

INPUT PERTURBATION AS AUGMENTATION FOR PRE-TRAINING DISTILLATION

Jack Cai

ABSTRACT

Knowledge distillation trains a student language model to match a teacher’s predictions on the training text. We ask whether the student also benefits from the teacher’s predictions on text that has been deliberately perturbed. For an early part of pretraining we perturb the inputs and train the student to match the teacher on the perturbed text, and we spend the rest of the same token budget on ordinary distillation. Teacher and student usually have embeddings of different sizes, so we perturb the one-hot token vectors, which each model then maps through its own embedding table. We test three perturbations: additive token noise, blending two documents, and blending neighbouring tokens. When pretraining from scratch, the method improves on standard distillation at both scales we tried, by 0.025 nats at 135M parameters and by 0.020 nats at 0.5B (three seeds). At 135M this is 1.45 times the gain that distillation itself provides over next-token training. The gain depends on the teacher seeing the perturbed text: without a teacher it largely disappears, and perturbing only the student’s input, as pretraining augmentation usually does, recovers about a third of it, while softer targets and a weaker early teacher do not reproduce it. When 262M unique tokens are repeated for 16 epochs, the gain persists against a baseline with tuned weight decay. The perturbed students also quantize at least as well with GPTQ and AWQ, and at 0.5B they gain about one point of average zero-shot accuracy over seven tasks.

1 INTRODUCTION

Distillation from a larger model is now a standard way to train small language models (Hinton et al., 2015; Peng et al., 2025). The student learns to match the teacher’s next-token distribution, but only on the training text itself. The teacher can make predictions on any input, and how it responds to text that differs slightly from the training data is information the student never receives. This paper asks whether giving the student some of that information during pretraining helps.

The usual tool for this kind of question is data augmentation. In language-model pretraining, augmentation typically corrupts the student’s input, by masking or replacing tokens or by adding noise to embeddings, and keeps the original next token as the target (Xu et al., 2026; Chen et al., 2026; Ni et al., 2025; Jain et al., 2023). A teacher makes a different kind of augmentation possible. It can predict on the corrupted input itself, so the target matches what the student actually sees. Training on teacher-labelled augmented inputs is established in task-specific distillation of language models (Jiao et al., 2020; Rashid et al., 2021) and in vision (Beyer et al., 2022; Wang et al., 2022). It has not been studied for distillation during pretraining from scratch, and it was not obvious that it would help there, since the student sees each token only once and the teacher’s predictions on clean text are already a rich target. We find that it does help, and that the teacher’s side is what matters. Perturbing only the teacher’s input recovers about 90% of the gain, while perturbing only the student’s input, the usual form of augmentation, recovers about a third.

Perturbing the input of a language model raises two practical problems. Tokens are discrete, so there is no small continuous change to make to them. And teacher and student embed tokens in spaces of different width, so noise added to one model’s embeddings has no counterpart in the other’s. We avoid both problems by perturbing the one-hot input vector in the shared vocabulary, which each model passes through its own embedding matrix. This gives a single perturbation that applies to models of any width, and we give both models the same draw. That choice turns out not to be

essential: independent draws work as well, and so does Gaussian noise added to each model’s own embeddings (Section 5).

Our contributions:

- Distillation on perturbed inputs for language models of different width, with three perturbation families and a two-stage training schedule (Section 2).
- Controlled pretraining experiments at two scales, every comparison paired on identical data (Section 3). At 135M the best variant gains 0.025 nats over standard distillation, and at 0.5B it gains 0.020 nats over three seeds (Section 4).
- How the gain depends on the perturbation size and the length of the perturbed stage, and why the noise should be sized to how strongly it changes the teacher’s predictions (Section 4).
- Ablations locating the gain (Section 5). It requires the teacher, comes mostly from perturbing the teacher’s input, and does not depend on sharing the draw or on the space in which the noise is added. Softened targets and a weaker teacher in the first stage do not reproduce it.
- Data-constrained pretraining (Section 6): with 16 epochs over 262M unique tokens, the gain persists on top of a tuned weight decay, and is about 2.6 times larger with the teacher than without.

2 METHOD

Shared one-hot perturbations (Fig. 1a). Let $o_t \in \mathbb{R}^{|V|}$ be the one-hot vector of token t . A perturbation replaces o_t by a sparse weighted combination $\tilde{o}_t = \sum_j w_j o_{v_j}$ of a few vocabulary items. Each model computes its input embedding as $E^\top \tilde{o}_t$ with its own embedding matrix E . Teacher and student receive the same (v_j, w_j) , so they see the same perturbation whatever their embedding width. The loss is the forward KL divergence from the teacher’s next-token distribution to the student’s at every position (temperature 1).

Perturbation families.

- **Noise.** $\tilde{o}_t = o_t + \sigma u_t$, where u_t places independent Gaussian weights $\mathcal{N}(0, 1/k)$ on $k=8$ tokens drawn uniformly from the vocabulary (with replacement, so the true token can occasionally be among them), resampled at every position; the weights are not normalized. σ sets the size of the perturbation relative to the true token.
- **Parallel blend.** Two independent documents a, b are mixed position by position, $\tilde{o}_t = \lambda o_{a_t} + (1 - \lambda) o_{b_t}$.
- **Adjacent blend.** Consecutive token pairs of one document are merged into one position, $\tilde{o}_t = \lambda o_{x_{2t}} + (1 - \lambda) o_{x_{2t+1}}$. Sequences are drawn twice as long, so inputs keep their full length.

Both blends consume two raw tokens per model position, so during the perturbed stage they read more unique text than the baseline; all token budgets are sized so that no token is seen twice. We test whether the extra text matters in Section 5.

Two-stage schedule. Training consists of a perturbed stage covering a fixed fraction of the run, followed by ordinary distillation on clean inputs for the remainder (Fig. 1b). The learning rate follows a warmup–stable–decay schedule (Hu et al., 2024): 10% warmup, a constant plateau spanning the stage boundary, and a 10% final decay, which therefore always takes place on clean inputs. The baseline is the identical run without the perturbed stage.

3 EXPERIMENTAL SETUP

Models and data. *135M scale:* a SmolLM2-135M architecture trained from random initialization and distilled from SmolLM2-1.7B (Allal et al., 2025), for 8000 steps of 512×1024 tokens (4.2B

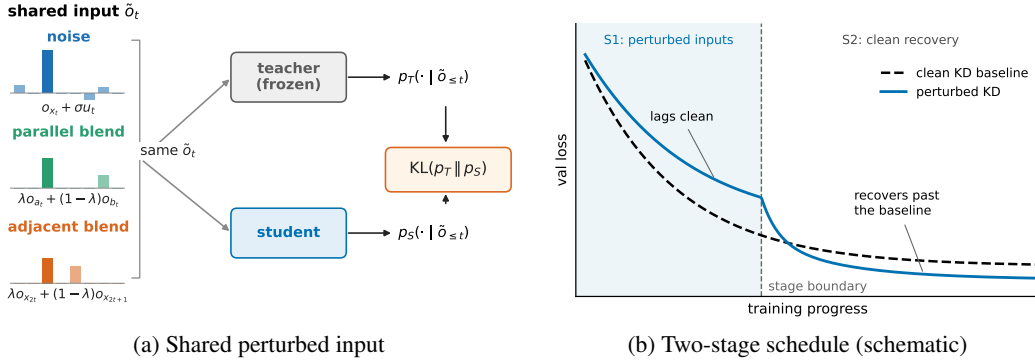


Figure 1: **Method.** (a) A perturbation replaces each token’s one-hot vector with a sparse weighted combination of vocabulary items. The left side shows the three families on a toy vocabulary, with the position’s own token as the dark bar. Teacher and student receive the same vector and map it through their own embedding tables, so the perturbation is the same even though their widths differ. The student is trained to match the frozen teacher’s next-token distribution. (b) While the inputs are perturbed, the student’s loss on clean validation text lags behind standard distillation. Once the perturbation is switched off, the loss recovers quickly, overtakes the baseline and finishes below it.

tokens) on a 58.6B-token FineWeb-Edu corpus (Penedo et al., 2024). *0.5B scale:* a Qwen2.5-0.5B architecture trained from random initialization and distilled from Qwen2.5-1.5B (Qwen Team, 2025), for 19073 steps of 256×2048 tokens (10B tokens) on a 65B-token FineWeb-Edu corpus. No data is repeated at either scale, except in the data-constrained experiments of Section 6.

Paired comparisons. All runs at a scale read the same data in the same order and start from the same initialization, and differ only in the training objective. As a check, runs from the same seed agree on validation loss at the first evaluation to within 0.004 nats. The next-token-prediction reference uses the same data stream.

Evaluation. At 0.5B every final model is scored on the full held-out validation split (2441 non-overlapping windows of 2048 tokens, identical for every run). We report the difference to standard distillation paired per window, with its standard error. At 135M we report the validation loss from the training run’s evaluation pass; differences below about 0.003 nats are within its resolution. Most configurations are trained once. The main 135M ablations, the data-constrained comparison and the best 0.5B configuration are repeated with two or three initialization seeds, always on the same data order.

4 RESULTS

All three perturbations improve on standard distillation at 135M (Fig. 2a–b). For reference, distillation itself improves on next-token prediction by 0.017 nats (3.137 against 3.154). Adjacent blending is the strongest variant. At $\lambda=0.8$ it lowers the loss by a further 0.0250 nats, 1.45 times the gain from distillation itself. Noise reaches -0.0193 at $\sigma=0.4$, and parallel blending -0.0177 at $\lambda=0.7$. Each family has a clear optimum, and going past it hurts. Noise improves up to $\sigma=0.4$ and is harmful by $\sigma=1.2$ ($+0.033$). Gaussian noise on the embeddings behaves the same way, with its best value at $\sigma=0.8$ in its own units (-0.021 , mean of two seeds). Parallel blending becomes harmful at $\lambda=0.6$ and 0.5 ($+0.021$ and $+0.037$).

Length of the perturbed stage (Fig. 2c, f). For both blends at 135M, the best perturbed stage covers 30% of training. A 15% stage gives less of the gain, and with a 70% stage both blends end up worse than next-token prediction. Noise is more forgiving. It does best with a 50% stage and still helps at 75% (-0.007). Blended inputs are much further from natural text than small noise, which is presumably why less training on them is useful. The 0.5B runs show the same ordering. Adjacent blending is again best with a 30% stage (-0.0204 ± 0.0003 , against -0.0188 at 15% and -0.0117

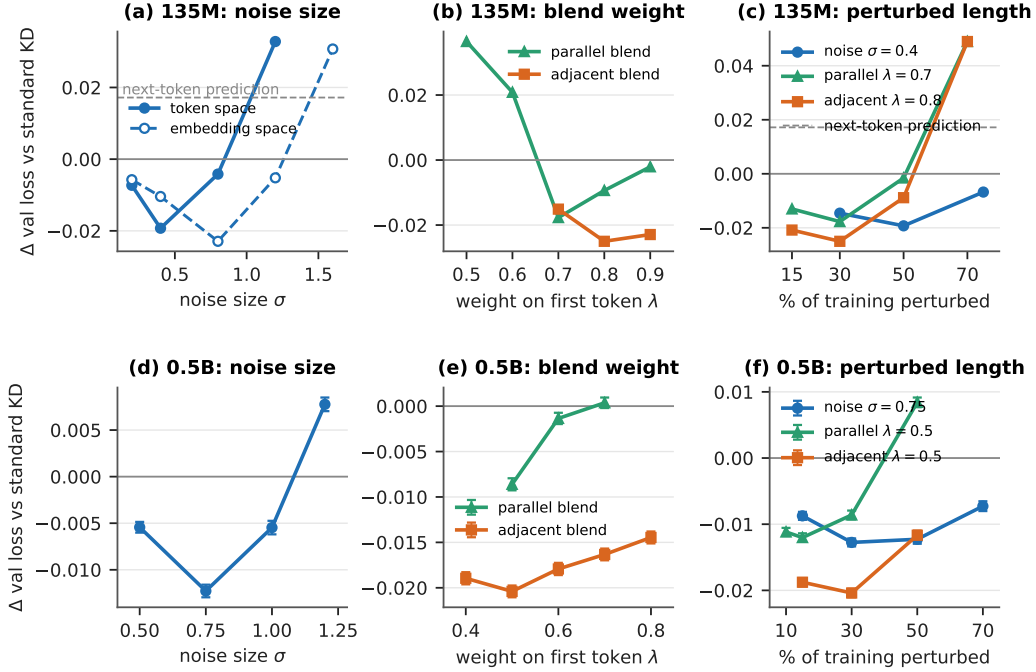


Figure 2: **Change in final validation loss against standard distillation** (below zero is better). (a–c) 135M student with the SmolLM2-1.7B teacher, 4.2B tokens. The dashed line is next-token prediction without a teacher. (a) Noise size, with a 50% perturbed stage. (b) Blend weight for both blends, with a 30% perturbed stage. The open markers in (a) are Gaussian noise added to each model’s input embedding; their σ is relative to the embedding’s root-mean-square value, so it is not in the same units as token noise. (c) Length of the perturbed stage at each family’s best size. (d–f) 0.5B student with the Qwen2.5-1.5B teacher, 10B tokens, scored on the full validation split. Error bars are ± 2 paired standard errors. (d) Noise size and (e) blend weight, with the same perturbed-stage lengths as in (a, b). (f) Length of the perturbed stage at each family’s best size.

at 50%), and noise gains about the same with a 30% or a 50% stage. Parallel blending at $\lambda=0.5$, the heaviest perturbation we use, does best with only 15%, and a 50% stage makes it harmful.

The gains hold at 0.5B (Fig. 2d–e). With the Qwen2.5-1.5B teacher, adjacent blending is again the strongest variant. Its best setting is an even blend of the two tokens ($\lambda=0.5$), which improves on standard distillation by 0.0204 ± 0.0003 nats. We repeated this run and its baseline with two more initialization seeds and obtained 0.0212 and 0.0187 . As the blend becomes less even the gain shrinks steadily, to 0.0145 at $\lambda=0.8$. Noise peaks at $\sigma=0.75$ (-0.0123 ± 0.0003) and gains less than half as much at $\sigma=0.5$ or 1.0 . Parallel blending also does best with an even blend (-0.0086 ± 0.0003), and its gain is gone by $\lambda=0.7$. Because the two documents play symmetric roles, $\lambda=0.5$ is the heaviest blend this family allows. At 135M, with a different teacher, parallel blending peaked at $\lambda=0.7$ and heavier blends hurt. The best blend weight, like the best noise size, depends on the teacher.

Noise should be sized to the teacher. At 0.5B we chose $\sigma=0.75$ before training, by measuring how much the noise raises the frozen teacher’s own loss ($+0.6$ nats at $\sigma=0.7$ and $+1.0$ at 0.8). Much larger noise, $\sigma=1.2$, raises the teacher’s loss by 3.5 nats, and training with it is harmful ($+0.0078 \pm 0.0004$). Smaller noise, $\sigma=0.5$, barely changes the teacher’s predictions ($+0.07$ nats at $\sigma=0.4$) and gives less than half the gain. The noise has to be large enough to change what the teacher predicts, but not so large that its predictions stop being informative.

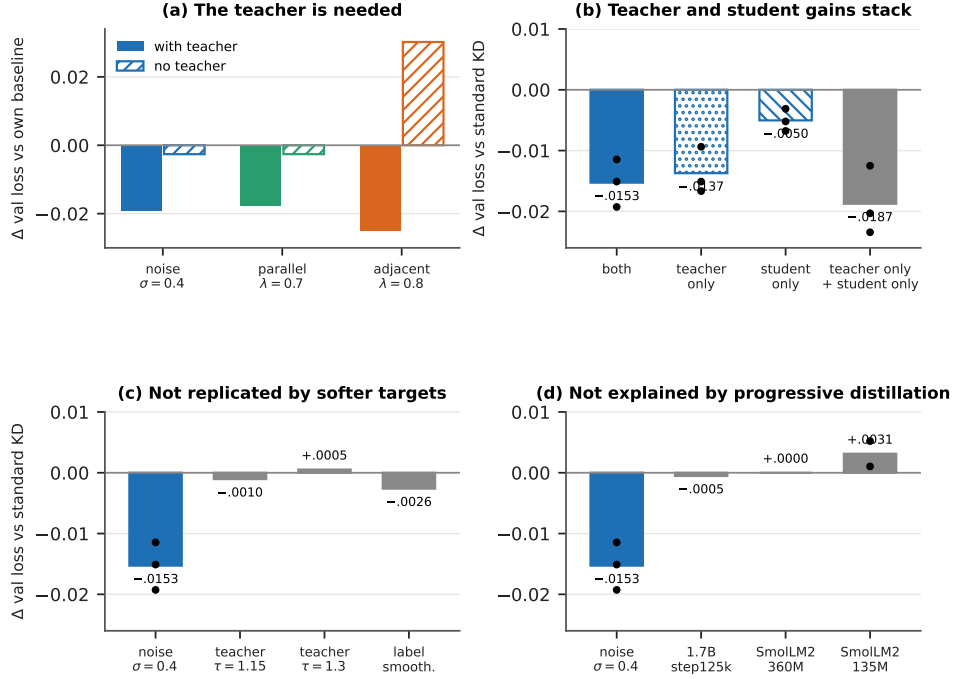


Figure 3: **Ablations** (135M, SmolLM2-1.7B teacher; difference in final validation loss, lower is better). Bars show the mean over initialization seeds and dots the individual seeds, each compared with standard distillation from the same seed. (a) Each family at its best setting, with and without the teacher, each compared with its own unperturbed baseline (standard distillation or next-token prediction). (b) Token noise given to both models, to the teacher only, or to the student only. The grey bar is the sum of the two one-sided gains. (c) Clean inputs, with the teacher’s target softened instead, either by a temperature τ ($\tau=1.15$ matches the entropy of the teacher’s prediction on $\sigma=0.4$ inputs) or by label smoothing to the same entropy. (d) Clean inputs for the first half of training, distilled from a weaker model (an early checkpoint of the teacher, or a smaller model) instead of the teacher.

5 ABLATIONS

We test four alternative explanations of the main result (Fig. 3). All ablations are at 135M and, unless noted, use token noise with $\sigma=0.4$ for the first half of training, the best noise setting. Its gain is -0.0193 for the seed shown in Fig. 2 and -0.0153 averaged over three seeds. Where we have several initialization seeds, bars show the mean and dots the individual seeds, and each seed is compared with standard distillation from the same seed.

Without a teacher (Fig. 3a). Running the same noise or parallel-blend schedule with ordinary next-token targets and no teacher improves on next-token prediction by only 0.0026 nats, 13–15% of the gain with a teacher on the same seed. For adjacent blending the contrast is starker: without a teacher, training on the blended inputs ends 0.030 nats behind plain next-token prediction, although the same inputs give the largest gain of all when a teacher labels them.

Teacher side and student side (Fig. 3b). Next we perturb only one model’s input. When only the teacher’s input is perturbed, the student sees clean text but is trained toward the teacher’s prediction on a perturbed version of it. This recovers almost all of the gain (-0.0137 , against -0.0153 with both perturbed; means over three seeds). When only the student’s input is perturbed, with the teacher’s prediction on the clean text as the target, the gain is much smaller (-0.0050), not much more than the same noise gives with no teacher at all (-0.0026). This student-only arm is the usual form of data augmentation for language models, which perturbs the input and keeps the target (Xu et al., 2026; Chen et al., 2026), and it obtains about a third of the gain. The two one-sided gains

sum to -0.0188 , a little more than the gain with both sides perturbed, so the two effects combine roughly additively.

Softer targets (Fig. 3c). A perturbed input makes the teacher less certain (the entropy of its prediction rises from 1.97 to 2.46 nats at $\sigma=0.4$), so the gain might come from a flatter target alone. It does not. Softening the teacher’s prediction on clean inputs to the same entropy with a temperature gives -0.0010 , and a stronger temperature ($\tau=1.3$) or label smoothing to the same entropy gains at most 0.0026 ($+0.0005$ and -0.0026). Applying the usual distillation temperature to both models, which reweights the loss without changing its optimum, is slightly harmful ($+0.0026$ at $\tau=1.15$, $+0.0078$ at $\tau=2$).

A weaker early teacher (Fig. 3d). A perturbed teacher is also a worse predictor, and distilling first from a weaker or partly trained teacher can act as a curriculum (Panigrahi et al., 2025). To test this, we distil the first half of training on clean inputs from a weaker model instead. None of the weaker teachers helps. SmoLM2-135M, whose loss is close to that of the 1.7B teacher on $\sigma=0.4$ inputs, gives $+0.0031$ (two seeds), SmoLM2-360M gives 0.0000, and the earliest published checkpoint of the 1.7B teacher gives -0.0005 .

Design choices that do not matter. Sharing the draw between teacher and student is not necessary, since independent draws give the same gain (Appendix C). Perturbing in token space is not necessary either, because Gaussian noise added to each model’s input embedding reaches a similar gain at its own best size (Fig. 2a).

6 DATA-CONSTRAINED PRETRAINING

When unique data are limited, training repeats them for several epochs, and regularization becomes important: properly tuned weight decay changes conclusions (Kim et al., 2026; Xu et al., 2026), and input corruption with clean targets helps next-token training (Xu et al., 2026; Chen et al., 2026; Ni et al., 2025). We test whether the gain of distilling on perturbed inputs survives this setting and a tuned baseline. At 135M, with the same 4.2B-token budget, training repeats the first 262M tokens of the corpus for 16 epochs. The tokens are cut into non-overlapping 1024-token windows, reshuffled each epoch with the same shuffle for every run. We sweep AdamW weight decay over 0.1, 0.3, 1, 3 and 10 for standard distillation (10 diverges) and use token noise $\sigma=0.4$ for the first 50% of training (Fig. 4).

Distillation already absorbs most of the cost of repetition. At the default weight decay, 16 epochs raise the final loss of next-token prediction by 0.18 nats but that of standard distillation by only 0.065. Tuning weight decay matters for both: the best value, 1.0, lowers the loss of standard distillation by 0.033 and of next-token prediction by 0.11.

With tuned weight decay. At weight decay 1.0, distilling on perturbed inputs improves on standard distillation by 0.023 and 0.020 nats for the two seeds, while standard distillation itself varies by only 0.001 between them. Noise does not shift the best weight decay. Without a teacher the same noise gains 0.008, so the teacher makes the gain about 2.6 times larger. At the default weight decay the gain from noise looks larger (0.032), but much of that is regularization that tuning the weight decay also provides. On the tuned baseline the gain is about the same as without any repetition (0.019, at the default weight decay). With milder repetition, 4 epochs over 1.05B unique tokens, repetition costs standard distillation only 0.006 and the gain is 0.023 at the default weight decay.

7 QUANTIZATION AND DOWNSTREAM EVALUATION

We evaluate the final 0.5B students, standard distillation and each family’s best setting, after post-training weight quantization (Table 1) and on zero-shot benchmarks (Table 2). Quantization damage is the increase in validation loss on 512 fixed validation windows, paired across models; all linear layers except the output head are quantized to int4 or int3 with groups of 128 weights. We compare round-to-nearest (RTN, symmetric absmax scale per group) with GPTQ (Frantar et al., 2023)

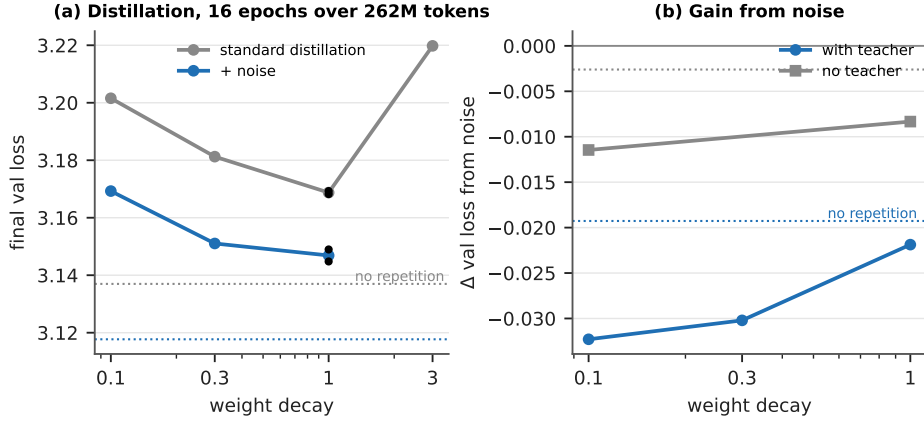


Figure 4: **Data-constrained pretraining** at 135M: 16 epochs over 262M unique tokens, same 4.2B-token budget. (a) Final validation loss of standard distillation with and without noise (token noise $\sigma=0.4$ for the first 50% of training) as a function of AdamW weight decay. Black dots are two initialization seeds at weight decay 1, and dotted lines show the same 4.2B-token budget without repeated data (weight decay 0.1). Weight decay 10 diverges. (b) The gain from the same noise with and without a teacher. Dotted lines again show the gain without repeated data. Exact values in Table 4 (Appendix B).

Table 1: **Increase in validation loss from post-training weight quantization** at 0.5B (nats; 512 fixed validation windows, paired across models). RTN: round-to-nearest; GPTQ and AWQ as implemented in llm-compressor. All linear layers except the output head are quantized; activations stay in 16 bits.

	full precision	int4, group 128			int3, group 128		
		RTN	GPTQ	AWQ	RTN	GPTQ	AWQ
standard KD	2.6782	0.205	0.028	0.019	6.226	0.160	0.099
noise $\sigma=0.75$	2.6642	0.317	0.027	0.019	5.623	0.117	0.097
parallel $\lambda=0.5$	2.6660	0.161	0.024	0.020	1.480	0.123	0.099
adjacent $\lambda=0.5$	2.6579	0.113	0.024	0.019	1.203	0.120	0.098

(symmetric) and AWQ (Lin et al., 2024) (asymmetric) as implemented in llm-compressor 0.14, both calibrated on 256 windows of 1024 tokens from each model’s own training data.

The gains survive quantization. With GPTQ at int4, every perturbed student keeps its advantage over standard distillation quantized the same way: -0.0244 ± 0.0011 nats for adjacent blending (-0.0203 before quantization, on the same 512 windows), -0.0166 for parallel blending and -0.0154 for noise. With AWQ at int4 the advantages are essentially unchanged (-0.0203 , -0.0118 and -0.0145). So with either method the perturbed students quantize at least as well as the standard one. Under plain round-to-nearest the results are mixed. At int4 the noise-trained student loses more than standard distillation (0.317 against 0.205 nats), while the blended students lose less.

Round-to-nearest and weight kurtosis. At int3 under round-to-nearest the blended students are far more robust. Int3 raises the loss of standard distillation by 6.2 nats, but that of adjacent and parallel blending by only 1.2 and 1.5. The difference holds at matched full-precision loss, comparing checkpoints from the middle of training (int3 damage 3.1 nats for standard distillation and 0.55 for adjacent blending), and it tracks the weight distribution. The mean excess kurtosis of the linear-layer weights is 0.52 for standard distillation and 0.54 for noise, but 0.14 for both blends. Parallel blending at $\lambda=0.7$, which gives no gain at 0.5B (Section 4), keeps the standard student’s kurtosis (0.52) and its RTN damage. Methods that compensate for quantization error remove most of the difference. With GPTQ the blended students’ int3 damage is 25% lower than standard distillation’s (0.12 against 0.16 nats), similar to noise, and with AWQ all students lose the same (0.10). The

Table 2: **Zero-shot accuracy** (%; lm-evaluation-harness; normalized accuracy for HellaSwag, ARC, PIQA and SciQ, accuracy for Winogrande and LAMBADA). Each family at its best setting for that scale. Δ Avg.: difference from standard distillation in the average over tasks, $\pm$ paired bootstrap standard error over examples. Best value per scale in bold.

	HellaSwag	ARC-E	ARC-C	PIQA	Wino.	LAMBADA	SciQ	Avg.	Δ Avg.
<i>0.5B</i>									
standard KD	45.0	57.9	29.0	68.0	52.7	37.5	77.3	52.5	–
noise $\sigma=0.75$	45.7	56.8	30.5	68.8	52.8	39.1	81.3	53.6	$+1.1 \pm 0.4$
parallel $\lambda=0.5$	45.5	58.1	30.3	68.3	54.1	36.7	77.5	52.9	$+0.5 \pm 0.3$
adjacent $\lambda=0.5$	45.9	58.0	30.6	69.7	53.4	36.3	80.9	53.5	$+1.1 \pm 0.3$
<i>135M</i>									
standard KD	30.1	44.4	25.2	60.1	52.3	21.3	64.0	42.5	–
noise $\sigma=0.4$	30.4	45.4	24.5	61.3	50.2	22.6	65.1	42.8	$+0.3 \pm 0.3$
parallel $\lambda=0.7$	30.5	46.1	25.1	60.3	50.0	22.1	62.6	42.4	-0.1 ± 0.4
adjacent $\lambda=0.8$	30.4	46.8	25.3	60.9	50.8	22.3	65.1	43.1	$+0.6 \pm 0.3$

activation statistics we checked, the outlier ratio of the inputs to each linear layer and the fraction of very large residual-stream activations, do not differ between the students.

Downstream accuracy. Averaged over seven zero-shot tasks (HellaSwag, ARC-Easy, ARC-Challenge, PIQA, Winogrande, LAMBADA, SciQ; lm-evaluation-harness), adjacent blending and noise both raise accuracy at 0.5B by about one point over standard distillation, from 52.5% to 53.5% and 53.6% (± 0.3 and ± 0.4 ; Table 2). Parallel blending gains 0.5 ± 0.3 . The largest gains are on SciQ ($+3.6$ and $+4.0$ points) and HellaSwag ($+0.9$ and $+0.8$), while both blends are about a point lower on LAMBADA. At 135M the differences in the average are within noise ($+0.6 \pm 0.4$ for adjacent blending).

Caveats. Apart from the best adjacent-blending setting, which we trained with three seeds (gains between 0.0187 and 0.0212 nats), each 0.5B configuration was trained once, and its error bars reflect evaluation sampling, not variation between training runs. At 135M, the gain from noise differs by up to 0.008 nats between initialization seeds. This is smaller than the main gains but comparable to several of the ablation differences, so differences of a few thousandths of a nat at 135M should be read as indicative. Individual downstream tasks are mostly within two standard errors of zero; only the averages are clearly separated. Perturbation sizes and stage lengths were selected on the same validation split on which the gains are reported, so the best settings carry some selection bias. Quantization was measured by validation loss rather than downstream accuracy, and only for weights. All students are small (at most 0.5B parameters) and trained for at most 10B tokens.

8 RELATED WORK

Distillation in pretraining. Distillation into small pretrained language models is well established (Hinton et al., 2015; Peng et al., 2025). Progressive distillation from intermediate teacher checkpoints can induce an implicit curriculum (Panigrahi et al., 2025); a weaker first-stage teacher does not reproduce our gain (Section 5). Work on language-model distillation has mostly changed the divergence or the sampling distribution, for example with on-policy or reverse-KL objectives (Gu et al., 2024; Agarwal et al., 2024). MiniPLM (Gu et al., 2025) uses the teacher to reweight the pre-training data, and Busbridge et al. (2025) give scaling laws for distillation. We keep the objective fixed and change the inputs on which it is evaluated.

Perturbations and derivative matching. Matching a teacher’s derivatives, or its responses to perturbations, goes back to Jacobian and Sobolev training (Srinivas & Fleuret, 2018; Czarnecki et al., 2017). In vision, Beyer et al. (2022) show that teacher and student should see the same augmented view. Task-specific distillation of language models already trains on teacher-labelled augmented text, e.g. with word replacement (Jiao et al., 2020) or adversarially masked text (Rashid et al., 2021), and Liu et al. (2021) inject noise into both teacher and student in self-distillation for

summarization. In vision, teacher-labelled augmentation is analysed by Wang et al. (2022) and Stanton et al. (2021); Noisy Student (Xie et al., 2020) instead noises the student and labels the clean input. We study the pretraining setting and separate the teacher-side and student-side contributions. Unlike Jacobian matching, our students do not learn the teacher’s local response to the perturbation.

Data-constrained pretraining and input augmentation. Repeating data for a few epochs is nearly as good as fresh data, with returns diminishing beyond about 16 (Muennighoff et al., 2023; Chai & Xiong, 2026). Under repetition, dropout and much larger weight decay help (Xue et al., 2023; Kim et al., 2026), and so does corrupting the input while keeping the clean target: an auxiliary loss on masked inputs (Xu et al., 2026), random token replacement (Chen et al., 2026), or masking, which partly explains why diffusion language models tolerate more repetition (Ni et al., 2025; Prabhudesai et al., 2025). Kim et al. (2026) also distill an ensemble into a smaller student, by training on the ensemble’s generated text. These methods perturb the student’s input only; in distillation, we find that most of the benefit comes from letting the teacher predict on the perturbed input.

Mixing and merging inputs. Mixup-style interpolation (Zhang et al., 2018) has been applied to language-model distillation in embedding space (Liang et al., 2021). Merging neighbouring tokens is used to save computation in vision transformers (Bolya et al., 2023) and in language-model pre-training (Peng, 2026), and embedding noise improves instruction fine-tuning (Jain et al., 2023). Our blends use related constructions as shared perturbations for distillation, and our ablation shows that their benefit depends on the teacher.

9 CONCLUSION

Distilling on perturbed inputs is a small change to standard distillation. Only the inputs of an early stage of training change, and the objective, the data and the token budget stay the same. It improved the student at both scales we tested, by 0.025 nats at 135M and 0.020 nats at 0.5B with the best variant, adjacent-token blending. Two things mattered most in our experiments: keeping the perturbed stage short, and sizing the perturbation so that the teacher can still predict well on it. The improvement comes from the teacher’s predictions on the perturbed inputs. This is why it mostly disappears without a teacher, and why the usual student-side augmentation recovers only part of it. One thing we do not yet understand is why a single random draw of the teacher’s target works better than the average of several draws (Appendix C).

REFERENCES

- Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2306.13649.
- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. SmoLLM2: When smol goes big — data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.
- Lucas Beyer, Xiaohua Zhai, Amélie Royer, Larisa Markeeva, Rohan Anil, and Alexander Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations (ICLR)*, 2023.
- Dan Busbridge, Amitis Shidani, Floris Weers, Jason Ramapuram, Etai Littwin, and Russ Webb. Distillation scaling laws. In *International Conference on Machine Learning*, 2025.
- Yekun Chai and Haoyi Xiong. What is a repeated token worth? the scaling geometry of multi-epoch pretraining. *arXiv preprint arXiv:2610.05591*, 2026.

- Chen et al. Demystifying training-time augmentation for data-constrained language model pretraining. *arXiv preprint arXiv:2606.16246*, 2026.
- Wojciech M. Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev training for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- Elias Frantar, Saleh Ashkboos, Torsten Hoefer, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023.
- Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations (ICLR)*, 2024.
- Yuxian Gu, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. MiniPLM: Knowledge distillation for pre-training language models. In *International Conference on Learning Representations*, 2025.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. MiniCPM: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.
- Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. NEFTune: Noisy embeddings improve instruction finetuning. *arXiv preprint arXiv:2310.05914*, 2023.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.
- Konwoo Kim, Suhas Kotha, Percy Liang, and Tatsunori Hashimoto. Pre-training under infinite compute. In *International Conference on Learning Representations*, 2026.
- Kevin J. Liang, Weituo Hao, Dinghan Shen, Yufan Zhou, Weizhu Chen, Changyou Chen, and Lawrence Carin. MixKD: Towards efficient distillation of large-scale language models. In *International Conference on Learning Representations (ICLR)*, 2021. *arXiv:2011.00593*.
- Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems (MLSys)*, 2024.
- Yang Liu, Sheng Shen, and Mirella Lapata. Noisy self-knowledge distillation for text summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 692–703, 2021.
- Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*, 2023.
- Jinjie Ni, Qian Liu, Longxu Dou, Chao Du, Zili Wang, Hang Yan, Tianyu Pang, and Michael Qizhe Shieh. Diffusion language models are super data learners. *arXiv preprint arXiv:2511.03276*, 2025.
- Abhishek Panigrahi, Bingbin Liu, Sadhika Malladi, Andrej Risteski, and Surbhi Goel. Progressive distillation induces an implicit curriculum. In *International Conference on Learning Representations*, 2025.

- Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. arXiv:2406.17557.
- Bowen Peng. Efficient pre-training with token superposition. *arXiv preprint arXiv:2605.06546*, 2026.
- Hao Peng, Xin Lv, Yushi Bai, Zijun Yao, Jiajie Zhang, Lei Hou, and Juanzi Li. Pre-training distillation for large language models: A design space exploration. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025. arXiv:2410.16215.
- Mihir Prabhudesai, Mengning Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. Diffusion beats autoregressive in data-constrained settings. *arXiv preprint arXiv:2507.15857*, 2025.
- Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.
- Ahmad Rashid, Vasileios Lioutas, and Mehdi Rezagholizadeh. MATE-KD: Masked adversarial TExt, a companion to knowledge distillation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 2021.
- Suraj Srinivas and François Fleuret. Knowledge transfer with Jacobian matching. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018. arXiv:1803.00443.
- Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A. Alemi, and Andrew Gordon Wilson. Does knowledge distillation really work? In *Advances in Neural Information Processing Systems*, 2021.
- Huan Wang, Suhas Lohit, Michael Jones, and Yun Fu. What makes a “good” data augmentation in knowledge distillation – a statistical perspective. In *Advances in Neural Information Processing Systems*, 2022.
- Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. Self-training with noisy student improves ImageNet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- Zhiwei Xu, Shihao Wu, Hanseul Cho, Wei Hu, and Yixin Wang. Data-constrained language model pretraining: Improved regularization and scaling laws. *arXiv preprint arXiv:2606.06888*, 2026.
- Fuzhao Xue, Yao Fu, Wangchunshu Zhou, Zangwei Zheng, and Yang You. To repeat or not to repeat: Insights from scaling LLM under token-crisis. In *Advances in Neural Information Processing Systems*, 2023.
- Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

A TRAINING DETAILS

B DATA-CONSTRAINED RESULTS

C MECHANISM PROBES

Shared or independent draws. Drawing the teacher’s and the student’s noise independently gives the same gain as sharing it (-0.0168 against -0.0153 , three seeds; Fig. 5). In line with this, the students do not learn how the teacher’s prediction responds to the perturbation. We measure each model’s response as the change in its next-token distribution under a small step along the noise direction. For every student we trained, this response is nearly uncorrelated with the teacher’s (correlation at most about 0.03), with or without noise in training. *One draw or the average of four.* Averaging the teacher’s predictions over four noise draws into one target, which has the same expected value as a single draw, gives a smaller gain than a single draw given to the teacher only (-0.0076 against -0.0122 on the same two seeds). This suggests that the randomness of the target helps, not only its expected value. The differences in this appendix are comparable to the spread between seeds, and we regard them as indicative.

Table 3: **Training configuration.** Both scales train the student from random initialization with a frozen teacher; the distillation loss is the forward KL divergence over the full vocabulary at temperature 1, with no next-token term.

	135M	0.5B
Student / teacher	SmolLM2-135M arch. / SmolLM2-1.7B	Qwen2.5-0.5B arch. / Qwen2.5-1.5B
Vocabulary (shared)	49,152	151,936
Data	FineWeb-Edu, 58.6B-token pool	FineWeb-Edu, 65B-token pool
Steps $\times$ batch $\times$ length	$8000 \times 512 \times 1024$ (4.2B tokens)	$19073 \times 256 \times 2048$ (10B tokens)
Optimizer	AdamW, $\beta=(0.9, 0.95)$, weight decay 0.1	same
Peak learning rate	6×10^{-4}	1.2×10^{-3}
Schedule	WSD: 10% warmup, 10% decay	same
Precision	bfloat16	bfloat16
Compute per run	≈ 20 H100-hours	≈ 90 H100-hours

Table 4: **Data-constrained pretraining** at 135M: final validation loss with 16 epochs over 262M unique tokens, by weight decay, against the same 4.2B-token budget without repetition (1 epoch). Two values: two initialization seeds. Noise: token noise $\sigma=0.4$ for the first 50% of training. Bold: the better of each objective with and without noise.

	1 epoch	16 epochs, weight decay			
	(wd 0.1)	0.1	0.3	1	3
Next-token prediction	3.1542	3.3365	—	3.2297	—
+ noise	3.1516	3.3250	—	3.2214	—
Standard distillation	3.1370	3.2016	3.1812	3.1682 / 3.1693	3.2198
+ noise	3.1177	3.1693	3.1510	3.1448 / 3.1490	—

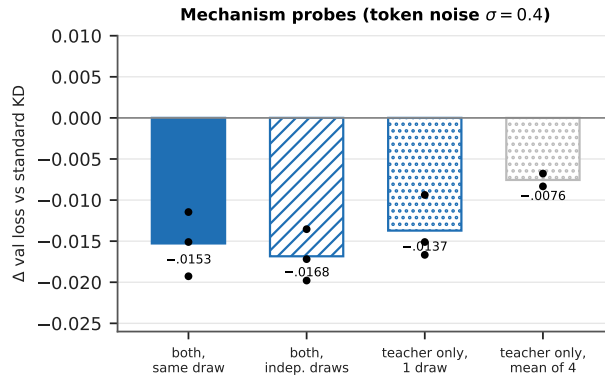


Figure 5: **Mechanism probes** (135M, token noise $\sigma=0.4$ for the first 50% of training; difference from standard distillation, mean over seeds with individual seeds as dots).