CORRECTING THE DROPOUT-LAYERNORM EXPECTATION GAP IMPROVES PROTEIN STRUCTURE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Although $\mathbb{E}[\mathrm{dropout}(x)] = x$, here we show that $\mathbb{E}[\mathrm{LayerNorm}(\mathrm{dropout}(x))]$ is *not* equal to $\mathrm{LayerNorm}(x)$. Accordingly, the pattern of a Dropout layer followed by a LayerNorm, which is common to many AlphaFold2-based protein structure predictors, produces a systematic bias at evaluation time that can hamper performance. To address this, we derive a closed-form, first-order correction for this gap, which we call a Dropout-LayerNorm Correction (DLC). DLC empirically matches the performance boost of large Monte Carlo dropout ensembles. We evaluate its effect across nine protein structure models (ESMFold, OpenFold, ABB3, FlashABB, Ibex, NbForge, Genie1, Genie2, Genie3) on both paired and single-chain antibody structures as well as one protein-ligand docking model (QuickBind). The correction is computationally negligible and improves accuracy in all ten models tested ($\sim 0.3\% - 13\%$), with a modest but consistent improvement in ESMFold and OpenFold and a substantial improvement in antibody-specific models. This work identifies the mathematical consequence of chaining together Dropout and LayerNorm and provides a free, principled adjustment to improve the evaluation performance of many pretrained models.

1 INTRODUCTION

Most modern neural networks are composed of a relatively small number of unique layers. Among these are dropout and normalization layers, which are used to improve generalization and stability. During training, dropout layers randomly replace each entry with a zero with probability p (Srivastava et al., 2014). This creates an imbalance during evaluation, since the magnitude of each entry will be, in expectation, $1/(1-p)$ larger. To compensate for this, standard dropout implementations, such as the PyTorch `nn.Dropout` layer, multiply by $1/(1-p)$ during training. This means that $\mathbb{E}[\mathrm{dropout}(x)] = x$ and empirically fixes the eval-mode gap (Srivastava et al., 2014). However, underlying this fix is the assumption that dropout layers are always followed by a layer which is invariant to changes in variance, such as random linear layers. In practice, sometimes dropout layers are followed by non-linear blocks such as residual+LayerNorm or, in the case of AlphaFold2, just LayerNorm (Ba et al., 2016; Vaswani et al., 2017; Dauparas et al., 2022; Jumper et al., 2021).

The final layer of each StructureModule block in AlphaFold2-style models finishes with a Transition into a BackboneUpdate (Jumper et al., 2021; Ahdritz et al., 2024). Practically, this reads as Dropout→LayerNorm→Linear→Translation (see Figure 2). This means that any bias in the expectation of dropout+LayerNorm is propagated directly into the expectation of the translation of each amino acid. In the following sections, we show that this leads to a systematic "overshoot" of approximately $5-6\%$ which materially affects the quality of predicted structures. However, we also provide a fix for this effect without the need for retraining. We derive a closed-form correction which scales roughly as $\sqrt{q} = \sqrt{1-p}$. This correction can be applied cheaply after every dropout+LayerNorm block and nearly perfectly fixes the overshoot and improves structural accuracy.

Our contributions are the following:

1. We identify a systematic bias that exists in all neural networks with a dropout layer directly followed by a LayerNorm

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.

2. We provide a closed form fix which can be patched directly into LayerNorm
3. We test our fix on ten biomolecular structure models in a therapeutically relevant protein context (immunoglobulins) and show that it improves accuracy in almost all cases

2 THEORY

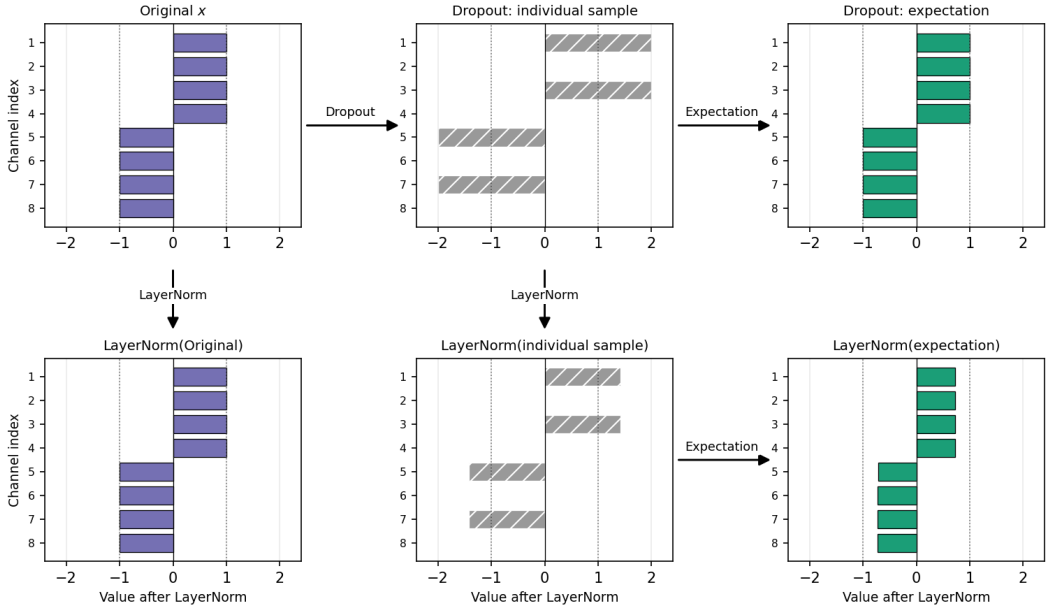


Figure 1: Toy example isolating the mechanism (d=8, q=0.5, $x = (1, 1, 1, 1, -1, -1, -1, -1)$, so μ=0, σ=1 by construction). Dropout's own expectation exactly recovers x (unbiased per channel, middle-top), and LayerNorm alone is a no-op on this particular x (bottom-left), but LayerNorm's expectation after dropout does not recover x: it shrinks to $\sim \sqrt{q} \cdot x$ (bottom-right).

2.1 SETUP

Consider a single token's pre-dropout representation $x \in \mathbb{R}^d$ immediately upstream of a LayerNorm. Inverted dropout keeps each channel independently with probability q (drop probability $p = 1 - q$) and rescales by $1/q$:

$$z_k = \frac{m_k x_k}{q}, \qquad m_k \sim \text{Bernoulli}(q) \text{ i.i.d.} \tag{1}$$

This is exactly unbiased per channel ($\mathbb{E}[z_k] = x_k$) (Srivastava et al., 2014). LayerNorm then computes

$$\text{LN}(z)_k = \gamma_k \cdot \frac{z_k - \text{mean}(z)}{\text{std}(z)} + \beta_k, \tag{2}$$

where $\text{mean}(z)$ and $\text{std}(z)$ are shared across all d channels. This sharing is the entire source of the eval-mode bias. In most cases, $\text{mean}(z) \approx 0$, and so $\text{LN}(z)_k$ is directly rescaled by changes in $\text{std}(z)$, such as the variance shift induced by dropout.

2.2 SIMPLIFIED DERIVATION (ZERO-MEAN CASE)

We give the cleanest version of the result here, under the simplifying assumption that the channel-wise mean of x is (approximately) zero, $\mu := \text{mean}_k(x_k) \approx 0$. For the non-zero mean derivation, see Appendix A.

With $\mu \approx 0$, $\mathbb{E}[\text{mean}(z)] = \mu \approx 0$, and the shared-variance expectation

$$\mathbb{E}[\text{var}(z)] \;\approx\; \frac{\sigma^2 + p\mu^2}{q} \;=\; \frac{\sigma^2}{q}, \tag{3}$$

where $\sigma^2 := \mathrm{var}_k(x_k)$. Substituting into $\mathbb{E}[\mathrm{norm}(z)_k] \approx (x_k - \mu)/\sqrt{\mathbb{E}[\mathrm{var}(z)]}$ gives

$$\mathbb{E}[\mathrm{norm}(z)_k] \;\approx\; \frac{x_k}{\sigma/\sqrt{q}} \;=\; \sqrt{q}\cdot \mathrm{norm}(x)_k. \tag{4}$$

So in the zero-mean case the correction collapses to a single global constant, $c = \sqrt{q}$, independent of μ, σ altogether. Writing $\mathrm{LN}_{\mathrm{full}}(x) := \gamma \cdot \mathrm{norm}(x) + \beta$ for LayerNorm's actual output (affine parameters included), the β-preserving eval-time correction is

$$\widehat{\mathrm{LN}}(x) \;=\; \sqrt{q}\cdot \mathrm{LN}_{\mathrm{full}}(x) \;+\; (1-\sqrt{q})\,\beta. \tag{5}$$

Only the $\gamma \cdot \mathrm{norm}(x)$ part is shrunk; β is recovered at full strength, which is exact algebraically (expand Eq. 5 and the β terms collapse to coefficient 1).

2.3 General ($\mu \neq 0$) correction

For non-zero μ, the correction factor c is well-approximated by $\sqrt{q/(1+p(\mu/\sigma)^2)}$. This is the version we use throughout the remainder of the manuscript. For a full derivation, see Appendix A. Intuitively, as the mean deviates from zero, the average effect of dropout becomes more extreme which increases the variance (and the required correction). As the $\mu \to 0$, $c \to \sqrt{q}$ which recovers the simplified case.

In Appendix A.4, we also derive a correction for the case of $\mathrm{LN}(\mathrm{dropout}(\mathrm{x}') + x)$, (i.e. a dropout vector added to the residual stream and then normalized). Here, $c = \sqrt{q/(q+p\rho^2)}$ where ρ is the ratio of the variance between x' and x. This is a common pattern in many neural networks such as GNNs and Transformers, but we found that practically, the effect was small enough to have limited impact on the models we tested. We leave further exploration as an interesting direction for future work. In Appendix B, we derive the equivalent correction for RMSNorm (Zhang & Sennrich, 2019) and BatchNorm (Ioffe & Szegedy, 2015), two other widely used normalization schemes which are not tested empirically in this paper.

3 Methods

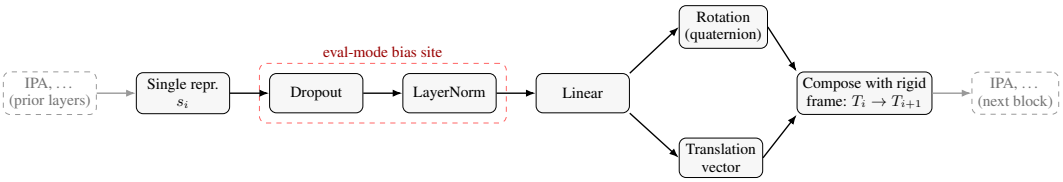


Figure 2: The dropout→LayerNorm→Linear→rigid-body-update pattern repeated at the end of every structure-module block. The single representation s_i passes through dropout and LayerNorm (the eval-mode bias site), then a Linear layer that directly outputs the block's rotation and translation update. As the biased LayerNorm output feeds the Linear layer that *is* the translation vector, the bias shows up directly as a systematic error in predicted translation length.

We test our correction on protein structure predictors which contain the Dropout→LayerNorm→Linear→Translation pattern. This includes general protein structure predictors like OpenFold (Ahdritz et al., 2024) and ESMFold (Lin et al., 2023) as well as a number of antibody-specific variants which omit the pair representation. One of the first antibody-specific AlphaFold2-inspired antibody structure predictors was ImmuneBuilder (Abanades et al., 2023) which uses an analog of Invariant Point Attention with no pair processing. Several similar models have since emerged including ABodyBuilder3 (ABB3) (Kenlay et al., 2024), FlashABB (Ellmen et al., 2026), and Ibex (Dreyer et al., 2025) for antibody structure prediction and NbForge (Ali et al., 2026) for nanobody/VHH structure prediction. Similarly, Genie1 (Lin & AlQuraishi, 2023), Genie2 (Lin et al., 2024), and Genie3 (Lin et al., 2026) use a variant of the OpenFold structure module for general protein structure generation.

Perhaps notably, many variants of OpenFold contain changes which avoid the phenomenon discussed in this work. For instance, ImmuneBuilder uses the default 10% dropout during the first

stage of training but then uses 0% dropout during the finetuning stage. Similarly, FrameFlow uses a dropout rate of 0% throughout the StructureModule (Yim et al., 2024). AlphaFold3-style models contain residual connections in the updates which may help to mitigate the variance shift (Abramson et al., 2024; Wohlwend et al., 2024).

We restrict our experiments to the study of antibodies and nanobodies. This ensures a fair comparison between the models and allows us to use the established practice of measuring full-structure and Complementarity-Determining Region H3 (CDRH3)-specific RMSDs; CDRH3 is the most challenging region of an immunoglobulin to model. For each model, we create a hook which fires post-LayerNorm at every site which is directly preceded by a dropout layer. In most models this happens twice per block: once in the IPA transition and once in the backbone transition. In Ibex, there is an additional hook which fires after the ESM-C language model embeddings (Candido et al., 2026; Dreyer et al., 2025). In NbForge, there is an additional hook for the pair updates (Ali et al., 2026).

4 Experiments

4.1 Translation length is systematically wrong, and the correction fixes it

We first sought to demonstrate that the dropout+LayerNorm eval gap does actually produce the predicted overshoot. We performed an MC simulation of 100 random seeds with dropout on and measured the ratio of the length of the first translation (block 0) between the deterministic models (corrected + uncorrected) and the MC average for ABB3. The results are shown in Figure 3. We found that the uncorrected version did tend to overshoot the MC average by about 5.6% ($\sim 1/\sqrt{0.9}$) whereas our corrected version nearly perfectly matched the mean translation. We also measured the ratio of the distance traveled to the total displacement over all residues for the corrected and uncorrected models. The uncorrected model had to travel about 18% further, consistent with a trajectory of chronically overshooting and needing to correct at every block.

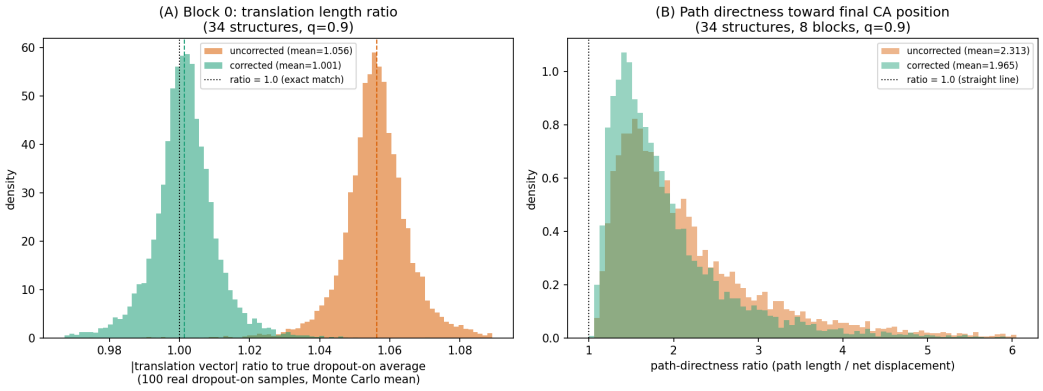


Figure 3: ABB3, both-sites corrected vs. uncorrected, 34-structure test subset, $q = 0.9$. **(A)**: ratio of the first structure-module block's predicted backbone-translation vector length to the true 100-sample Monte Carlo dropout-on average (mean $1.056 \rightarrow 1.001$). **(B)**: path-directness ratio (aggregate per-block Cα displacement across all 8 blocks divided by net displacement from block 0 to block 7). Here, 1.0 corresponds to a perfectly straight path whereas higher values are associated with more "zigzagging". The corrected mean drops from 2.313 to 1.965, showing the block-0 magnitude bias is not an isolated artifact but part of a systematically less direct trajectory toward the final structure.

4.2 Theory matches the true Monte Carlo expectation

We tested the impact of the correction on the final RMSD error of ABB3 structure predictions. This test broadly matches similar works in MC dropout (Srivastava et al., 2014; Gal & Ghahramani, 2016; Li et al., 2018) where the mean prediction (coordinates) is taken over n distinct samples. The results

are shown in Figure 4. When measured over the entire structure and over only the CDRH3, the single-sample dropout-on evaluation was more accurate than the (standard) dropout-off evaluation. Taking the mean coordinates improved accuracy monotonically with the number of samples. This agrees with a model where dropout introduces an extra noise term which decays as $O(\frac{1}{\sqrt{n}})$. The MC simulation also approached a structure with a similar RMSD to our corrected structure which shows that our correction maps well to the unbiased dropout-on mean. Thus, our correction is a cheap approximation of the MC mean which also avoids complications with nonphysical structures induced by averaging coordinates.

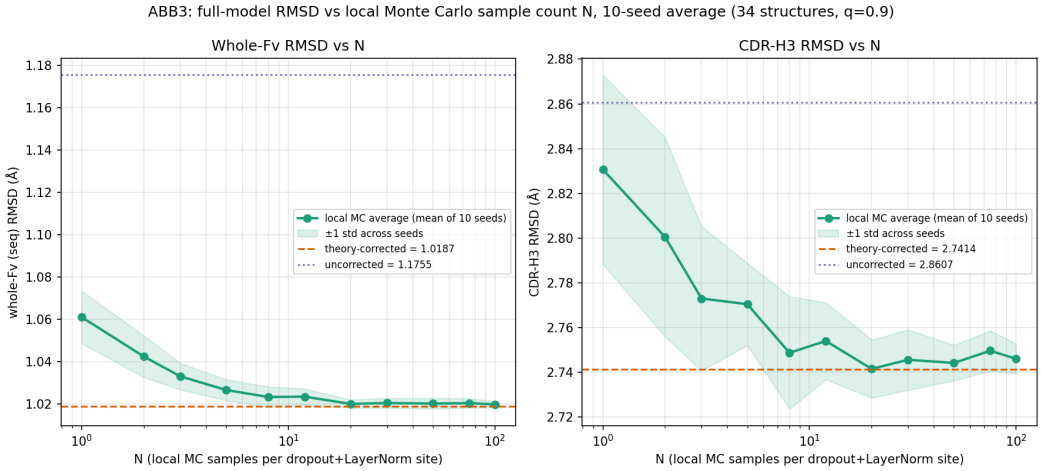


Figure 4: ABB3-predicted whole-Fv and CDR-H3 test RMSD as a function of N, the number of Monte Carlo samples used at every dropout+LayerNorm site (log-x), averaged over 10 independent seeds (shaded band: ± 1 std across seeds). From $N{=}1$ the mean already sits well below the uncorrected line since we are introducing noise but removing bias. Both the mean and the across-seed spread converge onto the theory-corrected line by $N \approx 30$, showing that the theory corrects to an unbiased approximation of the MC RMSD.

4.3 Expanding the Pareto front of antibody structure prediction

As shown in the previous section, the corrected version of ABB3 was substantially more accurate than the uncorrected default. We also tested the effect on FlashABB, Ibex, ESMFold, and OpenFold. ESMFold and OpenFold were both trained for monomer structure prediction and so had relatively weak performance when evaluated on antibodies using the gap trick (Mirdita et al., 2022; Evans et al., 2022). The full results are shown in Appendix D.1 alongside ABodyBuilder2. We found that three models contributed to the Pareto front of speed and accuracy: FlashABB, ABB3, and Ibex. The correction results for these three models are shown in Figure 5. The correction substantially closes the accuracy gap between the models, including a full-Fv RMSD improvement of 10.9% for FlashABB and 13.3% for ABB3. This enables FlashABB to achieve an RMSD within $\sim$0.02Å of Ibex while operating $\sim$100x faster.

We also saw near-uniform improvements in CDRH3 accuracy, with the exception of Ibex. CDRH3 accuracy is incredibly noisy due to large structural variations and small sample size. As a check on whether the negative ensemble CDR-H3 result above is a genuine correction/ensembling interaction or a small-sample artifact of the 34-structure subset, Appendix D.2 repeats the same comparison on an independent, larger sample of 100 structures from Ibex's own training set and sees an improvement over CDRH3.

4.4 Improved nanobody structure prediction

Nanobodies (or VHHs) are an increasingly popular class of therapeutic due to their small size which allows them to access tighter epitopes and makes them easier to manufacture and design (Jovčevska & Muyldermans, 2020). In contrast to regular antibodies which contain two (paired)

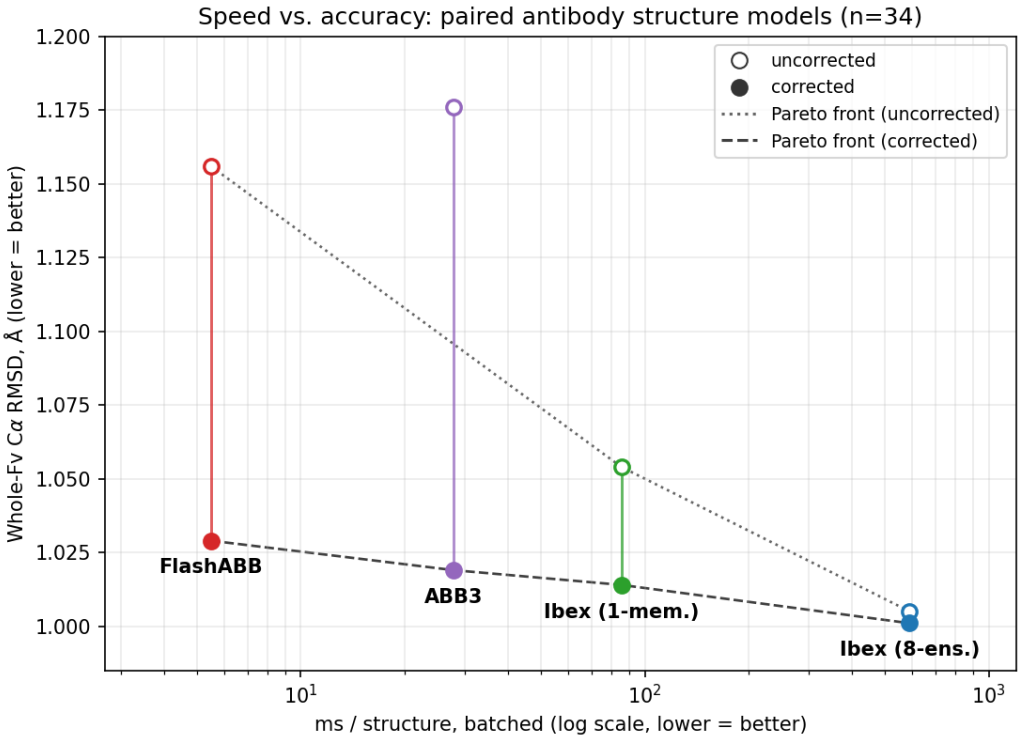


Figure 5: GPU inference speed vs. whole-Fv accuracy for the paired antibody models which comprise the Pareto front of speed and accuracy (FlashABB, ABB3, and Ibex). Dashed/dotted lines trace the Pareto front (no model both faster and more accurate) before and after correction. FlashABB and ABB3 are both single-member models, and would presumably derive a similar benefit from taking an ensemble over 8 models.

chains, nanobodies are comprised of only a single chain. We tested the benefit of our correction on popular nanobody structure prediction tools NanoBodyBuilder2 (NBB2), Ibex, and NbForge. We also tested the nanobody structure performance of general protein structure predictors ESMFold and OpenFold. All models were tested on the test set of the SAbDab2 release (Capel et al., 2026). Note that this contains substantial overlap with the training set of some of these models and so absolute performance should be interpreted with caution. However, as shown in Table 1, we saw an improvement for whole-domain structure prediction in all models under our correction. We also saw an improvement in CDRH3 performance in all models except for NbForge. As with the Ibex paired antibody results from Section 4.3, the apparent NbForge performance hit is likely due to noise in CDRH3 structure prediction and does not hold up under different random subsets of SAbDab2-nano.

4.5 More reliable confidence prediction

Many antibody structure prediction models include a pLDDT head which can be used to estimate the error in single predictions (Kenlay et al., 2024). Unlike in ESMFold/AlphaFold2, the pLDDT head for most antibody structure predictors is conditioned on the single representation (which suffers from the dropout+LN bias). We tested whether our correction improved models' own estimates of the uncertainty of their predictions. The pLDDT heads are trained to predict per-residue LDDT which we averaged over the entire sequence and over only the CDRH3 residues. The results are shown in Table 2. Adding our correction substantially improved ABB3 and FlashABB pLDDT accuracy. The Spearman correlation was nearly 2x over the entire sequence for FlashABB, in line with a much better calibrated signal. We also saw a strong improvement in the whole-domain Spearman correlation for NbForge, although only a modest improvement in the other measures. Both Ibex models saw little to no improvement in pLDDT after applying the correction. The details of Ibex

Table 1: Nanobody (VHH) full-backbone RMSD, uncorrected vs. both-sites corrected (both-sites+plm_in for Ibex), 100 real held-out VHH structures. NbForge is shown both correcting its usual two sites (IPA, transition_s) and correcting all three (adding transition_z, on the pair representation). Correcting the third site slightly hurts performance, although a cross-validated sweep of q found that $q = 0.94$ recovered performance (not shown). *The precise training details for ESMFold, Ibex, and NbForge are not publicly available, so it is possible they were finetuned with a lower dropout rate. †NBB2's (NanoBodyBuilder2) structure module (same public `ImmuneBuilder` package as ABB2) finetunes with no dropout and so cannot be corrected.

Model	Condition	Full VHH	Impr.	CDR-H3	Impr.
NBB2 (1 member)†	–	1.241	–	2.374	–
NBB2 (4-member ensemble)†	–	1.189	–	2.202	–
Ibex (1 member)*	Raw	1.123	–	2.057	–
	+DLC	1.101	+2.0%	1.998	+2.9%
Ibex (8-member ensemble)*	Raw	1.099	–	2.011	–
	+DLC	1.097	+0.2%	2.000	+0.6%
NbForge	Raw	1.108	–	1.902	–
	+DLC (2 sites)	1.072	+3.3%	1.912	−0.5%
	+DLC (3 sites)	1.090	+1.7%	1.930	−1.5%
ESMFold (no recycling)*	Raw	2.080	–	5.014	–
	+DLC	2.069	+0.5%	4.981	+0.6%
OpenFold (no recycling)	Raw	2.085	–	4.797	–
	+DLC	2.068	+0.8%	4.741	+1.2%

training are not public, so it is possible that these models were finetuned with zero dropout, and/or that the pLDDT head was trained using frozen eval weights of the main model.

Table 2: Our correction improves models' predicted uncertainty (pLDDT) accuracy. The mean absolute error (MAE) was reduced and the Pearson and Spearman correlations were improved for all models except Ibex. It is possible that the Ibex pLDDT head was trained with frozen structure prediction weights and no dropout.

		Variable region			CDR-H3		
Model	Condition	MAE	Pearson	Spearman	MAE	Pearson	Spearman
ABB3	Raw	4.233	−0.648	−0.265	12.525	−0.670	−0.787
	+DLC	3.291	−0.681	−0.425	10.400	−0.664	−0.788
FlashABB	Raw	4.641	−0.625	−0.199	11.002	−0.680	−0.772
	+DLC	3.579	−0.680	−0.396	10.035	−0.685	−0.774
Ibex (paired)*	Raw	3.271	−0.647	−0.294	11.997	−0.630	−0.759
	+DLC	3.287	−0.651	−0.372	11.820	−0.616	−0.745
Ibex (nano)*	Raw	5.613	−0.598	−0.493	8.674	−0.689	−0.673
	+DLC	5.715	−0.611	−0.493	8.896	−0.705	−0.646
NbForge	Raw	6.018	−0.520	−0.410	8.811	−0.599	−0.668
	+DLC	5.964	−0.560	−0.556	8.596	−0.600	−0.708

4.6 THE GENIE FAMILY

Genie1 (Lin & AlQuraishi, 2023), Genie2 (Lin et al., 2024), and Genie3 (Lin et al., 2026) are generative backbone design models based on the OpenFold architecture under a diffusion framework. The Genie models share the same structure module architecture to the regression models above,

including identical dropout→LayerNorm sites. We evaluate all three with the same protocol: at 100 evenly spaced diffusion timesteps t (the Genie3 default) we sample noise, run the denoiser, and compare the corrected vs. uncorrected noise-prediction loss, on the 100-nanobody test set.

The three models differ slightly in which dropout+LayerNorm sites exist. All three share the structure module's 2-sites-per-block pattern, but Genie1 uses only 5 structure-net blocks, whereas Genie2 and Genie3 both use 8. Genie3's trunk contains an additional single representation IPA/transition block. This gives 10 corrected sites for Genie1, 16 for Genie2, and 26 for Genie3 (8 structure-net blocks $\times$ 2, plus 5 trunk blocks $\times$ 2).

The results are shown in Table 3. Our correction improved performance in all models, with a particularly strong effect in Genie 1 and 2. The correction closes most of the performance gap between Genie 2 and 3 over the nanobody set.

Table 3: Genie1/Genie2/Genie3, LN-level correction vs. diffusion timestep t (mean over 100 evenly spaced t values), on the 100-nanobody set.

	Sites	Uncorrected loss	+DLC-corrected loss	Impr.
Genie1	10	1.1444	1.1097	$+3.03\%$
Genie2	16	0.8372	0.8145	$+2.71\%$
Genie3	26	0.8064	0.7955	$+1.35\%$

In Figure 8 (Appendix D), we additionally plot the magnitudes and directions of the corrected and uncorrected score vectors compared to the Monte Carlo mean and the true noise for Genie3 over two random monomers. As with the ABB3 example, the uncorrected model consistently overshoots the Monte Carlo mean with dropout on, whereas the corrected version matches it almost exactly. However, all models consistently undershoot the true noise which may mean that although the correction helps the model loss, it may lead to overly conservative structures in practice. We leave further investigations into the impact on designs for future work.

4.7 QUICKBIND: PROTEIN-LIGAND DOCKING

QuickBind (Treyde et al., 2024) is a protein-ligand docking model based on the OpenFold IPA implementation. As with the previous models, it contains the dropout+LayerNorm bias in the IPA and structure module transitions. We evaluated the impact of our correction on the RMSD of QuickBind predictions over the PoseBusters Benchmark set (Buttenschoen et al., 2023) (428 protein-ligand complexes; 427 after excluding one receptor over 2000 residues).

Compared to antibody structure prediction, docking models are more likely to predict structures in the wrong mode entirely. In these cases, a more faithful realization of the model's underlying belief of the pose does not necessarily translate to improved RMSD. To isolate the two factors (epitope selection vs. placement), we show the RMSD improvement over the whole set and over only those predictions which were sufficiently close to the ground truth in the uncorrected prediction. The results are shown in Table 4.

Table 4: QuickBind ligand-atom RMSD, uncorrected vs. corrected, on the PoseBusters Benchmark set, restricted to structures where the RMSD of the uncorrected prediction is within a cutoff. "Improved" is the fraction of structures where the corrected prediction has lower RMSD than uncorrected. t is the paired t-statistic on the per-structure corrected−uncorrected difference ($|t| \gtrsim 1.96$ is significant at $p < 0.05$).

Uncorrected RMSD threshold	n	Raw	+DLC	Impr.	Improved	t
< 2 Å	70	1.332	1.313	+1.40%	46/70	-1.68
< 3 Å	120	1.815	1.787	+1.54%	76/120	-3.08
< 5 Å	204	2.668	2.655	+0.50%	119/204	-1.19
None (full set)	427	9.384	9.343	+0.43%	240/427	-1.07

Over the full benchmark the effect is small and not statistically significant ($t = -1.07$). However, when restricted to structures which were already close to correct without the correction, the effect becomes much more consistent. At the < 3 Å cutoff the improvement is significant ($t = -3.08$, $p \approx 0.002$) with a 63/37 skew toward improvement. The < 2 Å threshold is a very strong filter for positive uncorrected results, which may be artificially excluding predictions for which the correction would help.

5 DISCUSSION

In this work we introduce DLC, a computationally negligible fix to a systematic bias in neural networks containing dropout layers directly upstream of LayerNorms. The correction is computable in closed form and reliably improves performance across all 10 models tested. In some models, such as ESMFold, the effect is small but consistent. In others, such as ABB3, the effect is large enough to meaningfully expand the Pareto front for antibody structure prediction.

Why does the bias seem to affect some models more than others? One hypothesis is that some models learn a more robust capacity to correct in the later layers. For instance, ESMFold and AlphaFold2/OpenFold have shared weights for the layers in the structure module. This means that if they overshoot after the first layer, they can likely close the distance in future layers. This is in contrast to antibody structure prediction models like ABB3 which learn separate weights in each layer. This may mean that these models end up learning to take single large jumps in the early layers and use the later layers for small structural tweaks. If there is bias in the large early jumps, this may be unrecoverable later on.

One of the questions this work allows us to answer is why the relatively straightforward combination of dropout+LayerNorm is so rare in frontier models. The bias induced by this pattern means that, paradoxically, higher dropout rates actually degrade generalization performance. The previous solution has been to minimize the variance gap by either reducing the dropout rate or introducing a residual connection. The residual pattern is nearly ubiquitous in modern machine learning: it is contained in GNNs like ProteinMPNN (Dauparas et al., 2022), the pair module of AlphaFold2/3 (Jumper et al., 2021; Abramson et al., 2024), and almost all Transformer-based language models (Vaswani et al., 2017; Touvron et al., 2023). We ran a small number of experiments on small models with residual connections and found that, with a dropout rate of 10% and typical variance ratios between the layer representations and the residual stream, this strategy does seem to practically solve the expectation gap. However, it may be that larger models are more sensitive to the accumulated error or that dropout rates exceeding 10% push the expectation gap far enough to impact generalization performance. DLC offers a fairer assessment of the impact of higher dropout rates which may improve generalization in the training of frontier models.

Additionally, there are some models which show that there are benefits of dropout+LayerNorm. CogView demonstrated that separating normalization from the residual stream (thereby introducing a dropout+LayerNorm) improved numerical stability of their vision Transformer (Ding et al., 2021). It is likely that CogView would be improved by implementing DLC. Our work unlocks the ability to experiment with models that include this motif, without the (previously) inexplicable validation penalty they normally incur. It is possible that this will lead to developments in future models for protein structure prediction/design, image processing, natural language, and more.

AI USE STATEMENT

In this work, we used generative AI tools to polish writing, help develop theoretical models, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, refine hypotheses, provide feedback on research methodology or experiments, implement methods, clean and reformat datasets, and support qualitative and thematic data analysis.

We have not used generative AI tools to assist with translation, generate synthetic datasets, or interpret results. Additionally, we used generative AI tools to create/edit software code (including for visualization) and draft boilerplate sections of the manuscript which were later rewritten (except for certain pieces of the Appendix). We have reviewed all AI-assisted work. AI was not used to formulate the key idea behind the DLC correction, but it was used to assist in solving for the closed-form versions. We have independently checked the proofs, manually searched for relevant literature and manually compared the mathematical advances, and reviewed the patch code for applying the correction. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

In Section 2, and the Appendix, we provide a complete description of the mathematical correction that was applied to all models discussed in this work. The antibody datasets are standard in the field, and the models can be run using existing code except for the application of the DLC patch. We will provide code to reproduce all experiments exactly.

REFERENCES

Brennan Abanades, Wing Ki Wong, Fergus Boyles, Guy Georges, Alexander Bujotzek, and Charlotte M. Deane. ImmuneBuilder: Deep-Learning models for predicting the structures of immune proteins. *Communications Biology*, 6(1):1–8, May 2023. ISSN 2399-3642. doi: 10.1038/s42003-023-04927-7. URL https://www.nature.com/articles/s42003-023-04927-7. Number: 1.

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, June 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07487-w. URL https://www.nature.com/articles/s41586-024-07487-w.

Gustaf Ahdritz, Nazim Bouatta, Christina Floristean, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J. O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, Arkadiusz Nowaczynski, Bei Wang, Marta M. Stepniewska-Dziubinska, Shang Zhang, Adegoke Ojewole, Murat Efe Guney, Stella Biderman, Andrew M. Watkins, Stephen Ra, Pablo Ribalta Lorenzo, Lucas Nivon, Brian Weitzner, Yih-En Andrew Ban, Shiyang Chen, Minjia Zhang, Conglong Li, Shuaiwen Leon Song, Yuxiong He, Peter K. Sorger, Emad Mostaque, Zhao Zhang, Richard Bonneau, and Mohammed AlQuraishi. OpenFold: retraining AlphaFold2 yields new insights into its learning mechanisms and capacity for generalization. *Nature Methods*, 21 (8):1514–1524, August 2024. ISSN 1548-7105. doi: 10.1038/s41592-024-02272-z. URL https://www.nature.com/articles/s41592-024-02272-z.

Montader Ali, Mateusz Jaskolowski, Matthew Greenig, Tung H. Nguyen, Mia Crnogaj, Eva Smorodina, Aubin Ramon, Haowen Zhao, Monica L. Fernández-Quintero, Elodie Ghedin, Victor Greiff, and Pietro Sormanni. Disulphide and sequence-encoded conformational priors guide nanobody structure prediction, February 2026. URL https://www.biorxiv.org/content/10.64898/2026.02.13.705647v1. ISSN: 2692-8205 Pages: 2026.02.13.705647 Section: New Results.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization, July 2016. URL http://arxiv.org/abs/1607.06450. arXiv:1607.06450 [cs, stat].

Martin Buttenschoen, Garrett M. Morris, and Charlotte M. Deane. PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences, August 2023. URL http://arxiv.org/abs/2308.05777. arXiv:2308.05777 [physics, q-bio].

Salvatore Candido, Thomas Hayes, Alexander Derry, Roshan Rao, Zeming Lin, Robert Verkuil, Bryan Z. Wu, Jin Sub Lee, Elise S. Bruguera, Jehan A. Keval, Mykhailo Kopylov, John E. Pak, Wesley Wu, Neil Thomas, Samson Mataraso, Alvin Hsu, Ashton C. Trotman-Grant, Kilian Fatras, Allan dos Santos Costa, Rohil Badkundri, Halil Akın, Deniz Oktay, Jonathan Deaton, Elizabeth Montabana, Hrishita Sitwala, Yue Yu, Marius Wiggert, Dylan Alexander Carlin, Anthony W. Goering, Tomasz Blazejewski, McCullen Sandora, Michael Hla, Tina Z. Jia, Leon H. Kloker, Nicholas J. Sofroniew, Masatoshi Uehara, Jassi Pannu, Sharrol Bachas, Daniel S. Liu, Tom Sercu, and Alexander Rives. Language Modeling Materializes a World Model of Protein Biology, June 2026. URL https://www.biorxiv.org/content/10.64898/2026.06.03.729735v1. ISSN: 2692-8205 Pages: 2026.06.03.729735 Section: New Results.

Henriette L. Capel, Odysseas Vavourakis, Benjamin H. Williams, Christopher R. Taylor, and Charlotte M. Deane. SAbDab2: The structural antibody database in the age of machine learning, June 2026. URL https://www.biorxiv.org/content/10.64898/2026.06.16.732554v1. ISSN: 2692-8205 Pages: 2026.06.16.732554 Section: New Results.

J. Dauparas, I. Anishchenko, N. Bennett, H. Bai, R. J. Ragotte, L. F. Milles, B. I. M. Wicky, A. Courbet, R. J. de Haas, N. Bethel, P. J. Y. Leung, T. F. Huddy, S. Pellock, D. Tischer, F. Chan, B. Koepnick, H. Nguyen, A. Kang, B. Sankaran, A. K. Bera, N. P. King, and D. Baker. Robust deep learning–based protein sequence design using ProteinMPNN. *Science*, 378(6615):49–56, October 2022. doi: 10.1126/science.add2187. URL https://www.science.org/doi/10.1126/science.add2187.

Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, and Jie Tang. CogView: Mastering Text-to-Image Generation via Transformers, November 2021. URL http://arxiv.org/abs/2105.13290. arXiv:2105.13290 [cs.CV].

Frédéric A. Dreyer, Jan Ludwiczak, Karolis Martinkus, Brennan Abanades, Robert G. Alberstein, Pan Kessel, Pranav Rao, Jae Hyeon Lee, Richard Bonneau, Andrew M. Watkins, and Franziska Seeger. Conformation-aware structure prediction of antigen-recognizing immune proteins. *mAbs*, 18(1):2602217, December 2025. ISSN 1942-0862. doi: 10.1080/19420862.2025.2602217. URL https://doi.org/10.1080/19420862.2025.2602217.

James Dunbar and Charlotte M. Deane. ANARCI: antigen receptor numbering and receptor classification. *Bioinformatics*, 32(2):298–300, January 2016. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv552. URL https://doi.org/10.1093/bioinformatics/btv552.

Isaac Ellmen, David Errington, Matthew I. J. Raybould, and Charlotte M. Deane. Modelling antibody structures at the speed of language, June 2026. URL https://www.biorxiv.org/content/10.64898/2026.06.03.729879v1. ISSN: 2692-8205 Pages: 2026.06.03.729879 Section: New Results.

Richard Evans, Michael O'Neill, Alexander Pritzel, Natasha Antropova, Andrew Senior, Tim Green, Augustin Žídek, Russ Bates, Sam Blackwell, Jason Yim, Olaf Ronneberger, Sebastian Bodenstein, Michal Zielinski, Alex Bridgland, Anna Potapenko, Andrew Cowie, Kathryn Tunyasuvunakool, Rishub Jain, Ellen Clancy, Pushmeet Kohli, John Jumper, and Demis Hassabis. Protein complex prediction with AlphaFold-Multimer, March 2022. URL https://www.biorxiv.org/content/10.1101/2021.10.04.463034v2. Pages: 2021.10.04.463034 Section: New Results.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1050–1059. PMLR, June 2016. URL https://proceedings.mlr.press/v48/gal16.html.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, March 2015. URL http://arxiv.org/abs/1502.03167. arXiv:1502.03167 [cs.LG].

Ivana Jovčevska and Serge Muyldermans. The Therapeutic Potential of Nanobodies. *Biodrugs*, 34 (1):11–26, 2020. ISSN 1173-8804. doi: 10.1007/s40259-019-00392-z. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC6985073/.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL https://www.nature.com/articles/s41586-021-03819-2. Number: 7873.

Henry Kenlay, Frédéric A Dreyer, Daniel Cutting, Daniel Nissley, and Charlotte M Deane. ABodyBuilder3: improved and scalable antibody structure predictions. *Bioinformatics*, 40(10):btae576, October 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae576. URL https://doi.org/10.1093/bioinformatics/btae576.

Xiang Li, Shuo Chen, Xiaolin Hu, and Jian Yang. Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift, January 2018. URL http://arxiv.org/abs/1801.05134. arXiv:1801.05134 [cs.LG].

Yeqing Lin and Mohammed AlQuraishi. Generating Novel, Designable, and Diverse Protein Structures by Equivariantly Diffusing Oriented Residue Clouds, June 2023. URL http://arxiv.org/abs/2301.12485. arXiv:2301.12485 [q-bio.BM].

Yeqing Lin, Minji Lee, Zhao Zhang, and Mohammed AlQuraishi. Out of Many, One: Designing and Scaffolding Proteins at the Scale of the Structural Universe with Genie 2, May 2024. URL http://arxiv.org/abs/2405.15489. arXiv:2405.15489 [q-bio.BM].

Yeqing Lin, Minji Lee, Aakarsh Vermani, Ellena Jiang, Sebastiaan De Cooman, Matej Špeťko, and Mohammed AlQuraishi. Fast and Ultra-Capable Protein Design: Advancing the Frontier Through Atomistic SE(3)-Equivariance with Genie 3, May 2026. URL https://www.biorxiv.org/content/10.64898/2026.05.01.722168v1. ISSN: 2692-8205 Pages: 2026.05.01.722168 Section: New Results.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, March 2023. doi: 10.1126/science.ade2574. URL https://www.science.org/doi/10.1126/science.ade2574.

Milot Mirdita, Konstantin Schütze, Yoshitaka Moriwaki, Lim Heo, Sergey Ovchinnikov, and Martin Steinegger. ColabFold: making protein folding accessible to all. *Nature Methods*, 19(6):679–682, June 2022. ISSN 1548-7105. doi: 10.1038/s41592-022-01488-1. URL https://www.nature.com/articles/s41592-022-01488-1. Number: 6.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. ISSN 1533-7928. URL http://jmlr.org/papers/v15/srivastava14a.html.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023. URL http://arxiv.org/abs/2302.13971. arXiv:2302.13971 [cs.CL].

Wojtek Treyde, Seohyun Chris Kim, Nazim Bouatta, and Mohammed AlQuraishi. QuickBind: A Light-Weight And Interpretable Molecular Docking Model, October 2024. URL http://arxiv.org/abs/2410.16474. arXiv:2410.16474 [q-bio.BM].

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, December 2017. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs].

Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Tally Portnoi, Itamar Chinn, Jacob Silterra, Tommi Jaakkola, and Regina Barzilay. Boltz-1 Democratizing Biomolecular Interaction Modeling, November 2024. URL https://www.biorxiv.org/content/10.1101/2024.11.19.624167v1. Pages: 2024.11.19.624167 Section: New Results.

Jason Yim, Andrew Campbell, Emile Mathieu, Andrew Y. K. Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S. Veeling, Frank Noé, Regina Barzilay, and Tommi S. Jaakkola. Improved motif-scaffolding with SE(3) flow matching, July 2024. URL http://arxiv.org/abs/2401.04082. arXiv:2401.04082 [q-bio.QM].

Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization, October 2019. URL http://arxiv.org/abs/1910.07467. arXiv:1910.07467 [cs.LG].

A FULL ($\mu \neq 0$) DERIVATION

This appendix gives the full derivation of the general correction $c(\mu, \sigma) = \sqrt{q/(1 + p(\mu/\sigma)^2)}$ stated in Section 2.2 without the zero-mean simplifying assumption, and its damped extension to residual-bypass sites. Notation follows Section 2: $x \in \mathbb{R}^d$ is the pre-dropout activation, $\mu := \mathrm{mean}_k(x_k)$, $\sigma^2 := \mathrm{var}_k(x_k)$, $z_k = m_k x_k/q$ with $m_k \sim \mathrm{Bernoulli}(q)$ i.i.d., $p = 1 - q$. All approximations drop $O(1/d)$ terms, which vanish as the channel dimension d grows and are already negligible at the channel widths used throughout this paper.

A.1 EXACT PER-CHANNEL MOMENTS OF z

Since $\mathbb{E}[m_k] = q$ and $\mathrm{Var}(m_k) = pq$,

$$\mathbb{E}[z_k] = x_k, \qquad \mathrm{Var}(z_k) = \frac{\mathrm{Var}(m_k)}{q^2}\, x_k^2 = \frac{p}{q}\, x_k^2, \tag{6}$$

and z_k, z_j are independent for $k \neq j$ (the masks are drawn independently per channel).

A.2 MEAN AND VARIANCE OF THE LAYERNORM STATISTICS

Write $M := \mathrm{mean}_k(z_k) = \frac{1}{d}\sum_k z_k$. Linearity gives $\mathbb{E}[M] = \frac{1}{d}\sum_k x_k = \mu$ exactly, for any μ (this does not require the zero-mean assumption). For the shared variance $V := \mathrm{mean}_k(z_k^2) - M^2$, first

$$\mathbb{E}\left[\frac{1}{d}\sum_k z_k^2\right] = \frac{1}{d}\sum_k \left(\mathrm{Var}(z_k) + \mathbb{E}[z_k]^2\right) = \frac{1}{d}\sum_k \left(\frac{p}{q} + 1\right) x_k^2 = \frac{\sigma^2 + \mu^2}{q}, \tag{7}$$

using $\frac{1}{d}\sum_k x_k^2 = \sigma^2 + \mu^2$ and $1 + p/q = 1/q$. Since $\mathrm{Var}(M) = \frac{1}{d^2}\sum_k \mathrm{Var}(z_k) = \frac{p(\sigma^2+\mu^2)}{qd} = O(1/d)$, $\mathbb{E}[M^2] = \mathrm{Var}(M) + \mu^2 \approx \mu^2$, and therefore

$$\mathbb{E}[V] \;\approx\; \frac{\sigma^2 + \mu^2}{q} - \mu^2 \;=\; \frac{\sigma^2 + p\mu^2}{q}, \tag{8}$$

which is exactly the shared-variance expectation quoted (for the zero-mean case only) in Section 2.2 – here it holds for general μ.

A.3 THE GENERAL CORRECTION

Treating V as concentrated near its mean (the same leading-order decoupling of numerator and denominator used in Section 2.2),

$$\mathbb{E}[\mathrm{norm}(z)_k] \;\approx\; \frac{\mathbb{E}[z_k] - \mathbb{E}[M]}{\sqrt{\mathbb{E}[V]}} \;=\; \frac{x_k - \mu}{\sqrt{(\sigma^2 + p\mu^2)/q}} \;=\; \sqrt{\frac{q}{1 + p(\mu/\sigma)^2}} \cdot \mathrm{norm}(x)_k. \tag{9}$$

Defining

$$c(\mu, \sigma) \;:=\; \sqrt{\frac{q}{1 + p(\mu/\sigma)^2}}, \tag{10}$$

the same β-preserving algebra as Eq. 5 applies unchanged, with $c(\mu, \sigma)$ in place of $\sqrt{q}$:

$$\widehat{\mathrm{LN}}(x) \;=\; c(\mu, \sigma) \cdot \mathrm{LN}_{\mathrm{full}}(x) \;+\; (1 - c(\mu, \sigma))\, \beta. \tag{11}$$

Setting $\mu = 0$ recovers $c = \sqrt{q}$ (Eq. 5) as the special case, as claimed in Section 2.2. This is the formula, computed live from each residue's own pre-dropout (μ, σ), used at every undamped correction site in this paper.

A.4 DAMPED VARIANT FOR RESIDUAL-BYPASS SITES

Some sites do not feed dropout's output directly into LayerNorm; instead a bypass term is added first, $y = h + z$, where h is untouched by this dropout call and $z_k = m_k v_k/q$ is the dropped-out contribution (e.g. ProteinMPNN-style $h + \mathrm{dropout}(v)$, or Genie3's `PairTransition` input,

which sums an undropped running pair representation with two independently-dropped triangle-update terms – Section 4.6). Let $x := h + v$ denote the (deterministic, no-dropout) combined signal, $\mu := \mathrm{mean}_k(x_k)$, $\sigma^2 := \mathrm{var}_k(x_k)$. Repeating the moment calculation with h_k held fixed (it carries no dropout randomness at this site):

$$\mathbb{E}\left[\frac{1}{d}\sum_k y_k^2\right] = \frac{1}{d}\sum_k \left[(h_k + v_k)^2 + \frac{p}{q}v_k^2\right] = (\sigma^2 + \mu^2) + \frac{p}{q}\overline{v^2}, \tag{12}$$

where $\overline{v^2} := \frac{1}{d}\sum_k v_k^2$ is the mean-square of the dropped-out term alone. As before $\mathbb{E}[\mathrm{mean}_k(y)^2] \approx \mu^2$, so

$$\mathbb{E}[V_y] \;\approx\; \sigma^2 + \frac{p}{q}\,\overline{v^2} \;=\; \sigma^2\left(1 + \frac{p}{q}\rho^2\right), \qquad \rho^2 \;:=\; \frac{\overline{v^2}}{\sigma^2}, \tag{13}$$

and following the same steps as above,

$$\mathbb{E}[\mathrm{norm}(y)_k] \;\approx\; \sqrt{\frac{q}{q + p\rho^2}} \cdot \mathrm{norm}(x)_k, \qquad c_{\mathrm{damped}} \;:=\; \sqrt{\frac{q}{q + p\rho^2}}. \tag{14}$$

ρ^2 is the ratio of the combined signal's mean-square that comes from the dropped-out term v relative to the full signal's variance; it is 0 when the bypass totally dominates ($v \to 0$, so $c_{\mathrm{damped}} \to 1$, no correction needed) and grows as v's contribution grows. Eq. 14 is the general case: setting $h = 0$ (no bypass, $v = x$, so $\overline{v^2} = \sigma^2 + \mu^2$ and $\rho^2 = 1 + (\mu/\sigma)^2$) gives $c_{\mathrm{damped}} = \sqrt{q/(q + p + p(\mu/\sigma)^2)} = \sqrt{q/(1 + p(\mu/\sigma)^2)} = c(\mu, \sigma)$, exactly recovering Eq. 10 – the undamped correction used everywhere else in this paper is the $h = 0$ special case of the same formula.

B Extension to other normalization schemes

The dropout-induced shared-denominator mechanism analyzed in Section 2 and Appendix A is not specific to LayerNorm. We derive the equivalent correction for two other widely used normalization schemes: RMSNorm (Zhang & Sennrich, 2019), which shares LayerNorm's per-example normalization but omits mean-centering, and BatchNorm (Ioffe & Szegedy, 2015), which normalizes using statistics computed across the batch (and, at eval time, a moving average accumulated during training) rather than per example. Neither is used by any model tested in this paper, so both corrections are theoretical extensions, not empirically validated here.

B.1 RMSNorm

RMSNorm normalizes by the root-mean-square rather than the standard deviation, with no mean subtraction and (conventionally) no shift parameter:

$$\mathrm{RMSNorm}(x)_k = \gamma_k \cdot \frac{x_k}{\mathrm{rms}(x)}, \qquad \mathrm{rms}(x) := \sqrt{\mathrm{mean}_k(x_k^2) + \epsilon}. \tag{15}$$

With $z_k = m_k x_k / q$ as in Section 2, RMSNorm's denominator uses the raw (uncentered) second moment, which was already computed exactly in Appendix A (the numerator of Eq. 8, before the $-\mu^2$ centering term is subtracted):

$$\mathbb{E}\big[\mathrm{mean}_k(z_k^2)\big] = \frac{\sigma^2 + \mu^2}{q} = \frac{S^2}{q}, \qquad S^2 := \mathrm{mean}_k(x_k^2). \tag{16}$$

Using the same concentration approximation as Section 2.2,

$$\mathbb{E}[\mathrm{RMSnorm}(z)_k] \;\approx\; \frac{x_k}{\sqrt{S^2/q}} \;=\; \sqrt{q} \cdot \mathrm{RMSnorm}(x)_k, \tag{17}$$

so $c_{\mathrm{RMS}} = \sqrt{q}$ – the same constant as the zero-mean LayerNorm case (Eq. 5), except here it is *exact for every* μ, not just $\mu \approx 0$. LayerNorm's μ-dependence (Eq. 10) comes entirely from subtracting a non-inflated $\mathbb{E}[M]^2 \approx \mu^2$ from the inflated raw second moment $(\sigma^2 + \mu^2)/q$; RMSNorm never performs that subtraction, so its correction collapses to the single global constant $\sqrt{q}$ regardless of μ. The corrected estimator is simply $\widehat{\mathrm{RMSNorm}}(x) = \sqrt{q} \cdot \mathrm{RMSNorm}(x)$.

B.2 BatchNorm

BatchNorm normalizes each channel using statistics computed across the batch dimension rather than across channels within one example, and at eval time replaces these with a fixed moving average accumulated during training:

$$\mathrm{BN}(x)_k = \gamma_k \cdot \frac{x_k - \mathrm{running_mean}_k}{\sqrt{\mathrm{running_var}_k + \epsilon}} + \beta_k. \tag{18}$$

Unlike LayerNorm, this is a *fixed affine map* at eval time (the normalizing statistics do not depend on the current input) so there is no per-example shared-denominator randomness to analyze. Instead, the question is whether $\mathrm{running_mean}_k, \mathrm{running_var}_k$, accumulated from dropout-corrupted training batches, correctly reflect the dropout-free statistics that a test-time (dropout-off) input actually has.

Let μ_k, σ_k^2 denote channel k's true population mean and variance over the data distribution (no dropout). During training, each batch element's pre-BN activation is itself dropped out, $z_k^{(b)} = m_k^{(b)} x_k^{(b)}/q$; repeating the moment calculation of Appendix A, but marginalizing over the data distribution (index b, across the batch) instead of over channels (index k), gives the same expression with the roles of μ, σ swapped from a per-example, cross-channel quantity to a per-channel, cross-example one:

$$\mathrm{running_mean}_k \to \mu_k, \qquad \mathrm{running_var}_k \to \frac{\sigma_k^2 + p\mu_k^2}{q}. \tag{19}$$

At test time, dropout is off, so BatchNorm's fixed affine map is applied directly to x_k (not to any dropped-out quantity), and the *correct*, uninflated denominator would be σ_k, not $\sqrt{\mathrm{running_var}_k}$. The ratio between the two is

$$c(\mu_k, \sigma_k) \;=\; \frac{\sigma_k}{\sqrt{\mathrm{running_var}_k}} \;=\; \sqrt{\frac{q\,\sigma_k^2}{\sigma_k^2 + p\mu_k^2}} \;=\; \sqrt{\frac{q}{1 + p(\mu_k/\sigma_k)^2}}, \tag{20}$$

exactly the same functional form as LayerNorm's general correction (Eq. 10), but with μ, σ now the per-channel statistics across the training population rather than the cross-channel statistics of a single example. This matches (once translated to our notation) the variance-shift ratio $\Delta(q) = \mathrm{Var}_{\mathrm{test}}(X)/\mathrm{Var}_{\mathrm{train}}(X)$ derived independently by Li et al. (2018) for Dropout-directly-before-BatchNorm ($\Delta(q) = c(\mu, \sigma)^2$): their result establishes that this variance mismatch degrades accuracy in convolutional networks, but does not give a closed-form correction; $c(\mu_k, \sigma_k)$ above is that correction.

Because $\mathrm{running_mean}_k = \mu_k$ exactly (dropout is unbiased), Eq. 19 can be inverted to express the correction purely in terms of BatchNorm's own two buffers, with no separate estimate of μ_k, σ_k required:

$$c_k \;=\; \sqrt{q - p \cdot \frac{\mathrm{running_mean}_k^2}{\mathrm{running_var}_k}}, \tag{21}$$

and the same β-preserving correction as Eq. 5 applies with c_k in place of $\sqrt{q}$. Unlike the LayerNorm/RMSNorm corrections, which need a live hook on every forward pass's pre-dropout activation (since μ, σ vary per example), Eq. 21 is a *one-time* transform of a trained model's frozen buffers – c_k can be computed once, after training, and folded directly into $\mathrm{running_var}_k$ (or γ_k), at zero added inference cost.

C Further pLDDT calibration

ABB3-style structure models feed the final block's corrected output directly into a second LayerNorm. The second LayerNorm removes the rescaling, which means the pLDDT head receives a biased representation. This site is structurally distinct from every correction site used elsewhere in this paper: the randomness it inherits is not a fresh elementwise Bernoulli mask, but the residual variance of $z_1 = \mathrm{LN}_1(\mathrm{dropout}(x))$ left over *after* matching its mean via $\hat{c}_1$ – i.e. a second nonlinear (shared-denominator) operation stacked on top of an already-corrected quantity.

C.1 AN EFFECTIVE CORRECTION CAN BE DERIVED AND VALIDATED

Applying the moment-calculation technique of Section A.4 to this compound case (deriving $\mathrm{Var}(\mathrm{norm}(z_1)_k)$ from first principles rather than assuming the raw-dropout template applies) gives a damped-shaped correction $c_2 = \sqrt{q/(q+p\rho^2)}$ with $\rho^2 = c_1^2 \cdot S/(\sigma_x^2 \cdot \mathrm{Var}_k(\hat{c}_1))$, $S := \mathrm{mean}_k((\mathrm{weight}_1)_k^2 x_k^2)$ – computable live from quantities already available at this site (the pre-dropout x, LN_1's own weight, and c_1). Two independent checks confirm this derivation is correct: (1) isolating *only* the last transition block's dropout (every other stochastic site held at eval, so x is exactly the fixed quantity the derivation assumes) and comparing against a 100-sample Monte Carlo estimate of the true $\mathbb{E}[\mathrm{LN}_2(z_1)]$, the ρ^2-corrected estimate cuts the mean gap to ground truth roughly in half versus a mean-only (c_1-only) correction (0.255 vs. 0.507, n=2273 residues); (2) a sweep over a multiplier k on p inside the c_2 formula confirms k=1 (the as-derived formula) minimizes this gap, ruling out a missing constant factor.

C.2 THE CORRECT ESTIMATE IS POORLY CALIBRATED

Despite (1)-(2), applying this correction – or the model's own real p=0.1, or an empirically curve-fit constant p – to `pLDDT_head.layer_norm` *degrades* pLDDT calibration (mean absolute error between predicted pLDDT and the true lDDT-Cα of that same predicted structure), consistently on both the 34-structure test subset and an independent random 100-structure sample of ABB3's own train split (ruling out an out-of-distribution explanation):

Correction	MAE (test)	MAE (train, 100)
Uncorrected (status quo)	3.315	3.679
Model's real p=0.1 (naive reuse)	3.412 (-2.9%)	3.763 (-2.3%)
Empirically fit constant p=0.157	3.481 (-5.0%)	3.822 (-3.9%)
Derived ρ^2 (per-residue, live)	3.396 (-2.4%)	3.751 (-2.0%)

The ranking (derived ρ^2 least harmful, flat empirical constant most harmful) tracks how theoretically grounded each variant is, so this isn't a broken derivation – it's a correct estimate of the wrong target. Sweeping a free scalar c_2 directly against calibration (rather than against the true $\mathbb{E}[\mathrm{LN}_2(z_1)]$) finds a real, reproducible optimum at $c_2 \approx 1.75$ – i.e. *amplifying* away from bias_2, the opposite direction from every theoretically-motivated correction above – giving a genuine $\sim 7\%$ MAE improvement, identically on both test ($3.315 \to 3.079$) and train ($3.679 \to 3.422$).

Correcting the structure module measurably improves accuracy but mean predicted pLDDT is essentially flat. pLDDT's *ranking* of residues by expected accuracy is not broken. Spearman correlation with true Cα error is actually strongest at c_2=1 (-0.425) and the derived ρ^2 correction (-0.431), both better than uncorrected (-0.265), but its *absolute scale* is stuck calibrated to the noisier, pre-correction accuracy regime pLDDT was trained against, since dropout was always on during training. The empirically-optimal c_2=1.75 partially repairs this absolute mismatch (mean pLDDT rises to 95.78, appropriately higher for the now-better structures) at a real cost to discrimination (Spearman drops to -0.384). A coarse single-scalar rescale can only shift the whole distribution, not fix calibration and sharpness simultaneously the way retraining `pLDDT_head` against corrected-model outputs presumably would.

D ADDITIONAL ROBUSTNESS CHECKS

D.1 FULL ANTIBODY RESULTS

Table 5: Uncorrected vs. corrected RMSDs for the paired antibody models on the 34-structure test subset. CDR-H3 RMSDs are computed by aligning to heavy framework residues. Ibex is reported both as a single ensemble member (member 0 only, no ensembling) and as the full 8-member deep ensemble (closest model to mean). ESMFold and OpenFold are run using the AlphaFold "gap trick" with no recycling. *We do not have access to ESMFold's or Ibex's exact training details, so it is possible they were finetuned with no dropout. †ABB2's (ABodyBuilder2) structure module (public `ImmuneBuilder` package) instantiates its dropout+LayerNorm sites with `dropout=0.0` architecturally, never overridden at any training stage – the correction is an exact no-op (uncorrected and corrected outputs are bit-identical). As with every other model in this table, no OpenMM refinement post-processing is applied (raw structure-module output only); ABB2's own published accuracy assumes that refinement step, so the rows above are not directly comparable to that figure. Whole-Fv is full-backbone (N/Cα/C/O), same convention as CDR-H3 and every other row. Reported both as a single ensemble member (`antibody_model_1`) and as the full 4-member ensemble under `ImmuneBuilder`'s own blind (no-ground-truth) selection rule – align every member's Cα trace onto a common frame, then pick the member closest to the cross-model consensus mean.

Model	Condition	Whole-Fv	Impr.	CDR-H3	Impr.
ABB2 (1 member)†	–	1.110	–	2.968	–
ABB2 (4-member ensemble)†	–	1.095	–	2.862	–
ABB3	Raw	1.176	–	2.861	–
	+DLC	1.019	+13.3%	2.741	+4.2%
FlashABB	Raw	1.156	–	2.695	–
	+DLC	1.029	+10.9%	2.656	+1.5%
Ibex (1 member)*	Raw	1.054	–	2.846	–
	+DLC	1.014	+3.8%	2.791	+2.0%
Ibex (8-member ensemble)*	Raw	1.005	–	2.675	–
	+DLC	1.001	+0.5%	2.712	−1.4%
ESMFold*	Raw	1.423	–	3.232	–
	+DLC	1.420	+0.2%	3.224	+0.2%
OpenFold	Raw	1.420	–	3.691	–
	+DLC	1.417	+0.2%	3.674	+0.5%

D.2 IBEX ON A RANDOM 100-STRUCTURE TRAINING-SET SUBSAMPLE

As a check on whether the negative ensemble CDR-H3 result of Section 4.3 is a genuine correction/ensembling interaction or a small-sample artifact of the 34-structure test subset, this repeats the same comparison on an independent, much larger sample: 100 structures drawn at random (seed 42) from Ibex's own published training set (`docs/ibex_train.csv` in the Ibex source repository), restricted to the 8026/8395 structures already overlapping our own training split (so ground-truth atom positions and AHO-scheme CDR-H3 boundaries, the latter recomputed per structure via ANARCI, did not need to be rebuilt from scratch) (Dunbar & Deane, 2016). Framework-aligned CDR-H3 and whole-Fv RMSD, same convention as Table 5.

D.3 ESMFOLD AND OPENFOLD WITH THEIR OWN DEFAULT RECYCLING

Table 1 uses no-recycling for ESMFold and OpenFold, for a consistent single-forward-pass comparison across every model in that table. Both models also support recycling by default (3 iterations); this table repeats the same nanobody-100 comparison with recycling enabled, same full-backbone/CDR-H3 convention throughout.

Table 6: Ibex correction impact on a random 100-structure subsample of Ibex's own training set (not the 34-structure held-out test subset used elsewhere in this paper), member 0 and the 8-member medoid ensemble. *We do not have access to Ibex's exact training/finetuning details (e.g. whether dropout was disabled during finetuning, as in ImmuneBuilder's convention), so we cannot rule out that this affects how the correction interacts with this model specifically.

Model	Condition	Whole-Fv	Impr.	CDR-H3	Impr.
Ibex (1 member)*	Raw	0.9070	–	1.479	–
	Both sites	0.8800	+3.0%	1.419	+4.1%
	Both sites + plm_in	0.8793	+3.1%	1.411	+4.6%
Ibex (8-member ensemble)*	Raw	0.8926	–	1.431	–
	Both sites	0.8786	+1.6%	1.408	+1.6%
	Both sites + plm_in	0.8760	+1.9%	1.388	+3.0%

Table 7: ESMFold and OpenFold on the nanobody-100 set with 3 recycling iterations (their own default), rather than the no-recycling convention used in Table 1. *We do not have access to ESMFold's exact training/finetuning details, so it is possible it was finetuned with no dropout.

Model	Condition	Whole-domain	Impr.	CDR-H3	Impr.
ESMFold (3 recycles)*	Raw	1.835	–	4.186	–
	+DLC	1.827	+0.4%	4.160	+0.6%
OpenFold (3 recycles)	Raw	1.721	–	3.654	–
	+DLC	1.716	+0.3%	3.639	+0.4%

D.4 Statistical significance of the ESMFold/OpenFold improvement

Table 1's ESMFold and OpenFold improvements are modest in absolute terms ($+0.5\%$, $+0.8\%$) compared to the antibody-specific models, so we checked whether they are statistically distinguishable from noise using the same paired-t convention as Table 4 ($|t| \gtrsim 1.96 \Rightarrow p < 0.05$), on the per-structure corrected−uncorrected difference over the same 100 nanobody structures.

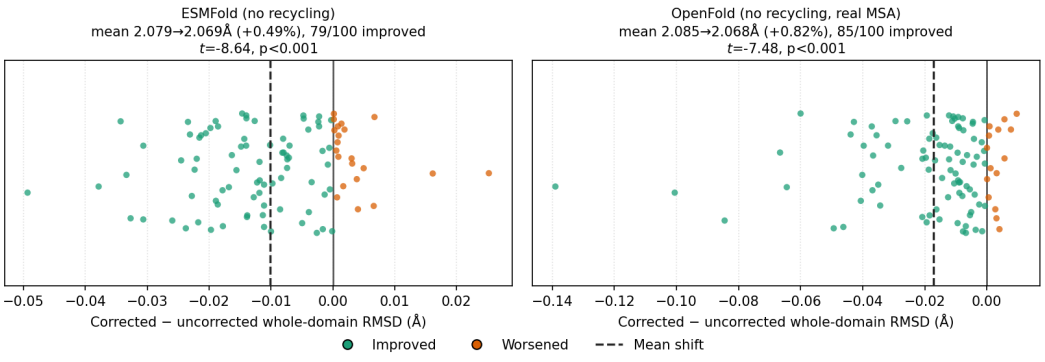


Figure 6: Per-structure whole-domain RMSD shift (corrected−uncorrected) for ESMFold and OpenFold, no-recycling, on the same 100 nanobody structures as Table 1. Each point is one structure; green/orange indicate whether that structure's RMSD improved or worsened under correction. Both shifts are significant despite their small mean magnitude ($t = -8.64$, $p < 0.001$ for ESMFold; $t = -7.48$, $p < 0.001$ for OpenFold) – the correction reliably nudges the large majority of structures (79/100 and 85/100) in the same direction, it is just a small nudge per structure for these two general-purpose, non-antibody-specific models.

D.5 CALIBRATION: SWEEPING THE ASSUMED q

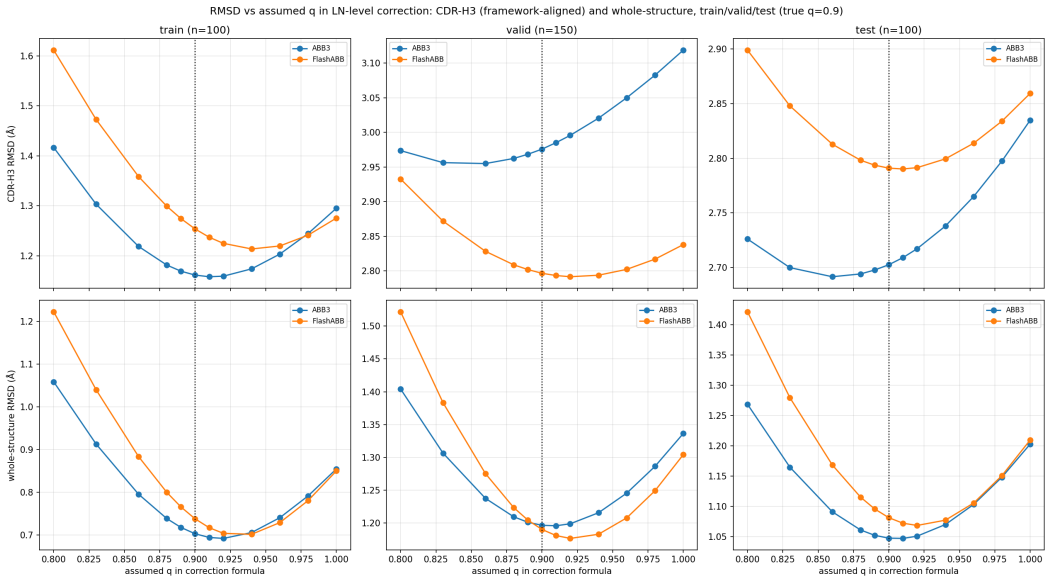


Figure 7: RMSD as a function of the assumed keep-probability q used inside the correction formula (Eq. 5/general form), for CDR-H3 (framework-aligned) and whole-structure RMSD, on train/valid/test splits, ABB3 and FlashABB. The true configured $q = 0.9$ is marked.

Throughout this work, we correct using the reported dropout rates for each model (typically 0.1). In Figure 7, we show RMSD as a function of the assumed keep rate, q, for ABB3 and FlashABB. In all cases, $q = 0.9$ is better than the uncorrected version $q = 1$. However, the optimal values did not always sit at exactly $q = 0.9$. It possible that this is simply a noise artifact driven by large outliers favouring slight (over/under)shoot predictions. Additionally, although we show that the correction fixes the overshoot problem almost exactly (Figure 3), it may be that downstream layers are sensitive to changes in variance induced by our correction.

D.6 GENIE3 NOISE-PREDICTION DECOMPOSITION

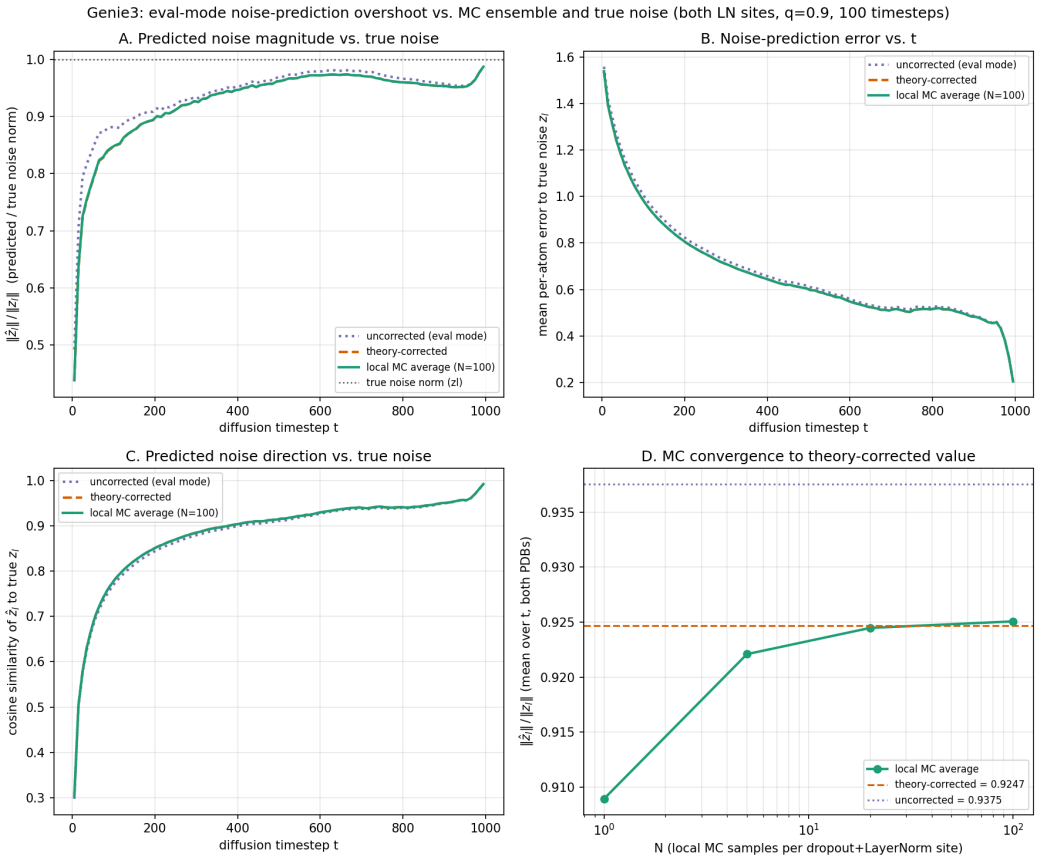


Figure 8: Genie3 noise-prediction decomposition vs. diffusion timestep t (all 26 LN sites corrected – `structure_net` and `pair_transform_net`, $q = 0.9$, 100 evenly spaced t values, 2 monomer structures $\times$ 12 noise draws per t). **A**: predicted/true noise-norm ratio – uncorrected overshoots the $N{=}100$ MC ceiling at 97/100 t values; all modes undershoot the true-noise reference line ($y = 1$). **B**: per-atom error to the true noise vector z_l, decomposing Table 3's loss numbers. **C**: cosine similarity of predicted to true noise direction – nearly identical across modes, showing the bias is a magnitude effect only. **D**: local MC average (mean over all t and both structures) converging onto the theory-corrected value as N grows from 1 to 100, well below the uncorrected reference.