

Peer Review

Review of: "Transformers Can Navigate Mazes With Multi-Step Prediction"

Evgenii Arsentev¹

1. Independent researcher

1. What the work claims

The paper asks a narrow, well-posed question: if you keep the data, the parameter count, and the decoding procedure fixed, does replacing next-token prediction with MLM-U (the uniform-rate masking objective of reference [5], which predicts any subset of solution-path tokens from any other) change how well a small transformer finds shortest paths in grid mazes? Two maze families are used. DFS mazes are generated with the library of [25], serialized as edge lists with one token per cell, and have long, unique solution paths. A* mazes are taken from the deterministic dataset of Lehnert et al. [6], serialized as wall lists with separate x and y tokens, and have short paths with ties broken by a fixed heuristic. Accuracy is the exact match of the whole generated path against the reference path, on 2000 held-out mazes per setting (§4.4).

The headline results are these. On DFS mazes with 8M-parameter models trained on 100k mazes, MLM-U reaches 100 at 5x5 through 20x20 and 93.8 at 30x30, against 100, 45.2, 24.4, 20.6, and 18.8 for the autoregressive model (Table 1, rows Autoregressive and MLM-U, columns 5x5 to 30x30). On A* mazes, the 8M MLM-U model scores 98.5, 95.2, and 85.5 at 10x10, 20x20, and 30x30 (Table 2, row MLM-U 8M), which the author compares with numbers copied from [6], including a 175M next-token model trained with A* search traces at 98.5, 90.4, and 70.2 (Table 2, row "Next token 175M + A* trace supervision"). The abstract adds three secondary claims: MLM-U is 4x more sample-efficient, it converges about 2x faster in GPU hours on 5x5 mazes, and it benefits from scale (3M to 8M lifts 20x20 DFS accuracy from 85% to 100%, §5.3). A separate finding in §5.4 is that MLM-U with RoPE needs 32-bit positional encodings on large grids: 30x30 DFS accuracy moves from 40% to 93.8% when precision is raised from 16 to 32 bits.

I like this paper. The question is clean, the effect on DFS mazes is large, and the author reports things that cut against the method: the Table 6 result that MLM-U scores 0 full-path accuracy on 10x10 mazes

unless they are embedded in a 20x20 frame, the §5.1 caveat that next-token training crosses 40% faster on 10x10, and the Appendix C note that about half of the 20x20 A* failures are parse errors. The ordered-mask ablation in Appendix E.1 (73% per-token, under 4% full-path on 10x10 A*) is exactly the kind of control that makes a mechanism claim credible. My concerns are about arithmetic in the efficiency claims, one possible confound in the baseline, and the looseness of the A* comparison.

2. Whether the conclusion holds on the data

The main DFS result holds. I checked the size of the gap against sampling noise on the 2000-maze test set. For the smallest contrast that matters, 30x30 in Table 1, a normal-approximation 95% half-width at $p = 0.188$ is $1.96 \times \sqrt{0.188 \times 0.812 / 2000} = 1.96 \times 0.00874 = 1.7$ points, and at $p = 0.938$ it is $1.96 \times \sqrt{0.938 \times 0.062 / 2000} = 1.96 \times 0.00539 = 1.1$ points. A 75-point gap is not a test-set accident. What sampling noise cannot tell us about is training variance. As far as I can see, Tables 1 to 4 and Table 6 come from single training runs; only Figure 4 mentions three seeds. For the headline gap, this hardly matters. For the scaling claim (85% at 3M to 100% at 8M, §5.3 and Figure 6), and for the 16-bit versus 32-bit contrast of §5.4, it matters a great deal, because a single unlucky run of a small model on a hard objective can easily lose 15 points.

Possible confound: positional-encoding precision in the baseline. §5.4 shows that MLM-U on 30x30 DFS mazes goes from 40% to 93.8% when RoPE is computed in 32 rather than 16 bits, and the 93.8 is the number that appears in Table 1 (row MLM-U, column 30x30). So the MLM-U column was produced with a precision fix that was found by looking at MLM-U failures. I could not find in the text whether the decoder-only next-token baseline used RoPE at all, what positional scheme it used, or whether it was trained under AMP with 16-bit positions. If the baseline also uses RoPE under AMP, part of the degradation from 45.2 (10x10) to 18.8 (30x30) in Table 1 could be the same precision artifact rather than a property of the objective. The failure pattern described for next-token models in Appendix C ("frequent backtracks, traversals through walls") is not obviously the same as the "skipping some and duplicating others" pattern of §5.4, so I doubt this explains the whole gap. But it needs one rerun: the autoregressive baseline at 20x20 and 30x30 with 32-bit positions.

"Identical settings" is only partly true. The abstract says the models are "parameter-matched transformers ... under identical settings." On DFS mazes, the architectures differ (decoder-only for next token, encoder-decoder for MLM-U, §4.4), which the author justifies with Figure 8 at 10x10 only. That choice favors the baseline, and I have no objection to it. The hyperparameter treatment is asymmetric in

the other direction, though: next-token learning rate and weight decay were swept on 10x10 mazes for 500 epochs (Appendix B), and those values were then used at every grid size and for up to 3000 epochs, while MLM-U was run with a learning rate of 0.001 and no weight decay. The 10x10 selection is reasonable, but it does not tell us the next-token model was well tuned at 20x20 or 30x30, where the gap is widest. On A* mazes, there is no controlled comparison at all: every baseline row in Table 2 is "taken directly from [6]" (Table 2 caption), so training length, optimizer, precision, stopping rule, and evaluation code all differ in ways the paper does not document.

The A* claim is stated more strongly than Table 2 supports. The introduction says MLM-U "outperforms next token transformers trained with additional A* search trace supervision on complex mazes," and §5.2 says the 8M model "compares favorably with all models from [6]." At 20x20 and 30x30, that is true (95.2 vs 90.4, 85.5 vs 70.2, Table 2, rows MLM-U 8M and Next token 175M + A* trace). At 10x10, both are 98.5, a tie. The contrast with the plain 15M next-token model (13.3 vs 85.5 at 30x30) is the most informative one, since the size gap is small, but it is still a cross-paper comparison. I would soften the wording and say plainly that the A* baselines were not retrained.

The GPU-hours claim does not follow from the numbers in §5.1. This is the clearest arithmetic contradiction in the text. §5.1 gives 13.7 hours for next token and 17.7 hours for MLM-U on the same data for 2k epochs, and calls this an "additional 7% overhead." But $17.7 / 13.7 = 1.292$, which is a 29.2% overhead. The epoch speedup is given as 2.17x in the text and 2.03x in the Figure 4 caption. Dividing by the per-epoch cost ratio: $2.17 / 1.292 = 1.68$ and $2.03 / 1.292 = 1.57$. Neither is " $\sim 2x$," which is what §5.1 and the abstract claim. The right panel of Figure 4 may show different wall-clock numbers; as rendered on Qeios, I cannot read its values, so I can only say that the numbers written in the text give roughly 1.6x to 1.7x. It is also worth saying in the abstract that this efficiency result is for 5x5 mazes only, where both objectives reach 100%; on 10x10, the paper itself reports next token learning faster at first.

The 4x sample-efficiency figure is a 5x5 result. For 5x5 mazes, the ratio is well defined: $100k / 25k = 4$ (§5.1, Figure 3 left). For 10x10, the next-token model never reaches full accuracy, so there is no ratio, only a lower bound that could be anything. The abstract states "4x more sample efficient" without that qualifier. A smaller point: the data-efficiency runs fix the number of epochs at 2000 (§5.1), so a model trained on 25k mazes gets a quarter of the optimizer steps of one trained on 100k. That handicaps both objectives equally, but it means the figure measures samples and steps together.

Small numerical inconsistencies in the prose. §5.1 says MLM-U achieves "nearly 3x the performance of next token training on more complex 30x30 mazes"; from Table 1, the ratio is $93.8 / 18.8 = 4.99$, so the text

undersells the result. §5.1 also mentions a next-token peak of "46%" on 10x10, and §5.3 says "around 50%," against 45.2 in Table 1 (row Autoregressive, column 10x10). §5.3 says next token "yields 22% accuracy" on 20x20 against 20.6 in Table 1. These may come from different runs or checkpoints, which is itself a reason to report seeds, but as written, they look like the same quantity.

What the mechanism claim rests on. The title and abstract attribute the gain to "explicitly predicting multiple steps ahead (and backwards)." MLM-U changes several things at once: the masking pattern, the loss being restricted to path tokens (Appendix D.1: "the only tokens that can be masked are part of the solution path"), the unusual decoder that receives only copies of one learnable token plus RoPE (Appendix D.3), and the removal of teacher forcing. Appendix E.1 is a good start, since ordered right-side masking with the same architecture fails badly. I would still like to see at least one of: next-token training with the loss restricted to path tokens (I could not tell from §4.2 whether the baseline loss covers the maze description as well), a fixed-rate BERT-style mask, or the multi-token prediction heads of [4], which the paper cites as motivation but never runs. Without them, "multi-step prediction" is a plausible explanation rather than an established one.

Generalization. Table 6 deserves more weight in the discussion. MLM-U trained on 20x20 scores 2 per-token and 0 full-path on 10x10 mazes (row MLM-U), against 30 and 21 for next token (row Next Token), and only recovers to 100 and 100 when the small maze is pasted into a 20x20 frame. The paper also says both models score zero on larger mazes. So MLM-U seems to learn a solver tied closely to absolute position and grid layout. That does not undercut the in-distribution result, but it limits how far "long-term planning" can be read into it, and the Discussion should say so.

3. Reproducibility

This is a small-model, from-scratch study with a public repository (github.com/facebookresearch/maze_navigation_MLMU, given in the abstract), so it is in a better position than most. The architecture is specified well enough to count parameters: width 128, depth 40, 4 heads for 8M; width 256, depth 32, 4 heads for 25M (Appendix B). As a check, $12 \times 128^2 \times 40 = 7.9\text{M}$ and $12 \times 256^2 \times 32 = 25.2\text{M}$, which matches the stated sizes. Decoding is stated: greedy, temperature 0, left to right (§4.3). The optimizer is AdamW with betas 0.9 and 0.999 and a learning rate of 0.001 (Appendix B). The hardware for the timing run is 8 Tesla V100 32GB (§5.1). No pretrained models or API models are involved, so the usual problem of unversioned model snapshots does not arise here.

What I could not find:

- The configuration of the 3M model used in §5.3 and Figure 6. Only 8M and 25M are specified.
- Seeds for any table. Figure 4 uses three seeds; the others appear to be single runs, but the paper never says.
- The stopping rule. Training is "up to 3000 epochs" (§4.4). Which checkpoint is reported, and was it chosen on the same 2000 held-out mazes that produce the reported accuracy? Appendix D.2 calls the 2k A* mazes "validation," §4.4 calls the DFS 2000 a "held-out test set," and Appendix B selects hyperparameters on "held-out" accuracy. If these are the same sets, selection and reporting share data.
- Batch size. Appendix B says "128 and above," which is not a value.
- The positional scheme and numeric precision of the decoder-only baseline, and which MLM-U runs used 32-bit positions.
- Generator settings and seeds for the DFS mazes, and the version of the maze-dataset library [25]. The A* data come from [6], which helps, but the exact release should be named.
- Inference cost. The MLM-U decoder regenerates from mask tokens; how many forward passes per path does left-to-right decoding take, and how does it compare with the cached autoregressive baseline?

Hyperparameter selection is described inconsistently: §4.4 says the best configuration was chosen on "held-out shortest path accuracy," while Appendix B says "per-token held-out accuracy." Only one can be right.

4. What specifically to fix

1. **§5.1, compute paragraph, and the abstract.** 17.7 / 13.7 h is a 29.2% overhead, not 7%. Reconcile 2.17x (text) with 2.03x (Figure 4 caption), divide by 1.292, and restate the GPU-hours gain; on the numbers in the text, it is about 1.6x to 1.7x. Say in the abstract that it applies to 5x5 DFS mazes.
2. Rerun the autoregressive baseline at 20x20 and 30x30 DFS (Table 1) with 32-bit positional encodings, or state in §4.4 that it already used them. §5.4 shows this setting moves MLM-U from 40% to 93.8% at 30x30, so the question has to be closed for the baseline too.
3. **Tables 1 to 4, Table 6, Figure 6.** Report mean and spread over at least three seeds, at minimum for the 3M vs 8M scaling contrast and the 16-bit vs 32-bit contrast.
4. **Abstract and §1 bullet list.** Qualify "4x more sample efficient" as a 5x5 result (100k / 25k); for 10x10, only a lower bound exists.

5. **§5.1, first paragraph.** "Nearly 3x" should be about 5x ($93.8 / 18.8 = 4.99$, Table 1, column 30x30).
6. Harmonize the next-token numbers: 45.2 (Table 1, 10x10) versus "46%" (§5.1) and "around 50%" (§5.3); 20.6 (Table 1, 20x20) versus "22%" (§5.3). If they are different runs, say which.
7. **§4.1, DFS paragraph.** "Five levels of complexity ... spanning 5x5, 10x10, 20x20, and 30x30" lists four; add 15x15, which Table 1 reports.
8. **§5.2 and the introduction.** Replace "outperforms ... A* search trace supervision" with a statement that MLM-U ties at 10x10 (98.5 vs 98.5) and leads at 20x20 and 30x30 (Table 2), and that all baseline rows are copied from [6] and not retrained. Drop "identical settings" from the abstract for the A* part, or retrain at least the 15M next-token model under your pipeline.
9. Add one ablation that separates "multi-step" from the other changes MLM-U makes: path-only loss for next token, a fixed 15% mask, or the multi-head predictor of [4]. Place it next to Appendix E.1.
10. **§4.2.** State whether the next-token loss is computed over the whole sequence or only the solution path. Appendix D.1 restricts MLM-U to path tokens.
11. **§4.4 vs Appendix B.** Pick one selection metric (full-path or per-token), and state whether the selection set is disjoint from the 2000 reported mazes. Give the checkpoint rule for "up to 3000 epochs."
12. **Appendix B.** Give the 3M configuration and the exact batch size.
13. **Discussion.** Bring Table 6 into the limitations: MLM-U scores 0 full-path on unembedded 10x10 mazes, and both objectives score zero on larger ones.
14. Report inference cost per maze for both objectives (forward passes or wall-clock), since the efficiency argument is otherwise about training only.
15. Proofread. Examples: "multiple steps head" (§3), "explore the the" (§1), "in show on the right panel" (§4.1), "out of more than one are possibly the shortest ones" (§4.1 and D.2), "failed get the exact positions" (§5.4).

5. Recommendation

Major revisions. The central DFS result is large, clearly measured, and well outside test-set noise, and the paper is candid about several of its own weak spots. I expect it to survive revision. What keeps me from minor revisions is that one abstract-level claim ($\sim 2x$ in GPU hours) does not follow from the numbers printed in §5.1, the baseline may share the 16-bit positional problem that §5.4 fixed only for MLM-U, and the A* comparison is a cross-paper one presented as if it were controlled. The first is a rewrite; the second

needs one or two training runs; the third needs honest wording or one retrained baseline. None of this requires rethinking the study.

Declarations

Potential competing interests: No potential competing interests to declare.