Outcome Prediction Quiz: Generalization

CS 312 · 40 minutes · 100 points

How to answer. Seven questions, 100 points. Use your Assignment 6 experiments to predict the outcomes. Submit only answer pages S1–S4, printed double-sided on two sheets.

For Q1–Q6, assignment evidence and reasoning can recover up to 75% of missed points. Q7 does not allow rationales. Candidate plots show possible outcomes. Lower validation loss is better.

Reference recipe. Use the A6 DNA setting: 30,000 training sequences of length 1,024 (about 30M tokens), with the same 1,000 validation sequences. One epoch is one pass over this fixed pool.

Default: dense d8, 32 epochs, peak LR 0.003**, WD** 3.2**.** AdamW, batch 64, microbatch 8, 75 warmup updates, then linear decay to zero. No dropout or stochastic depth. d8 has eight blocks, width 512, MLP width 1,792, and vocabulary 19.

Unless stated otherwise, keep these defaults and compare terminal validation NLL. Ensembles average prediction probabilities, with dropout disabled at evaluation. Exact protocols are in Appendix A.

Q1. Architecture

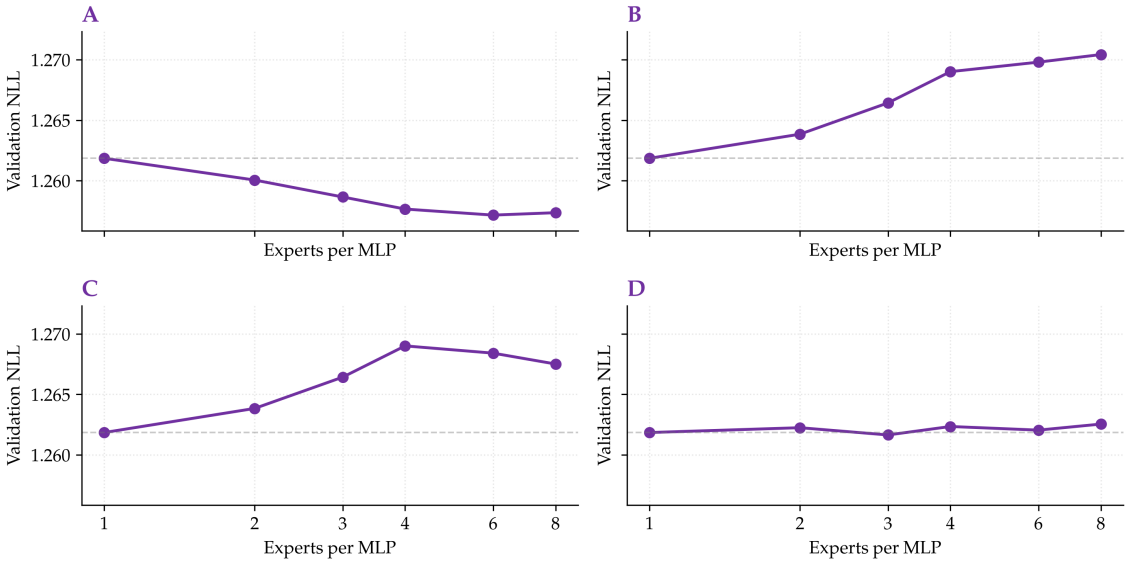
Difficulty: Easy

12 points

Code changes: Appendix A.1, page 9.

TLDR: Expert-count scaling at fixed hyperparameters.

(a) MoE expert count (6 points). Use the assignment's top-1, quantile-balanced MoE at the default recipe. Vary experts per MLP over $\{1, 2, 3, 4, 6, 8\}$, holding expert width and hyperparameters fixed; one expert is dense d8. Which plot matches mean terminal validation loss over five matched seeds?



Choice: A ☐ B ☐ C ☐ D ☐

TLDR: Attention Residuals at the default training recipe.

(b) Attention Residuals (6 points). Replace ordinary residual sums with learned softmax attention over earlier attention/MLP outputs and the embedding (Full Attention Residuals). At the default d8 recipe, does this give lower, similar, or higher validation loss than ordinary dense d8? Write $=$ for losses within 0.001 NLL (estimated run-to-run standard deviation), and $<$ or $>$ otherwise.

L_{AttnRes} ☐ L_{dense}

Give your choices and brief reasoning on the answer sheet.

Q2. Stochastic regularization

Difficulty: Medium

20 points

Write $<$ for a loss reduction greater than 0.001, $>$ for a loss increase greater than 0.001, and $=$ for a difference of at most 0.001. Compare the left model's loss with the right model's loss in each box.

Code changes: Appendix A.2, page 9.

TLDR: Dropout alone, model ensembles, and inference-mask ensembles.

(a) Attention dropout (12 points). Use probability 0.05, with no stochastic depth.

(i) Single model (4 points). Compare dropout with no dropout.

$L_{\text{dropout, K=1}}$ ☐ $L_{\text{no dropout, K=1}}$

(ii) Ten-model ensemble (4 points). Compare ensembles trained with and without dropout.

$L_{\text{dropout, K=10}}$ ☐ $L_{\text{no dropout, K=10}}$

(iii) Inference masks (4 points). For one dropout-trained model, compare standard inference (dropout off) with averaging prediction probabilities over five independent dropout masks (dropout on). Keep the weights fixed.

$L_{\text{five masks}}$ ☐ L_{standard}

TLDR: Stochastic-depth composition with dropout and ensembling.

(b) Stochastic depth (8 points). Stochastic depth (SD) randomly skips residual branches during training:

$$h \leftarrow h + \frac{m}{1-p_i} f(h), \qquad m = \begin{cases} 0 & \text{with probability } p_i, \\ 1 & \text{otherwise.} \end{cases}$$

Here f is an attention or MLP residual branch; rescaling preserves its expected contribution. For the six interior block applications, use $p_i = p_{\max} i/6$. Sample masks independently per sequence, branch, and application (including loop repeats), shared across tokens and channels. Keep the first and last blocks active. At evaluation use $h \leftarrow h + f(h)$, without masking or rescaling.

At $p_{\max} = 0.05$, SD improves dense d8 loss by about 0.0018. Does this benefit compose with:

(i) Attention dropout 0.05 **(4 points).**

$L_{\text{SD+dropout}}$ ☐ L_{dropout}

(ii) Five-model ensembling, without dropout (4 points).

$L_{\text{SD, K=5}}$ ☐ $L_{\text{no SD, K=5}}$

Give your choices and brief reasoning on the answer sheet.

Q3. Fine-tuning with few samples

Difficulty: Easy

12 points

Code changes: Appendix A.3, page 9.

Compare the A6 initializations: related DNA, DCLM text, IID nucleotide noise with BOS, and scratch. Fully fine-tune all parameters for 16 passes with default WD and linear decay. Average terminal validation loss over three matched seeds and target subsets.

TLDR: Learning-rate preference across initializations.

(a) Fine-tuning LR (6 points). At 3k samples, which initialization benefits from the smallest LR? Compare each initialization's best LR over $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ and select the one with the smallest loss-minimizing LR.

Related DNA ☐ Scratch ☐ Nucleotide noise ☐ DCLM ☐

TLDR: Initialization choice at two fine-tuning data budgets.

(b) Initialization (6 points). At LR 0.003, which initialization is best at each sample count? Each row is worth 3 points.

Samples	Scratch	Nucleotide noise	DCLM
30	☐	☐	☐
30,000	☐	☐	☐

Choices within 0.001 of the best mean earn full credit.

Give your choices and brief reasoning on the answer sheet.

Q4. Learning rate schedules

Difficulty: Medium

20 points

Code changes: Appendix A.4, page 10.

For (a–b), compare constant LR, 5%/10%/20%/50% warmup–stable–decay (WSD), and linear decay. The WSD percentage is the fraction of post-warmup updates spent linearly decaying to zero. Keep other defaults fixed; average terminal validation loss over three matched seeds.

Choose one schedule per row (2 points each). Any choice within 0.001 of the best mean earns full credit. Otherwise, award 1 point in (a) for an immediately neighboring schedule column in that row (no wrapping), and in (b) for the next-lowest loss after excluding all full-credit answers in that row.

TLDR: Schedule choice at a fixed learning rate.

(a) Default LR (10 points). Fix peak LR to 0.003. Which schedule is best at each horizon?

Epochs	Constant	5% WSD	10% WSD	20% WSD	50% WSD	Linear
16	☐	☐	☐	☐	☐	☐
32	☐	☐	☐	☐	☐	☐
64	☐	☐	☐	☐	☐	☐
128	☐	☐	☐	☐	☐	☐
256	☐	☐	☐	☐	☐	☐

TLDR: Schedule choice after learning-rate tuning.

(b) Tuned LR (10 points). For each schedule and horizon, select the LR with the lowest three-seed mean over

$$\{0.0001,\ 0.0003,\ 0.001,\ 0.003,\ 0.01\}.$$

Which schedule is best after tuning?

Epochs	Constant	5% WSD	10% WSD	20% WSD	50% WSD	Linear
16	☐	☐	☐	☐	☐	☐
32	☐	☐	☐	☐	☐	☐
64	☐	☐	☐	☐	☐	☐
128	☐	☐	☐	☐	☐	☐
256	☐	☐	☐	☐	☐	☐

Give your choices and brief reasoning on the answer sheet.

Q5. Optimization

Difficulty: Medium

12 points

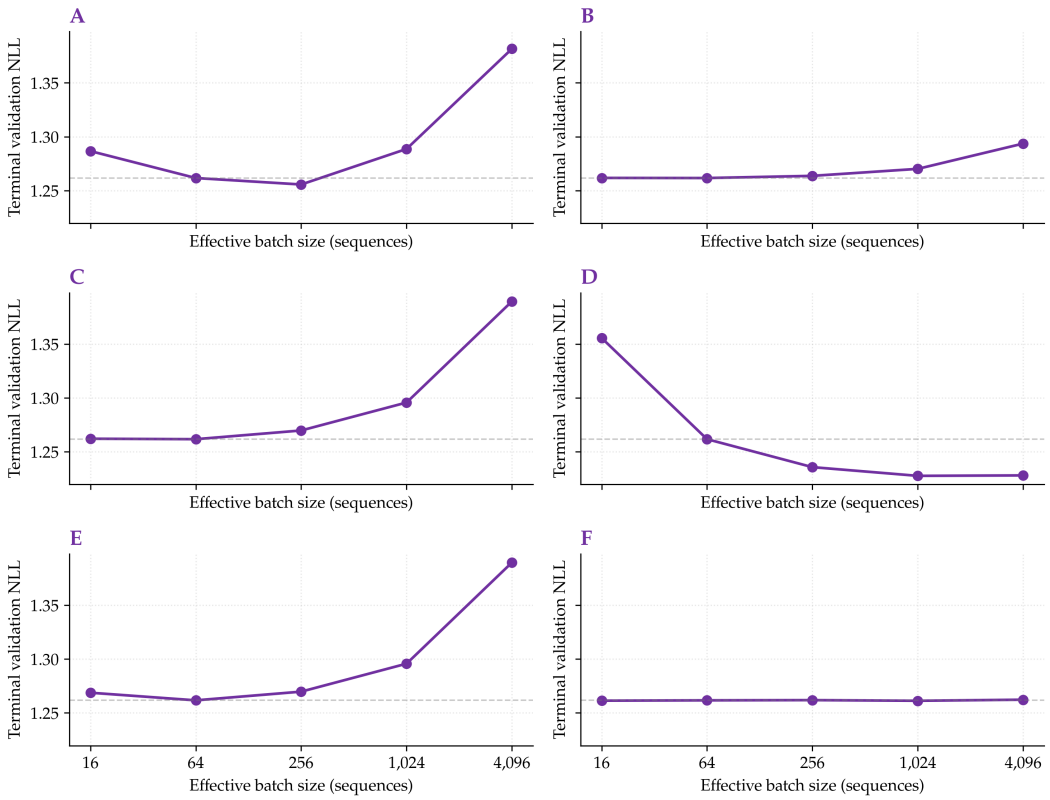
Code changes: Appendix A.5, page 10.

TLDR: Batch-size scaling under transferred hyperparameters.

(a) Batch size (6 points). Vary $B \in \{16, 64, 256, 1024, 4096\}$ under the default 32-epoch linear schedule, transferring both LR and WD:

$$\mathrm{LR}(B) = 0.003\sqrt{B/64}, \qquad \mathrm{WD}(B) = 3.2\sqrt{B/64}.$$

Use $\lceil 4800/B \rceil$ warmup updates. Which plot matches mean terminal validation loss across the original and seed-replicated runs? Panels share axes and the $B = 64$ anchor.



Choice: A ☐ B ☐ C ☐ D ☐ E ☐ F ☐

TLDR: Optimizer ranking after learning-rate tuning.

(b) Optimizers (6 points). Tune LR over $\{0.001, 0.003, 0.01, 0.03, 0.1\}$ for each optimizer. Then average terminal validation loss over three matched seeds and rank H = AdamH, M = Muon, S = SAM ($\rho = 0.05$). Keep the other defaults; AdamH uses no WD. Write letters and only $<$ or $>$, from lowest to highest mean loss. There is no equality option or tie rule.

☐ ☐ ☐ ☐ ☐

Give your choices and brief reasoning on the answer sheet.

Q6. EMA, sharpness, and generalization

Difficulty: Hard

12 points

Code changes: Appendix A.6, page 10.

Use WD 3.2 and three matched seeds. Compare constant LR, constant LR with parameter EMA ($\beta = 0.99$ per update), and linear decay. EMA changes only the evaluated weights.

TLDR: Sharpness under constant LR, EMA, and linear decay.

(a) Sharpness (6 points). Fix LR to 0.003. Rank mean A3 Adam-preconditioned sharpness at 32 epochs: C = constant, E = constant with EMA, L = linear. This is the largest eigenvalue of $D^{-1/2}HD^{-1/2}$, where H is the loss Hessian and $D = \sqrt{\hat{v}} + \varepsilon_{\text{Adam}}$ uses saved Adam second moments. Use the constant trajectory's moments for both C and E. The 0.001 loss rule does not apply to sharpness.

$$\square \quad < \quad \square \quad < \quad \square$$

TLDR: LR-tuned loss across recipes and horizons.

(b) Loss with LR tuning (6 points). Peak LR is tuned over $\{0.0001, 0.0003, 0.001, 0.003\}$ separately for each recipe and epoch count. Select the lowest and highest LR-tuned mean terminal validation loss among these six options (3 points each):

Choice	Recipe	Epochs
A	Constant	32
B	Constant	256
C	Constant with EMA	32
D	Constant with EMA	256
E	Linear	32
F	Linear	256

Any option within 0.001 of the corresponding extreme earns full credit. After these full-credit ties, the second-best or second-worst loss tier earns 1.5 points, respectively.

Best loss: A ☐ B ☐ C ☐ D ☐ E ☐ F ☐

Worst loss: A ☐ B ☐ C ☐ D ☐ E ☐ F ☐

Give your choices and brief reasoning on the answer sheet.

Q7. Rank the packages

Difficulty: Hard

12 points

Code changes: Appendix A.7, page 10.

TLDR: Ranking the six selected intervention packages.

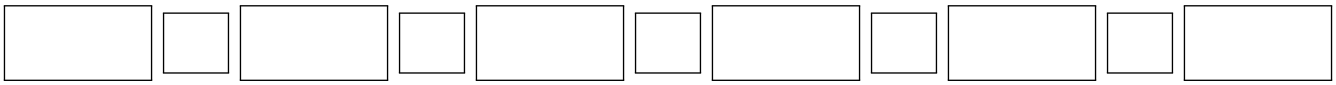
We construct one package from each question. Rank the interventions from lowest loss to highest loss.

This question does not allow any rationales, since they would be equivalent to previous parts.

Apply each package separately to the default recipe, using single-model terminal validation loss. Use all 30k target sequences; retain each question's stated tuning and horizon choices.

Package	Intervention
Q1	Best MoE expert count: the best choice from Q1(a).
Q2	Attention dropout with probability 0.05.
Q3	Related-DNA pretrained initialization, then full fine-tuning on all 30k target sequences for 16 passes with tuned LR.
Q4	Sweep LR, epoch count, and LR schedule: the best choice across Q4(b).
Q5	The best LR-tuned optimizer from Q5(b).
Q6	Sweep LR for constant, linear, and constant with EMA at 32 or 256 epochs: the best choice from Q6(b).

Write package numbers 1–6 in the large boxes and only $<$ or $>$ in the small boxes. For this problem, rank the numerical mean losses strictly; there is no equality option or 0.001 tie rule.



Appendix A: essential code changes

Base: A6 `7f0db01`. Selected hunks only; green adds code, red removes it.

Complete runnable code: `answer_key_code.zip`.

A.1 — Architecture

One expert is dense; retain quantile routing, combine weight Ep_{selected}, and router WD 0.1. Full AttnRes mixes earlier raw branches and the embedding using zero-initialized queries and RMS-normalized keys.

```
--- /dev/null
+++ b/experiments/a6_generalization/quiz_q1.py
@@ -0,0 +7,4 @@ expert-count sweep
+        for experts in (1, 2, 3, 4, 6, 8):
+            label = f'experts{experts}-s{seed}'
+            runs[label] = recipe(label, seed, experts=experts,
+                                 architecture='dense' if experts == 1 else 'moe')
--- /dev/null
+++ b/experiments/a6_generalization/quiz_attention.py
@@ -0,0 +76,6 @@ Full Attention Residuals
+        for i, block in enumerate(self.model.layers):
+            x = self.routers[2*i](sources)
+            sources.append(block.self_attn(block.input_layernorm(x), position_embeddings=rope))
+            x = self.routers[2*i+1](sources)
+            sources.append(block.mlp(block.post_attention_layernorm(x)))
+        return self.lm_head(self.model.norm(self.routers[-1](sources)))
```

A.2 — Stochastic regularization

Drop interior attention/MLP branches independently per sequence; keep boundary blocks active. Average probabilities before NLL. Use ten model members for dropout and five for SD. Inference masks enable attention dropout only.

```
--- /dev/null
+++ b/experiments/a6_generalization/quiz_support.py
@@ -0,0 +9,13 @@ install_sd
+def install_sd(model, probability):
+    """Independent per-sequence masks for each interior residual branch."""
+    def hook(p):
+        def apply(module, inputs, update):
+            if not model.training or p == 0:
+                return update
+            keep = torch.rand((len(update), 1, 1), device=update.device) >= p
+            return update * keep.to(update.dtype) / (1 - p)
+        return apply
+    middle = list(model.model.layers)[1:-1]
+    for i, block in enumerate(middle, 1):
+        for branch in (block.self_attn, block.mlp):
+            branch.register_forward_hook(hook(probability * i / len(middle)))
@@ -0,0 +57,4 @@ inference masks only
+            model.eval()
+            if masks:
+                for block in model.model.layers:
+                    block.self_attn.train()  # SD remains disabled: model.training=False.
@@ -0,0 +77,3 @@ ensemble_nll
+def ensemble_nll(bank):
+    """Leading axis = members or masks; average probabilities BEFORE taking log."""
+    return float(-np.log(np.asarray(bank, dtype=np.float64).mean(0)).mean())
```

A.3 — Full fine-tuning

Use nested prefixes of one shuffled pool per seed. Update every parameter; warmup is $\min(75, \text{updates} - 1)$. Select LR from matched-seed means.

```
--- a/experiments/a6_generalization/training.py
+++ b/experiments/a6_generalization/training.py
@@ -57 +58,6 @@
-    train_rows, val = data(c, root)
+    if c.subset_seed is None:
+        train_rows, val = data(c, root)
+    else:
+        pool, val = data(replace(c, ntrain=30000), root)
+        indices = np.random.default_rng(c.subset_seed).permutation(len(pool))[:c.ntrain]
+        train_rows = pool[indices]
```

A.4 — LR schedules

Fraction 0 is constant; 1 is linear. Other fractions specify the post-warmup cooldown. Constant horizons share one trajectory; others train separately.

```
--- /dev/null
+++ b/experiments/a6_generalization/quiz_support.py
@@ -0,0 +24,7 @@ lr_factor
+def lr_factor(step, total, warmup, fraction):
+    if warmup and step < warmup:
+        return step / warmup
+    if fraction == 0:
+        return 1.
+    cooldown = max(1, round((total - warmup) * fraction))
+    return min(1., max(0., (total - step) / cooldown))
```

A.5 — Batch size and optimizers

Use warmup $\lceil 4800/B \rceil$. SAM uses the same minibatch in both passes, radius 0.05. Muon auxiliary LR is $0.1\times$ matrix LR.

```
--- /dev/null
+++ b/experiments/a6_generalization/quiz_q5.py
@@ -0,0 +13,6 @@ common LR sweep
+    grids = {name:(.001, .003, .01, .03, .1) for name in ('adamh','muon','sam')}
+    for optimizer, lrs in grids.items():
+        for lr in lrs:
+            label = f'{optimizer}-lr{lr:g}-s1043'
+            runs[label] = recipe(label, 1043, optimizer=optimizer, lr=lr,
+                                 wd=0. if optimizer == 'adamh' else 3.2, rho=.05)
--- /dev/null
+++ b/experiments/a6_generalization/quiz_optim.py
@@ -0,0 +58,7 @@ Muon versus auxiliary AdamW
+        for name, p in model.named_parameters():
+            matrix = p.ndim == 2 and '.layers.' in name
+            mode = 'muon' if matrix else 'adam'
+            lr = config.lr if matrix else .1 * config.lr
+            wd = config.wd if id(p) in decayed else 0.
+            group = groups.setdefault((mode, lr, wd), dict(params=[], mode=mode, lr=lr, weight_decay=wd))
+            group['params'].append(p)
@@ -0,0 +66,4 @@ AdamH
+    if config.optimizer == 'adamh':
+        from dataclasses import replace
+        assert config.wd == 0
+        return original(model, replace(config, optimizer='hyperball'))
```

A.6 — EMA

Start EMA from zero and use $\beta = 0.99$. Restore raw weights after evaluation. For curvature, use the same FP32 64×1024 held-out probe and $D^{-1/2}HD^{-1/2}$; EMA uses the raw trajectory's Adam moments. Require convergence; omit WD and terminal-LR scaling from the probe.

```
--- a/experiments/a6_generalization/training.py
+++ b/experiments/a6_generalization/training.py
@@ -140,0 +147,4 @@ after optimizer.step()
+        if average:
+            with torch.no_grad():
+                for n, p in model.named_parameters():
+                    average[n].lerp_(p, 1 - c.ema_beta)
--- /dev/null
+++ b/experiments/a6_generalization/quiz_support.py
@@ -0,0 +38,3 @@ bias-corrected EMA evaluation
+            mass = -math.expm1(updates * math.log(beta))
+            for n, p in model.named_parameters():
+                p.copy_(average[n] / mass)
@@ -0,0 +103,2 @@ Adam preconditioner
+            vhat = state['exp_avg_sq'].float() / (1 - group['betas'][1] ** step)
+            diagonal[id(p)] = (vhat.sqrt() + group['eps']).rsqrt()
```

A.7 — Packages

No additional algorithm changes: select the Q1–Q6 recipes separately. The related-DNA package is the model after full fine-tuning on all 30k sequences for 16 passes with tuned LR. Use strict mean-loss order, with no rationales.

Name _____ **SUNet ID** _____

your Stanford login, e.g. `jdoe`; not the 8-digit ID number

Submit all four pages S1–S4. For Q1–Q6, cite the relevant assignment investigation and give observations and a rationale. Q7 allows no rationales.

Q1. Architecture — 12 points

(a) Expert-count plot

Choice: A ☐ B ☐ C ☐ D ☐

(b) L_{AttnRes} ☐ L_{dense}

Write $=$ for losses within 0.001 NLL; otherwise use $<$ or $>$.

Assignment problem cited	Observations from the experiment	Your rationale
(a)	(a)	(a)
(b)	(b)	(b)

Q2(a). Dropout and ensembling — 12 points

Write $<, =, >$; compare left loss with right loss at tolerance 0.001.

(a)(i) $L_{\text{dropout, K=1}}$ ☐ $L_{\text{no dropout, K=1}}$

(a)(ii) $L_{\text{dropout, K=10}}$ ☐ $L_{\text{no dropout, K=10}}$

(a)(iii) $L_{\text{five masks}}$ ☐ L_{standard}

Assignment problem cited	Observations from the experiment	Your rationale
(a)(i)	(a)(i)	(a)(i)
(a)(ii)	(a)(ii)	(a)(ii)
(a)(iii)	(a)(iii)	(a)(iii)

Name _____ **SUNet ID** _____

your Stanford login, e.g. `jdoe`; not the 8-digit ID number

Q2(b). Stochastic depth — 8 points

Write $<, =, >$; compare left loss with right loss at tolerance 0.001.

(b)(i) $L_{\text{SD+dropout}}$ ☐ L_{dropout}

(b)(ii) $L_{\text{SD, K=5}}$ ☐ $L_{\text{no SD, K=5}}$

Assignment problem cited	**Observations from the experiment**	**Your rationale**
(b)(i)	(b)(i)	(b)(i)
(b)(ii)	(b)(ii)	(b)(ii)

Q3. Fine-tuning with few samples — 12 points

(a) Initialization with the smallest loss-minimizing LR at 3k samples.

Related DNA ☐ Scratch ☐ Nucleotide noise ☐ DCLM ☐

(b) Best initialization at LR 0.003**; 16 passes.**

Samples	Scratch	Nucleotide noise	DCLM
30	☐	☐	☐
30,000	☐	☐	☐

Assignment problem cited	**Observations from the experiment**	**Your rationale**
(a)	(a)	(a)
(b)	(b)	(b)

Name _____ **SUNet ID** _____

your Stanford login, e.g. `jdoe`; not the 8-digit ID number

Q4. LR schedules 20 points

(a) Default LR 0.003

Epochs	C	5%	10%	20%	50%	L
16	☐	☐	☐	☐	☐	☐
32	☐	☐	☐	☐	☐	☐
64	☐	☐	☐	☐	☐	☐
128	☐	☐	☐	☐	☐	☐
256	☐	☐	☐	☐	☐	☐

(b) LR tuned

Epochs	C	5%	10%	20%	50%	L
16	☐	☐	☐	☐	☐	☐
32	☐	☐	☐	☐	☐	☐
64	☐	☐	☐	☐	☐	☐
128	☐	☐	☐	☐	☐	☐
256	☐	☐	☐	☐	☐	☐

Choose one schedule per row. C = constant; L = linear; percentages = WSD decay fractions.

Assignment problem cited	**Observations from the experiment**	**Your rationale**
(a)	(a)	(a)
(b)	(b)	(b)

Q5. Optimization 12 points

(a) Batch-size plot (6 points)

Choice: A ☐ B ☐ C ☐ D ☐ E ☐ F ☐

(b) LR-tuned optimizer loss (6 points).

H = AdamH, M = Muon, S = SAM ($\rho = 0.05$).

Lowest to highest three-seed mean loss; only $<$ or $>$. No equality option or tie rule.

	☐		☐	

Assignment problem cited	**Observations from the experiment**	**Your rationale**
(a)	(a)	(a)
(b)	(b)	(b)

Name _____ **SUNet ID** _____

your Stanford login, e.g. `jdoe`; not the 8-digit ID number

Q6(a). EMA and sharpness — 6 points

Lowest to highest mean Adam-preconditioned sharpness. Use C, E, L.

☐ < ☐ < ☐

Assignment problem cited	Observations from the experiment	Your rationale
(a)	(a)	(a)

Q6(b). EMA and loss — 6 points

LR tuned separately for each recipe and horizon over $\{0.0001, 0.0003, 0.001, 0.003\}$; $\beta = 0.99$ EMA.

Best loss: A☐ B☐ C☐ D☐ E☐ F☐

Worst loss: A☐ B☐ C☐ D☐ E☐ F☐

Assignment problem cited	Observations from the experiment	Your rationale
(b) Best loss	(b) Best loss	(b) Best loss
(b) Worst loss	(b) Worst loss	(b) Worst loss

Q7. Rank the packages — 12 points

Rank from lowest loss to highest loss. Write package numbers 1–6 and only $<$ or $>$. Rank numerical mean losses strictly; Q7 has no equality option or 0.001 tie rule.

Q1: Best MoE expert count: the best choice from Q1(a); Q2: attention dropout 0.05; Q3: related DNA followed by full fine-tuning; Q4: best tuned schedule/horizon; Q5: best LR-tuned optimizer; Q6: best tuned constant/linear/EMA recipe.

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

This question does not allow any rationales, since they would be equivalent to previous parts.