption sensitivity. Average volatility also separated sequence-based representations (0.042) from graph- and vector-based ones (0.070 and 0.065), the same split that appeared in the initialization–endpoint coupling of §5.1, so both measures agreed on which family reorganizes least. This relationship is robust, as cluster-bootstrap intervals over datasets, architectures, or featurizations all exclude zero ($[0.52, 0.82]$, $[0.64, 0.87]$, $[0.60, 0.85]$). Moreover, it is not driven by shared probe noise: replacing a run's volatility with the average volatility computed from runs with the same configuration, but a different seed, left the correlation nearly unchanged ($\rho = 0.71$), and it survived partialing out control-probe volatility ($\rho = 0.48$; Appendix I). The results hold under the corruption in Appendix H.

Volatility-based models selection. We found volatility only weakly correlated with validation performance (median within-dataset $\rho = 0.18$). In contrast, validation performance was uncorrelated with corruption sensitivity (median $\rho = 0.02$). Selecting, within each dataset, the run with the lowest volatility reduced corruption sensitivity by 51% relative to random selection, with lower sensitivity in all twelve datasets ($p < 10^{-3}$). Selecting by validation performance yielded no reduction ($+0.1\%$; lower in four of twelve). In all twelve datasets, however, the selected run was a BiLSTM, the architecture with the lowest average volatility; within individual architectures, the reduction ranged from 5% to 41%. Volatility-based selection thus reduced corruption sensitivity largely through the choice of architecture. We observed similar results under the second corruption in Appendix H).

Geometric trajectories do not predict corruption sensitivity. We compare volatility with geometric measures computed along the same training trajectory, including CKA, nearest-neighbor preservation, and effective rank between consecutive checkpoints, each summarized by the standard deviation of its first differences, exactly as for volatility. This contrasts the evolution of extractable content during training with the evolution of representation geometry over the same trajectory, providing an alternative way of summarizing training dynamics. None of the geometric summaries predicts corruption sensitivity, with pooled ρ between -0.04 and 0.05, and all of them change sign across input families, showing no consistent relationship with corruption sensitivity (Appendix F).

6 Discussion

We position the results of §5, taking up three threads: volatility as a summary measure (§6.1), its relation to geometric summaries (§6.2), and where this leaves volatility as a diagnostic (§6.3).

6.1 Volatility as a post-hoc summary of training dynamics

In §5.3, we observed that probe-trajectory volatility correlated with corruption sensitivity, both overall and within each input family. At the same time, volatility is a post-hoc training summary, so it is part of a broader line of work that compresses training dynamics into low-dimensional summaries (Vahedian et al., 2021; Jiang et al., 2024; Schiff et al., 2021). However, instead of predicting task-level outcomes, such as final accuracy, volatility relates training dynamics to the divergence of internal representations under controlled input perturbations. Section 5.2 shows that corrupted and baseline runs are indistinguishable in task-level performance while their internal representations diverge systematically with the duration of corruption, placing these dynamics outside the scope of summaries calibrated on task outcomes. Moreover, while prior work primarily focuses on early training, using partial trajectories to predict final performance, volatility summarizes the full training trajectory, characterizing the learned representations themselves, and may therefore capture structure that accumulates over the course of optimization. Taken together, these observations extend existing low-dimensional summaries of training dynamics to representation-level quantities.

6.2 CONTENT AND GEOMETRY SEPARATE

Section 5.3 shows that geometric summaries of representation similarity correlate negligibly with corruption sensitivity across a series of measures. That this decoupling holds across measures indicates that geometry does not capture differences in encoded information, consistent with prior work showing that geometric similarity and content can diverge (Ding et al., 2021). Geometric summaries also relate inconsistently to robustness across input formats, changing sign between families (Table 7), whereas volatility is positive in every family. At the same time, content-level reorganization does not simply mirror the amount of relational structure available in the input. Following Battaglia et al. (2018), graphs, sequences, and fixed-length vectors can be ordered by how much explicit relational structure they encode, yet the coupling between initial and peak extractability reported in Section 5.1 does not follow this ordering. Vector- and graph-based representations, at opposite ends of the hierarchy, are coupled to their initialization to a comparable degree ($\rho = 0.68$ and 0.62), while sequence-based representations, intermediate in the hierarchy, show the strongest coupling ($\rho = 0.90$). Content reorganization is therefore not a monotone function of relational strength. A plausible reason for this dissociation is that content and geometry are sensitive to different aspects of a representation: geometric measures can be largely determined by its high-variance directions (Ding et al., 2021), whereas a probe only requires a property to be aligned with some direction, regardless of the overall variance that direction carries. Training can therefore rotate or rescale directions, changing what is extractable, while leaving the global geometry unchanged, and conversely alter the geometry substantially without affecting extractable content (Davari et al., 2023).

6.3 VOLATILITY AS A DIAGNOSTIC

In §5.2, we showed that models indistinguishable under standard performance metrics may have divergent representations, and in §5.3 that volatility correlated with the magnitude of this divergence when measured against the same-seed baseline. More broadly, models with similar in-distribution performance can behave very differently under distribution shift (D'Amour et al., 2022), and task performance alone is insufficient for selecting among such models (Ortiz-Jiménez et al., 2021). The standard response is to augment evaluation with explicit stress tests or OOD benchmarks prior to deployment. While effective, such approaches require additional evaluation data that may not be available at model selection time, and depend on assumptions about the relevant shift. Summaries of training dynamics such as volatility could offer a complementary signal, as they are computed from each candidate's own training trajectory and require no auxiliary evaluation. The selection experiment in §5.3 points in this direction: among candidate runs that standard evaluation could not separate, choosing by volatility yielded runs with lower post-switch divergence, whereas choosing by validation performance did not, although this reduction was largely driven by the choice of architecture. This raises a broader question: do models that reorganize more during learning also differ systematically in robustness once trained? Volatility may offer a starting point for this question, and for exploring whether training dynamics can inform model selection beyond predictive performance.

7 CONCLUSIONS

We studied how training trajectories reflect the relational structure of model inputs. Tracking probe extractability over training and perturbing the structural information available early on, we found three things. Input formats induce distinct representational paths. Corruption invisible at the task level leaves format-specific traces in representations. And the volatility of extractable content during standard training predicts these traces where endpoint geometry and task performance do not. The dynamics of representation learning thus carry robustness-relevant information that is invisible at convergence. Extending such training-based summaries to other modalities and training settings may provide a basis for model selection grounded in representational stability rather than predictive performance alone.

AI USE STATEMENT

(This section is **required** and does not count toward the page limit.)

In this work, we used generative AI tools for the following tasks with required disclosure: generating parts of the code used to implement our experiments, and loading, cleaning, and reformatting the

experimental results prior to analysis. We have not used generative AI tools to generate synthetic data, develop theoretical models or conceptual frameworks, propose or refine hypotheses, design or provide feedback on the research methodology or experiments, or interpret results, and the formulation of mathematical claims and proofs, translation, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for tasks with recommended disclosure, namely to draft initial versions of parts of the paper text and to subsequently edit the paper for readability and length. We have reviewed all AI-assisted work: all AI-generated code was reviewed by the authors and tested for correctness by comparing its outputs against independent computations on small cases, all AI-assisted text was revised by the authors, and every claim, number, and result reported in the paper was produced by the authors' experiments and verified by the authors against the experimental outputs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 478(2266): 20210068, 2022. doi: 10.1098/rspa.2021.0068.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Guillermo Ortiz-Jiménez, Apostolos Modas, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. A neural anisotropic view of underspecification in deep learning. *arXiv preprint arXiv:2104.14372*, 2021.

Stanisław Jastrzębski, Maciej Szymczak, Stanislav Fort, Devansh Arpit, Jacek Tabor, Kyunghyun Cho, and Krzysztof Geras. The break-even point on optimization trajectories of deep neural networks. In *8th International Conference on Learning Representations (ICLR)*, 2020.

Jonathan Frankle, David J. Schwab, and Ari S. Morcos. The early phase of neural network training. In *8th International Conference on Learning Representations (ICLR)*, 2020.

Fatemeh Vahedian, Ruiyu Li, Puja Trivedi, Di Jin, and Danai Koutra. Leveraging the graph structure of neural network training dynamics. *arXiv preprint arXiv:2111.05410*, 2021.

Chunheng Jiang, Zhenhan Huang, Tejaswini Pedapati, Pin-Yu Chen, Yizhou Sun, and Jianxi Gao. Network properties determine neural network performance. *Nature Communications*, 15(1):5718, 2024. doi: 10.1038/s41467-024-48069-8.

Yair Schiff, Brian Quanz, Payel Das, and Pin-Yu Chen. Predicting deep neural network generalization with perturbation response curves. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 21176–21188, 2021.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. In *Workshop Track, 5th International Conference on Learning Representations (ICLR)*, 2017. arXiv:1610.01644.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022. doi: 10.1162/coli_a_00422.

Naomi Saphra and Adam Lopez. Understanding learning dynamics of language models with SVCCA. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3257–3267, 2019. doi: 10.18653/v1/N19-1329.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272, 2017.

David Weininger. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005.

David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, 2010. doi: 10.1021/ci100050t.

Taylor W. Webb, Steven M. Frankland, Awni Altabaa, Simon Segert, Kamesh Krishnamurthy, Declan Campbell, Jacob Russin, Tyler Giallanza, Zack Dulberg, Randall O'Reilly, John Lafferty, and Jonathan D. Cohen. The relational bottleneck as an inductive bias for efficient abstraction. *Trends in Cognitive Sciences*, 28(9):829–843, 2024. doi: 10.1016/j.tics.2024.04.001.

Giancarlo Kerg, Sarthak Mittal, David Rolnick, Yoshua Bengio, Blake Richards, and Guillaume Lajoie. On neural architecture inductive biases for relational tasks. *arXiv preprint arXiv:2206.05056*, 2022.

MohammadReza Davari, Nader Asadi, Sudhir Mudur, Rahaf Aljundi, and Eugene Belilovsky. Probing representation forgetting in supervised and unsupervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16712–16721, 2022.

Jianyu Zhang, David Lopez-Paz, and Léon Bottou. Rich feature construction for the optimization-generalization dilemma. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pages 26397–26411, 2022.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, 2019. doi: 10.18653/v1/D19-1275.

Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 183–196, 2020. doi: 10.18653/v1/2020.emnlp-main.14.

John Hewitt and Christopher D. Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 4129–4138, 2019. doi: 10.18653/v1/N19-1419.

Cheng-Han Chiang, Sung-Feng Huang, and Hung-yi Lee. Pretrained language model embryology: The birth of ALBERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6813–6828, 2020. doi: 10.18653/v1/2020.emnlp-main.553.

Max Müller-Eberstein, Rob van der Goot, Barbara Plank, and Ivan Titov. Subspace chronicles: How linguistic information emerges, shifts and interacts during language model training. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13190–13208, 2023.

Mohammad Sadegh Akhondzadeh, Vijay Lingam, and Aleksandar Bojchevski. Probing graph representations. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 206 of *Proceedings of Machine Learning Research*, pages 11630–11649, 2023.

Tolga Birdal, Aaron Lou, Leonidas J. Guibas, and Umut Şimşekli. Intrinsic dimension, persistent homology and generalization in neural networks. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 6776–6789, 2021.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529, 2019.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *Proceedings of the 15th European Signal Processing Conference (EUSIPCO)*, pages 606–610, 2007.

Frances Ding, Jean-Stanislas Denain, and Jacob Steinhardt. Grounding representation similarity through statistical testing. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 1556–1568, 2021.

MohammadReza Davari, Stefan Horoi, Amine Natik, Guillaume Lajoie, Guy Wolf, and Eugene Belilovsky. Reliability of CKA as a similarity measure in deep learning. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Thao Nguyen, Maithra Raghu, and Simon Kornblith. On the origins of the block structure phenomenon in neural network representations. *Transactions on Machine Learning Research*, 2022. arXiv:2202.07184.

Umit V. Ucak, Islambek Ashyrmamatov, and Juyong Lee. Improving the quality of chemical language model outcomes with atom-in-SMILES tokenization. *Journal of Cheminformatics*, 15(1):55, 2023. doi: 10.1186/s13321-023-00725-9.

Xinhao Li and Denis Fourches. SMILES pair encoding: A data-driven substructure tokenization algorithm for deep learning. *Journal of Chemical Information and Modeling*, 61(4):1560–1569, 2021. doi: 10.1021/acs.jcim.0c01127.

Guy W. Bemis and Mark A. Murcko. The properties of known drugs. 1. molecular frameworks. *Journal of Medicinal Chemistry*, 39(15):2887–2893, 1996. doi: 10.1021/jm9602928.

Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978. doi: 10.1109/TASSP.1978.1163055.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations (ICLR)*, 2019.

Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020. Presented at the AAAI 2021 Workshop on Deep Learning on Graphs (DLG-AAAI'21).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 5998–6008, 2017.

Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. doi: 10.1109/78.650093.

Zhenqin Wu, Bharath Ramsundar, Evan N. Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S. Pappu, Karl Leswing, and Vijay Pande. MoleculeNet: A benchmark for molecular machine learning. *Chemical Science*, 9(2):513–530, 2018. doi: 10.1039/C7SC02664A.

Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor W. Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. In *Proceedings of Neural Information Processing Systems, NeurIPS Datasets and Benchmarks Track*, 2021.

Brian W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, 1975. doi: 10.1016/0005-2795(75)90109-9.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep neural networks. *arXiv preprint arXiv:1711.08856*, 2017.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

{C. A.} Field and {A. H.} Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 69(3):369–390, June 2007. ISSN 1369-7412. doi: 10.1111/j.1467-9868.2007.00593.x.

A Colin Cameron and Douglas L Miller. A practitioner's guide to cluster-robust inference. *Journal of human resources*, 50(2):317–372, 2015.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006. URL http://jmlr.org/papers/v7/demsar06a.html.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300, 01 1995. ISSN 0035-9246. doi: 10.1111/j.2517-6161.1995.tb02031.x. URL https://doi.org/10.1111/j.2517-6161.1995.tb02031.x.

A MODEL PERFORMANCE

Table 1: Per-dataset baseline test metrics across architectures, mean ± std over seven seeds and three featurizations. Regression tasks (REG) report MSE (lower is better); classification tasks (CLF) report MCC (higher is better, range $[-1, 1]$). Best per row in **bold**. All configurations trained successfully, with no signs of collapse or non-convergence; the elevated standard deviation on FreeSolv reflects the dataset's large target range rather than training instability.

Dataset	Task	GIN	Graph Transformer	Sequence Transformer	BiLSTM	MLP
AR	Clf	$\mathbf{0.609 \pm 0.085}$	0.574 ± 0.105	0.593 ± 0.093	0.541 ± 0.097	0.603 ± 0.097
BACE	Clf	$\mathbf{0.598 \pm 0.037}$	0.593 ± 0.079	0.566 ± 0.060	0.564 ± 0.048	0.595 ± 0.061
BBBP	Clf	0.660 ± 0.055	0.647 ± 0.062	$\mathbf{0.685 \pm 0.050}$	0.653 ± 0.061	0.642 ± 0.052
CACO2	Reg	0.217 ± 0.055	0.219 ± 0.056	0.361 ± 0.072	0.312 ± 0.085	$\mathbf{0.190 \pm 0.023}$
CYP2C19	Clf	$\mathbf{0.645 \pm 0.027}$	0.628 ± 0.052	0.546 ± 0.043	0.548 ± 0.051	0.574 ± 0.047
CYP3A4	Clf	$\mathbf{0.211 \pm 0.141}$	0.195 ± 0.098	0.120 ± 0.087	0.145 ± 0.106	0.188 ± 0.125
ER	Clf	$\mathbf{0.314 \pm 0.059}$	$\mathbf{0.314 \pm 0.061}$	0.246 ± 0.063	0.235 ± 0.054	0.252 ± 0.048
ESOL	Reg	$\mathbf{0.490 \pm 0.107}$	0.537 ± 0.160	0.799 ± 0.312	0.641 ± 0.323	0.658 ± 0.156
FreeSolv	Reg	$\mathbf{1.437 \pm 0.649}$	2.138 ± 1.148	2.851 ± 1.722	2.447 ± 1.772	2.256 ± 0.765
HIV	Clf	$\mathbf{0.448 \pm 0.051}$	0.394 ± 0.079	0.348 ± 0.048	0.383 ± 0.047	0.420 ± 0.037
Lipo	Reg	$\mathbf{0.387 \pm 0.054}$	0.520 ± 0.175	0.843 ± 0.199	0.621 ± 0.194	0.546 ± 0.124
PPAR	Clf	$\mathbf{0.363 \pm 0.101}$	0.354 ± 0.136	0.268 ± 0.120	0.303 ± 0.101	0.329 ± 0.131

B ROBUSTNESS TO THE CHOICE OF VOLATILITY METRIC

The volatility reported in the main text is the standard deviation of the differences in probe selectivity between consecutive checkpoints, computed per property and averaged across probed properties. To verify that our conclusions do not depend on this specific aggregation, we recompute volatility under four alternative definitions and re-evaluate its Spearman correlation with corruption sensitivity. **All five definitions yield consistent rankings** (Table 2): pooled correlations lie in $[0.72,\ 0.78]$, and within-family correlations are similarly stable.

The four alternatives are: (i) the standard deviation of probe selectivity along the trajectory, $\mathrm{sd}_t(r_t^2)$; (ii) the length-normalized total variation $\frac{1}{T}\sum_t |\Delta r_t^2|$; (iii) the range $\max_t r_t^2 - \min_t r^2$; and (iv) the standard deviation of residuals after linear detrending of the trajectory. Definition (iv) is the most informative control: by removing the slow component of r^2 before measuring dispersion, it isolates fluctuation from drift. Its pooled correlation with corruption sensitivity is in fact the *highest* of the five ($\rho = 0.78$), ruling out the alternative explanation that volatility is merely a proxy for the magnitude of slow drift in the trajectory.

Within-family correlations follow the same pattern across all definitions: $\rho \in [0.69,\ 0.86]$ for vector, $[0.84,\ 0.94]$ for graph, and $[0.57,\ 0.70]$ for sequence. The systematically weaker correlation in the sequence family persists across every metric, including detrending, indicating a property of sequence trajectory dynamics rather than an artifact of the volatility definition.

Volatility definition	Pooled	Vector	Graph	Sequence
Std of r^2	0.750	0.690	0.876	0.572
Std of first differences *(main text)*	0.718	0.856	0.887	0.670
Total variation (length-normalized)	0.740	0.811	0.935	0.699
Range	0.733	0.689	0.837	0.566
Std of detrended residuals	**0.775**	0.847	0.872	0.657

Table 2: Spearman correlation ρ between alternative volatility metrics and corruption sensitivity, pooled and broken down by input family. Detrending yields the highest pooled correlation (**bold**), indicating that the volatility signal is not an artifact of slow drift in r^2.

C MODEL SPECIFICATION AND EXPERIMENTAL SCALE

Table 3: Architectural hyperparameters by model. All models use hidden dimension 64 and dropout 0.2. Training: Adam with learning rate 10^{-3}, weight decay 10^{-4}, 200 epochs, random $80/10/10$ split.

Model	Depth	Model-specific settings
GIN	3	readout: global add pool
GraphTransformer	3	heads $= 4$; readout: global add pool
Transformer	3	heads $= 4$; max length $= 256$; mean pooling
BiLSTM	3	hidden $= 32$ per direction (64 effective); mean pooling
MLP	3	input: 2048-bit count Morgan fingerprints

Experimental scale. The full study comprises a large set of training runs and probing experiments. For the baseline setting, each of the five model architectures is trained using three featurizations across the twelve datasets and seven random seeds, yielding $5 \times 3 \times 12 \times 7 = 1260$ independent training runs. For the corruption experiments, we train models using three (of the seven) seeds, four corruption durations ($N \in \{5, 20, 60, 120\}$), and two corruption types, resulting in $5 \times 3 \times 12 \times 3 \times 4 \times 2 = 4320$ additional training runs. Each trained model is probed at 21 checkpoints and evaluated on 32 properties. For each property, probes are fit both on the true target and on a shuffled-label control task to compute selectivity (Hewitt and Liang, 2019), yielding $21 \times 32 \times 2 = 1344$ probe fits per trained model. Across all baseline and corruption experiments, this corresponds to approximately $(1260 + 4320) \times 1344 \approx 7.50 \times 10^6$ individual probe fits.

D FEATURIZATION SPECIFICATIONS

To attribute observed effects to relational structure rather than to specific featurization choices, within each input family we vary the amount of local context exposed to the model across three settings. Below we specify each setting exactly.

D.1 GRAPH NODE FEATURIZATIONS

Each atom is encoded as a concatenation of one-hot or scalar features computed via RDKit.

baseline. Atom symbol only, one-hot encoded over {C, N, O, F, P, S, Cl, Br, I, Si, Se, B, H}.

1-hop. Atom symbol (one-hot over {C, N, O, F, P, S, Cl, Br, I, Si, Se, B, H}), augmented with three local descriptors:

- Number of heavy-atom neighbors, one-hot over $\{0, 1, 2, 3, 4\}$.
- Total number of attached hydrogens, one-hot over $\{0, 1, 2, 3, 4\}$.
- Hybridization state, one-hot over {S, SP, SP2, SP2D, SP3, SP3D, SP3D2}.

extended. The `1-hop` feature set augmented with four additional descriptors capturing structural context beyond the immediate neighborhood:

- Ring membership (binary).
- Aromaticity (binary).
- Atomic Randić contribution (scalar; per-atom contribution to the Randić connectivity index, which sums $1/\sqrt{d_i d_j}$ over bonded pairs (i, j) with degrees d_i, d_j).
- Atomic Wiener contribution (scalar; per-atom contribution to the Wiener index, the sum of shortest-path distances between this atom and all other atoms in the molecule).

Bond featurization (shared across all three settings). Each bond is encoded by its bond order, one-hot over $\{1, 1.5, 2, 3\}$ (single, aromatic, double, triple).

D.2 SMILES TOKENIZATIONS

Each SMILES string is tokenized using one of three schemes:

baseline. Atom-level tokenization, in which each token corresponds to a single atom symbol or syntactic character of the SMILES string (Weininger, 1988).

ais. Atom-in-SMILES tokenization (Ucak et al., 2023), in which each atom token is replaced by a composite token that encodes the atom together with a description of its immediate chemical environment, distinguishing the same atom in different local contexts.

smilespe. SMILES Pair Encoding (Li and Fourches, 2021), a byte-pair-encoding-style scheme in which frequent substrings are iteratively merged into single tokens, producing a vocabulary of multi-character chemical motifs.

For all three schemes, sequences are padded or truncated to a maximum length of 256 tokens.

D.3 ECFP RADII

Each molecule is encoded as an Extended-Connectivity Fingerprint (Rogers and Hahn, 2010) of length 2048, computed in count mode (each bit position records the count of occurrences of the corresponding substructure rather than its presence or absence). The three settings vary the radius of the substructures considered:

- **radius 0.** Each substructure corresponds to a single atom invariant.
- **radius 1.** Each substructure corresponds to an atom and its directly-bonded neighbors.
- **radius 2.** Each substructure corresponds to an atom and its 2-hop neighborhood.

E TASK PERFORMANCE UNDER INPUT CORRUPTION

Standardization and inference. Task metrics are not directly comparable across datasets, since regression is evaluated with mean squared error in the units of each target and classification with the Matthews correlation coefficient. We therefore standardize each paired difference between a corrupted run and its matched baseline by the between-run standard deviation of baseline performance on the same dataset, so that all differences are expressed in baseline run-to-run standard deviations, signed so that positive values indicate the corrupted run outperforms its baseline. Because paired differences are correlated within a dataset, sharing seeds and switch epochs, confidence intervals are computed with a cluster bootstrap over datasets (Field and Welsh, 2007; Colin Cameron and Miller, 2015), following the convention of treating the dataset as the unit of generalization in empirical model comparison (Demšar, 2006). Where multiple cells are tested, p-values are adjusted with Benjamini–Hochberg (Benjamini and Hochberg, 1995).

Per-cell breakdown of task-level performance. The pooled paired difference reported in the main text aggregates over four switch epochs $N \in \{5, 20, 60, 120\}$, five architectures, two task types, and three input families. To verify that this null result does not conceal a dose–response or directional effect within a particular cell, we report the standardized paired difference stratified along each axis. For every cell we report n, the mean difference with a 95% cluster-bootstrap confidence interval over datasets, and Cohen's d. Tables 4, 5, and 6 report the breakdown per N, per (input family, N), and per task type respectively.

The picture is consistent across stratifications. On the standardized scale the classification and regression splits agree in sign and magnitude ($d = -0.03$ and $d = -0.07$), so the pooled null is not produced by cancellation across task types with incomparable units. Across the twelve cells of the family-by-N table, none is significant (smallest adjusted $q = 0.084$), and no cell exceeds 0.13 baseline standard deviations in magnitude. Graph-based models are the only family with a consistent sign, remaining slightly below their baselines at every N (mean -0.098 standard deviations), but the effect does not scale with the duration of corruption ($\rho = -0.02$, $p = 0.62$). We conclude that the

absence of a task-level effect of corruption is robust across switch epochs, input families, and task types.

N	n	Mean Δ	95% CI	Cohen's d	p
5	540	–0.025	$[-0.085, +0.047]$	–0.039	0.423
20	537	–0.012	$[-0.055, +0.034]$	–0.018	0.604
60	540	–0.010	$[-0.061, +0.041]$	–0.014	0.720
120	540	–0.057	$[-0.134, +0.011]$	–0.084	0.096
Pooled	2153	–0.026	$[-0.066, +0.015]$	–0.038	0.223

Table 4: Paired difference Δ between corrupted and baseline runs, stratified by switch epoch N, in units of the between-run standard deviation of baseline performance on the same dataset. Sign-corrected so that positive values indicate the corrupted run outperforms its matched baseline. Confidence intervals and p-values come from a bootstrap that resamples datasets rather than individual runs. All CIs include zero.

Input family	N	n	Mean Δ	95% CI	Cohen's d	q
Fingerprint	5	108	0.064	$[-0.029, +0.158]$	0.101	0.310
Fingerprint	20	108	0.106	$[-0.005, +0.226]$	0.163	0.161
Fingerprint	60	108	0.061	$[-0.085, +0.210]$	0.089	0.725
Fingerprint	120	108	0.012	$[-0.083, +0.108]$	0.017	0.816
Graph	5	216	–0.089	$[-0.182, +0.006]$	–0.147	0.161
Graph	20	216	–0.105	$[-0.188, -0.028]$	–0.171	0.084
Graph	60	216	–0.070	$[-0.140, -0.000]$	–0.103	0.161
Graph	120	216	–0.129	$[-0.231, -0.023]$	–0.230	0.084
SMILES	5	216	–0.007	$[-0.080, +0.085]$	–0.009	0.816
SMILES	20	213	0.024	$[-0.037, +0.090]$	0.035	0.725
SMILES	60	216	0.014	$[-0.052, +0.075]$	0.018	0.816
SMILES	120	216	–0.021	$[-0.144, +0.094]$	–0.026	0.816

Table 5: Paired difference Δ per (input family, switch epoch N), in units of the between-run standard deviation of baseline performance on the same dataset. Confidence intervals come from a bootstrap that resamples datasets rather than individual runs; q is the Benjamini–Hochberg adjusted p-value across the twelve cells. No cell is significant, and effect sizes are uniformly small ($|d| \leq 0.23$).

Task type	n	Mean Δ	95% CI	Cohen's d	p
Classification	1437	–0.022	$[-0.075, +0.036]$	–0.029	0.429
Regression	720	–0.033	$[-0.091, +0.024]$	–0.073	0.125

Table 6: Paired difference Δ per task type, in units of the between-run standard deviation of baseline performance on the same dataset. On the standardized scale the two splits agree in sign and magnitude, so the pooled null is not produced by cancellation across task types with incomparable units.

F GEOMETRIC MEASURES VS CORRUPTION SENSITIVITY

This appendix complements §5.3, where we report that geometric summaries of the training trajectory do not predict corruption sensitivity. Each geometric measure is computed between consecutive checkpoints along the same trajectory used for volatility, and each resulting curve is summarized by the standard deviation of its first differences, so the measures track geometric displacement. We report CKA, nearest-neighbour preservation, and effective rank. The volatility row is included as a

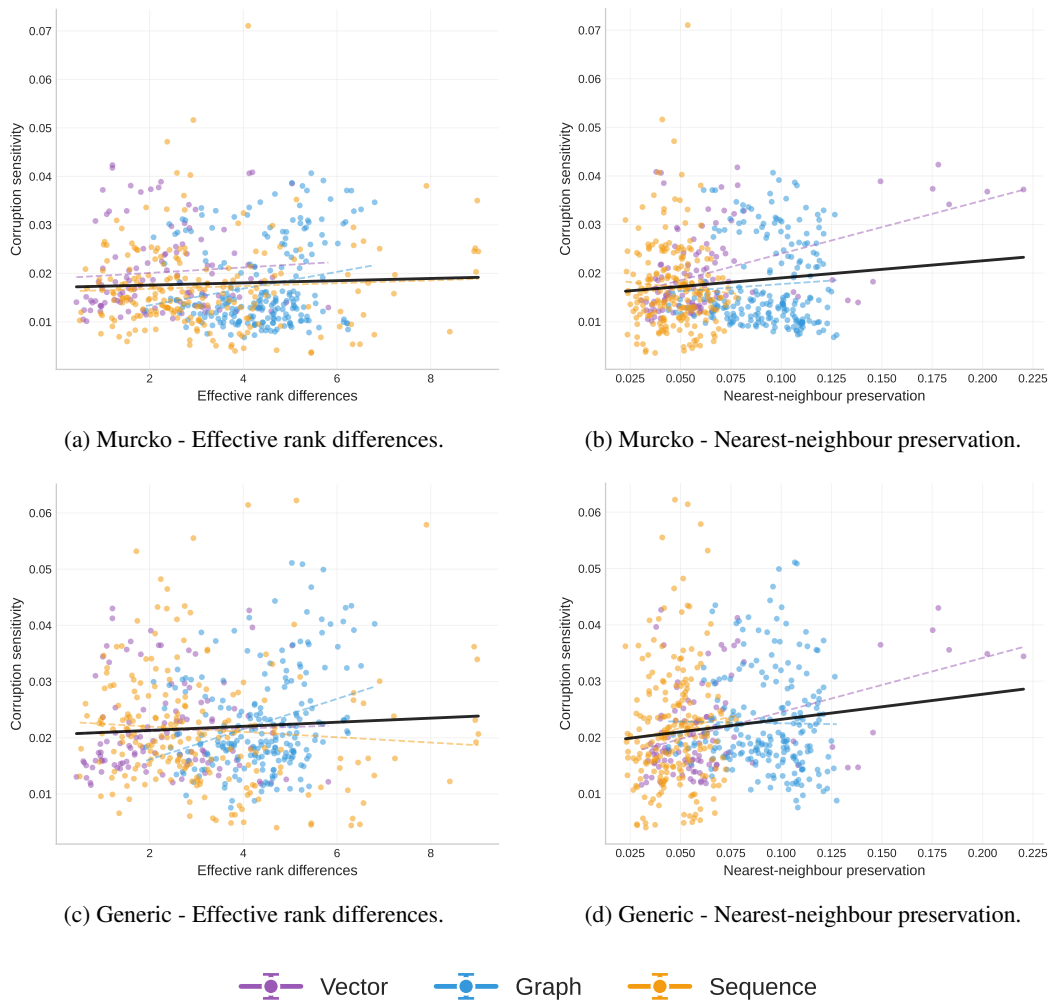
Table 8: Probed properties grouped by category. Property definitions follow standard implementations in RDKit.

Property	Category
Atomic Entropy	compositional
Halogen Count	compositional
Molecular Weight	compositional
Nitrogen Count	compositional
Oxygen Count	compositional
Sulfur Count	compositional
LogP	physicochemical
Molecular Refractivity	physicochemical
Aliphatic Rings Count	physicochemical
Aromatic Heterocycles Count	physicochemical
Aromatic Rings Count	physicochemical
H-Acceptors Count	physicochemical
H-Donors Count	physicochemical
Rotatable Bonds Count	physicochemical
TPSA	physicochemical
Amide Count	substructure counts
Amine Count	substructure counts
Carboxylic Count	substructure counts
Ether Count	substructure counts
Hydroxyl Count	substructure counts
Ketone Count	substructure counts
Sulfonamide Count	substructure counts
Balaban J Index	higher order topological
Estrada Index	higher order topological
Sombor Index	higher order topological
Spectral Radius	higher order topological
Branched Atoms Count	basic topological
Degree Entropy	basic topological
Distance Entropy	basic topological
Harary Index	basic topological
Mostar Index	basic topological
Ring Count	basic topological

reference. All entries are Spearman ρ with the post-switch normalized DTW divergence (corruption sensitivity), reported pooled and within each input family at the run grain.

Table 7: Spearman ρ between each trajectory summary and corruption sensitivity, pooled and within each input family. All measures are computed between consecutive checkpoints and summarized by the standard deviation of their first differences, so they differ from volatility only in the quantity tracked along the trajectory. Volatility is shown in bold as a reference. No geometric summary predicts corruption sensitivity, and all change sign across input families.

Measure	ρ_{pooled}	ρ_{Vector}	ρ_{Graph}	ρ_{Sequence}	n
Volatility (probe selectivity)	**+0.718**	**+0.856**	**+0.887**	**+0.670**	540
CKA	+0.046	+0.206	-0.048	-0.005	540
Nearest-neighbour preservation	+0.021	+0.413	-0.013	-0.071	540
Effective rank	-0.044	+0.128	+0.147	-0.034	540



(a) Murcko - Effective rank differences.

(b) Murcko - Nearest-neighbour preservation.

(c) Generic - Effective rank differences.

(d) Generic - Nearest-neighbour preservation.

Figure 4: Geometric summaries of embedding trajectories for both corruption types, across the three representation families (Vector, Graph, Sequence). Each point is a (model, property) pair; the solid black line is the pooled linear fit across all families. a Murcko corruption - effective rank differences; b Murcko corruption - nearest-neighbour preservation; c Generic corruption - effective rank differences; d Generic corruption - nearest-neighbour preservation.

G PROBED PROPERTIES

G.1 BEMIS–MURCKO DECOMPOSITION

The Bemis–Murcko decomposition (Bemis and Murcko, 1996), as implemented in RDKit's `MurckoScaffold` module, partitions each molecule into a structural *core* (rings and the linkers between them) and finer *peripheral components* (side chains and terminal groups attached to the core). During corrupted training, models observe only the core for the first N epochs; the full molecule is restored at epoch $N+1$ and training continues without further intervention. The partition of probed properties into *directly* and *indirectly affected* reflects which properties depend on the removed peripheral components.

G.2 PROPERTY PARTITION

The Bemis–Murcko decomposition *directly affects* eleven probed properties:

$\mathcal{P}_{\text{direct}} = \{$`Amine Count`, `Sulfonamide Count`, `Ketone Count`, `Ether Count`, `Amide Count`, `Carboxylic Count`, `Hydroxyl Count`, `Halogen Count`, `Sulfur Count`, `Nitrogen Count`, `Oxygen Count`$\}$.

All remaining probed properties are *indirectly affected*.

H GENERIC CORRUPTION

In this section we detail the *generic* corruption and present the obtained results.

Design. Under the *generic* corruption every molecule is transformed so that all of its atoms become carbon and every bond becomes single. This completely changes the atomic composition of the molecule, but leaves its topology intact. The transformation is orthogonal to the Bemis–Murcko decomposition: the Murcko corruption deletes atoms and bonds while leaving the identity of the remaining ones unchanged, whereas genericization keeps the entire adjacency structure and rewrites every label attached to it. Models are trained on genericized molecules for the first $N \in \{5, 20, 60, 120\}$ epochs and on full molecules thereafter, as in §5.2. Furthermore, since the generic corruption leaves topology intact, basic and higher-order topological properties should be invariant under this corruption, as the information needed to extract them is present throughout training. In contrast, the remaining probed properties depend on atomic composition to different degrees. This enables us to partition the probed properties into two sets that should be differentially impacted by the corruption, analogously to the partition in §5.2.

Task-level performance. When it comes to performance differences between baseline and corrupted runs, generic corruption leaves a small deficit on final performance. Corrupted runs perform 0.118 baseline standard deviations below their matched counterparts (95% CI $[-0.189, -0.046]$, $p = 0.002$, Cohen's $d = -0.12$), and the effect grows with the duration of the corruption, from -0.048 at $N = 5$ to -0.208 at $N = 120$. This dose–response effect is mostly driven by the graph family ($\rho = -0.13$, $p = 10^{-4}$) with a weaker contribution from sequences ($\rho = -0.07$, $p = 0.05$). Performance differences in vector based models are not statistically significant. Although the effect is small in absolute terms, it is consistent in sign and scales with N, in contrast to the Murcko based corruption. This difference is consistent with Achille et al. (2017), who find that early deficits to low-level input statistics can cause irreversible performance loss, whereas deficits that preserve them are recovered. Murcko based corruption preserves the local description of atoms and bonds, whereas genericization rewrites them, providing a plausible explanation about why only the latter leaves a lasting task-level mark.

Representational divergence replicates. Post-switch DTW divergence increases monotonically with N in all three families, as under the Murcko corruption, and to a larger magnitude for graphs and sequences: from 0.013 to 0.038 for graphs, 0.012 to 0.036 for sequences, and 0.015 to 0.033 for vectors. Moreover, the selective vs global disruption reported in §5.2 replicates as well. At $N = 120$, graph-based representations diverge by 0.044 on composition-dependent properties and 0.025 on topology-invariant ones, a separation of 0.019 that is roughly half their mean divergence. Vector-based representations show a separation of 0.003 (0.034 vs. 0.031) and sequence-based ones none at all (0.036 vs. 0.036). Thus, the selective divergence observed under the Murcko corruption replicates under the generic corruption, with graph-based representations again showing selective disruption, as opposed to the global disruption observed in vectors and sequences.

Volatility. Volatility, computed exactly as in §5.3, predicts corruption sensitivity under generic corruption as well ($\rho = 0.64$ pooled), and remains predictive within every input family (0.86 for vectors, 0.73 for graphs, 0.66 for sequences). Volatility therefore remains predictive under a **distinct** corruption, both across and within input families, suggesting that its relationship with corruption sensitivity is not specific to the intervention used to evaluate it. The model-selection procedure of §5.3 transfers as well: selecting, within each dataset, the run with the lowest volatility, measured on uncorrupted training and never evaluated against this corruption, reduces sensitivity to the generic corruption by 55% relative to random selection, with lower sensitivity in all twelve datasets.

I ROBUSTNESS OF THE VOLATILITY–SENSITIVITY RELATIONSHIP

The pooled correlation is not an artifact of aggregation. The 540 runs per corruption are 5 architectures $\times$ 3 featurizations $\times$ 12 datasets $\times$ 3 seeds, and runs sharing a dataset or an architecture are not independent. Nominal Spearman p-values, which assume 540 independent draws, are therefore too small. We instead compute percentile intervals from a bootstrap that resamples whole clusters, keeping all 540 runs rather than fragmenting the data into small cells. Resampling datasets, architectures, or featurizations gives 95% intervals of $[0.52, 0.82]$, $[0.64, 0.87]$, and $[0.60, 0.85]$ for the Murcko corruption, and $[0.40, 0.78]$, $[0.59, 0.77]$, and $[0.45, 0.82]$ for the generic corruption (Table 9). All intervals exclude zero by a wide margin under every choice of resampling unit.

A rank regression of corruption sensitivity on volatility with standard errors clustered on dataset results in slope $+0.718$, 95% CI $[+0.586, +0.850]$, $t = 10.6$ for the Murcko corruption, and $+0.641$, $[+0.479, +0.804]$, $t = 7.7$ for the generic one.

No single architecture or featurization drives the effect. Removing any one architecture leaves the pooled correlation in $[0.68, 0.76]$ under the Murcko corruption and $[0.62, 0.68]$ under the generic one, against full-data values of 0.72 and 0.64. Removing any one featurization leaves it in $[0.70, 0.76]$ and $[0.56, 0.71]$ respectively. The largest single deviation is the removal of the SMILES Pair Encoding featurization under the generic corruption (0.56), and even there the relationship remains strong.

The relationship is not an artifact of shared probe noise. Volatility and corruption sensitivity are both computed from probe trajectories. If their association were driven by run-level noise in probe fitting, common to predictor and target, it would disappear when the two are decoupled. We test this by replacing each run's own volatility with the mean volatility of runs sharing the same configuration: runs sharing its architecture, featurization, and dataset but differing in random seed. This estimate of volatility shares the configuration-level signal but none of this run's probe-fitting noise. The correlation is essentially unchanged, retaining 98.7% of its value pooled ($\rho = 0.709$ vs. 0.718) and between 97.5% and 101.3% within each input family (Table 10). The same holds under the generic corruption (97.8% retained).

Control probes. As a second check we compute volatility from the shuffled-label control probes used for the selectivity correction. These are fit on the same representations, so their volatility still tracks representational change and is not a pure noise channel; partialling it out is therefore a conservative correction. The partial correlation nonetheless remains substantial ($\rho = 0.48$ for the Murcko corruption, 0.37 for the generic one, both $p < 10^{-18}$).

Generalization beyond probe trajectories. Volatility and corruption sensitivity are both derived from probe trajectories. To test whether the relationship extends beyond the probing pipeline, we instead measure the divergence between the validation-loss curves of each corrupted run and its matched baseline over the post-switch epochs. Volatility is also correlated with this probe-independent measure, with a pooled Spearman $\rho = 0.31$ and within-family values of 0.55 for vector-, 0.72 for graph-, and 0.25 for sequence-based representations (all $p < 10^{-3}$). The lower pooled correlation reflects differences in the scale of validation loss across input formats. Thus, the association between volatility and corruption sensitivity is also present when the target is computed directly from validation loss, although less strongly than for the probe-based measure.

Corruption	Resampled unit	Clusters	ρ	95% CI
Murcko	Dataset	12	0.718	[0.516, 0.824]
Murcko	Architecture	5	0.718	[0.642, 0.872]
Murcko	Featurization	9	0.718	[0.596, 0.851]
Murcko	Dataset | Vector	12	0.856	[0.641, 0.905]
Murcko	Dataset | Graph	12	0.887	[0.674, 0.928]
Murcko	Dataset | Sequence	12	0.670	[0.536, 0.783]
Generic	Dataset	12	0.641	[0.403, 0.775]
Generic	Architecture	5	0.641	[0.585, 0.770]
Generic	Featurization	9	0.641	[0.452, 0.815]
Generic	Dataset | Vector	12	0.855	[0.656, 0.904]
Generic	Dataset | Graph	12	0.731	[0.341, 0.861]
Generic	Dataset | Sequence	12	0.656	[0.499, 0.787]

Table 9: Pooled Spearman ρ between volatility and corruption sensitivity, with 95% percentile intervals from a bootstrap resampling whole clusters. The point estimate is the same in every row; only the resampling unit changes.

Corruption	Scope	Own seed	Sibling seeds	Retained (%)
Murcko	Pooled	0.718	0.709	98.7
Murcko	Vector	0.856	0.835	97.5
Murcko	Graph	0.887	0.871	98.1
Murcko	Sequence	0.670	0.679	101.3
Generic	Pooled	0.641	0.627	97.8
Generic	Vector	0.855	0.815	95.3
Generic	Graph	0.731	0.703	96.3
Generic	Sequence	0.656	0.652	99.4

Table 10: Spearman ρ between volatility and corruption sensitivity when each run's own volatility is replaced by the mean volatility of its sibling seeds, which share the configuration but not the run's probe-fitting noise.

J Model selection by volatility

This appendix gives the per-dataset detail behind the selection experiment in §5.3.

Procedure. Within each dataset we consider the 45 baseline runs (5 architectures $\times$ 3 featurizations $\times$ 3 seeds) as candidate models. Random selection is scored as the mean corruption sensitivity over all candidates; selection by volatility takes the candidate with the lowest volatility; selection by validation performance takes the candidate with the best validation metric, oriented so that higher is better in both task types. All three are evaluated on the same quantity, the run's mean post-switch DTW divergence from its corrupted counterparts.

Per-dataset results. Selection by volatility gives lower corruption sensitivity than random selection in all twelve datasets, by 51% on average (Wilcoxon $p = 5 \times 10^{-4}$), and lands on or within one rank of the best available candidate in five of them. Selection by validation performance gives $+0.1\%$ on average and improves on random selection in four of twelve datasets. Table 11 reports each dataset.

Relation to validation performance. Within datasets, volatility correlates weakly with validation performance (median $\rho = 0.18$ pooled; 0.22, 0.05, and -0.04 for vector, graph, and sequence), and validation performance correlates negligibly with corruption sensitivity (median $\rho = 0.02$ pooled; 0.04, 0.14, and 0.03). The two criteria are therefore close to independent.

Dataset	n	Random	By val.	By vol.	Oracle	Vol. vs random (%)
AR	45	0.0136	0.0185	0.0051	0.0046	−62.6
BACE	45	0.0162	0.0167	0.0131	0.0115	−19.0
BBBP	45	0.0155	0.0112	0.0101	0.0081	−35.0
CACO2	45	0.0266	0.0285	0.0145	0.0125	−45.5
CYP2C19	45	0.0107	0.0099	0.0042	0.0036	−61.1
CYP3A4	45	0.0317	0.0336	0.0179	0.0144	−43.6
ER	45	0.0125	0.0106	0.0052	0.0052	−58.1
ESOL	45	0.0195	0.0134	0.0076	0.0076	−61.2
FreeSolv	45	0.0272	0.0305	0.0107	0.0103	−60.4
HIV	45	0.0140	0.0141	0.0054	0.0054	−61.5
Lipo	45	0.0115	0.0130	0.0075	0.0060	−34.5
PPAR	45	0.0162	0.0168	0.0053	0.0040	−67.5

Table 11: Corruption sensitivity of the model selected by each criterion, per dataset. *Random* is the mean over all candidates, *oracle* the best available. Selection by volatility improves on random selection in every dataset.