

Peer Review

Review of: "Protein Structure Prediction in the 3D HP Model Using Deep Reinforcement Learning"

Evgenii Arsentev¹

1. Independent researcher

1. What the work claims

The authors treat folding in the 3D Hydrophobic-Polar lattice model as a sequential decision problem: the chain is built as a self-avoiding walk on the cubic lattice, one residue per step, and a Deep Q-Network with experience replay and a target network learns where to place the next residue, with the reward tied to the number of non-consecutive H-H contacts (§2.1, §2.2). Two Q-network designs are proposed. For short chains ($N \leq 36$) there is FFNN-R, a fixed sparse random reservoir followed by fully connected layers (§2.3.1). For longer chains there is LSTM-A, an LSTM whose outputs pass through multi-head attention before the Q-value head (§2.3.2). Two simpler variants, a plain FFNN and an LSTM that uses only the last hidden state (LSTM-OLH), serve as internal baselines (§3).

The abstract claims three things: that FFNN-R reaches optimal conformations "with 25% fewer training episodes"; that LSTM-A matches best-known energy values (BKVs) on longer sequences; and that both architectures show "consistent achievement of optimal conformations." The evidence is Table 3, which compares the final energies against BKVs from reference [18], a genetic algorithm (GA), BILS and TPPSO, plus learning curves (Figures 5 and 6), attention maps (Figure 7), and a short efficiency section (§3.1).

2. Whether the conclusion holds on the data

I want to start with what is genuinely good here. The problem formulation is clean and honest about its simplifications. The benchmark is a standard one, taken with its reference values from a published source, so readers can place the results. The authors report both successes and failures in Table 3 instead of hiding the hard sequences, they give trainable parameter counts (264,445 for FFNN-R against 6,324,741 for LSTM-A, §3), and they state the hardware and framework versions (one NVIDIA H100, PyTorch 2.5.0,

CUDA 11.8). Using a reservoir as a cheap feature map inside a DQN for lattice folding is, as far as I know, a new combination, and the comparison against LSTM-OLH and a plain FFNN is the right kind of internal ablation. The code is declared public. That is a solid base.

The difficulty is that the abstract and the conclusion say more than Table 3 shows. I went through it row by row.

LSTM-A on the 3d set. In Table 3 (column LSTM-A), the model reaches the BKV on 3d1 to 3d4: -11, -13, -9, and -18. On every longer sequence it stays above the BKV: 3d5 gives -33 against -35, 3d6 -30 against -31, 3d7 -32 against -34, 3d8 -40 against -44, 3d9 -51 against -55. Summed over 3d5 to 3d9, that is $2 + 1 + 2 + 4 + 4 = 13$ contacts short of the reference. So LSTM-A matches the BKV on 4 of 9 sequences, and on none of the five sequences longer than 36 residues, which are exactly the ones it was designed for. The abstract's "matching best-known energy values" for longer sequences and the Conclusion's "Both consistently match best-known energy values across benchmark sequences" are not what the table says. The text of §3 is more careful about 3d6 to 3d9 ("approached BKVs for more challenging sequences like 3d6-3d9"), but in the same sentence it says LSTM-A "matched BKVs for sequences 3d1-3d5", while Table 3 gives -33 for 3d5 against a BKV of -35. I would bring the abstract, §3, and the Conclusion into line with the table.

There is a real positive result hidden in the same rows, and I would state it instead. LSTM-A beats the GA column on 3d5 (-33 vs -32), 3d7 (-32 vs -30), 3d8 (-40 vs -37), and 3d9 (-51 vs -50), and loses to it only on 3d6 (-30 vs -31). It is much better than BILS everywhere. That is an honest, publishable claim.

FFNN-R on the A set. In the second block of Table 3, the sequence labels do not render, so I read the rows in the order of Table 2. Column FFNN-R then shows the BKV reached on A1 to A5 and A8 to A10, eight of the ten listed sequences, which agrees with the sentence in §3. On A6, it gives -11 against -12, and on A7, -12 against -13. Table 2 lists an eleventh sequence, A11, but Table 3 has only ten rows. I could not find a result for A11 anywhere in the rendered text. If it was run, it belongs in the table; if not, a sentence saying why would help.

The 25% claim. The abstract says FFNN-R reaches optimal conformations "with 25% fewer training episodes", and §3 says it needs "approximately 25% fewer training episodes" than the vanilla FFNN. I could not find the numbers behind this: no table of episodes-to-BKV per sequence and per architecture, and Figures 5 and 6 show only minimum energy against episode for four sequences. As rendered on Qeios, Table A.1, which the caption of Table 3 says holds "all experiments" (and calls "Table 4"), is not

legible, so the figure may live there. As it stands, the headline efficiency number cannot be checked from the paper.

Variance and repeated runs. §4 states that FFNN and FFNN-R "demonstrate robust performance across multiple independent training runs" and show "lower variance in final energy values" than ant colony optimisation or PERM. No number of runs, no standard deviations, and no success rates are given. DQN results on combinatorial problems are known to vary a lot between seeds, so a single best energy per sequence tells us what the method *can* find, not what it *usually* finds. I would report, for each sequence and architecture, the number of seeds, the best, mean, and worst final energy, and the fraction of runs that reach the BKV.

Missing baselines. The closest prior work is cited but not compared. Reference [15] (Yang et al., 2023) applies deep RL to the HP model, and the self-avoiding-walk formulation in §2.1 follows it; FoldingZero [9] is also a learned approach. Neither appears in Table 3. The GA, BILS, and TPPSO columns are not tied to a reference number in the text either, so the reader cannot tell which implementation and which budget they represent. Adding [15] as a column, and citing the source of each existing column, would make the comparison much stronger.

Compute comparisons. §4 says LSTM-A requires "significantly less computational time" than genetic algorithms and Monte Carlo methods, and that both models need "fewer computational resources" than chain-growth and evolutionary methods. §3.1 gives training times for LSTM-A (8 hours on average for sequences up to 36 residues, up to 48 hours for 3d5 to 3d9) and 1.5 hours for FFNN-R, but no run time is given for any baseline. Without that, the comparison is an impression, not a result.

Attention interpretation. The reading of Figure 7 goes further than the figure can carry. Saying that Head 1 shows "secondary structure detection" is a strong statement for a two-letter lattice model; I would either support it (for example, by relating attention peaks to the contact map of the final fold) or soften it. I also noticed a possible inconsistency: the caption gives a scale from purple (0.0) to yellow (0.05), and the initial yellow, uniform matrices would fit $1/N$ for $N = 20$ (3d1), which is a good sign if 3d1 is the sequence shown; the text does not say which sequence Figure 7 uses. But §4 then describes Head 2's "deeper blue-green regions" as "strong position-specific interactions", while on the stated scale blue-green is lower attention than yellow. It would help to check which way the colour bar runs.

3. Reproducibility

Some things are in place, and I am glad to see them: hardware, PyTorch and CUDA versions, the optimiser (Adam, learning rate 0.001) and the smooth L1 loss (§3), batch size 32 (§4), the LSTM hidden size of 512 (§2.3.2), reservoir sizes and sparsity (1000 nodes, 10% connectivity, §3.1), and the statement that code and data are on GitHub. The benchmark sequences are printed in full in Tables 1 and 2, which is very useful, although the BKV column of both tables is empty as rendered on Qeios (the values can be read from Table 3, except for A11).

Several items I could not find in the rendered text, and some may simply have been lost in the math rendering:

- The exact reward function. §2.2 says the reward is "based on the energy of the achieved fold", but not whether it is $-E$, a shaped version, or whether dead-end episodes (the ND state in §2.2.3) are penalised.
- The discount factor γ , the ε schedule, the replay buffer size, the target-network update period, and the warm-up length.
- The number of LSTM layers (§2.3.2 speaks of "an LSTM layer" and then of "the LSTM layers").
- Random seeds and the number of independent runs per sequence.
- The repository URL and a commit hash; the declaration says "a public GitHub repository" without a link in the text as rendered.
- Episode budgets per architecture. §3.1 gives 500,000 episodes for LSTM-A and about 200,000 for FFNN-R, but it is not said whether the plain FFNN and LSTM-OLH received the same budget, which matters for the 25% claim.

There are also internal inconsistencies in the architecture description that would stop someone from rebuilding the models exactly:

- §2.3.1 gives the FFNN-R hidden layers as 512, 256, 128, and 84. §3.1 describes them as "256 and 84 nodes". The parameter count in §3 (264,445) cannot fit the first version: a 1000-node reservoir feeding a 512-unit layer alone needs $1000 \times 512 = 512,000$ weights. A $1000 \rightarrow 256 \rightarrow 84 \rightarrow 5$ stack gives $256,256 + 21,588 + 425 = 278,269$ parameters with biases, closer but still not equal. Please state the actual layers and show how 264,445 is obtained.
- The reservoir is said to follow an Erdős-Rényi topology (§2.3.1) and also to be "initialized using Xavier uniform initialization" (§3). The spectral radius is not given, and the input matrix is called "trainable",

which differs from the usual echo-state setup. All of this is fine as a design choice, but it should be stated once and consistently.

- The caption of Figure 3 says "an 8-head attention mechanism"; §2.3.2 and §4 say four heads, and Figure 7 shows four.
- §2.2.3 first lists the actions as F, B, R, L, U, D, then says the six action slots are ND, F, L, R, U, D, while the network output in Figure 2 is a (5, 1) tensor. I think the intended meaning is five moves plus a termination flag in the state, but it took me some time to work out.
- §2.3.1 says a reservoir of about 3000 is used for $N = 48$ and 50, while §4 and §4.1 say FFNN-R performance degrades beyond 36 residues, and Table 3 reports FFNN-R only on 27-residue sequences. Was a 3000-node FFNN-R ever run, and on what?

4. What specifically to fix

1. **Abstract and §5 (Conclusion).** Replace "matching best-known energy values" and "consistently match best-known energy values", and in §3 "matched BKVs for sequences 3d1-3d5", with what Table 3 shows: BKV reached on 3d1 to 3d4 and on 8 of 10 A-sequences; on 3d5 to 3d9 LSTM-A is 1 to 4 contacts above the BKV but better than GA on four of five.
2. Support the "25% fewer training episodes" claim (abstract, §3) with a table: episodes to first reach the BKV, per sequence, for FFNN and FFNN-R, over several seeds.
3. Add multi-seed statistics to Table 3 (best, mean $\pm$ SD, success rate). Without them, the "lower variance" sentence in §4 should go.
4. Table 3: add a column for reference [15], cite the source of the GA, BILS, and TPPSO values, and add A11 or explain its absence (it is in Table 2).
5. Make Table A.1 readable on Qeios, and fix the Table 3 caption, which refers to it as "Table 4".
6. §2.3.1 vs §3.1 vs §3: reconcile the FFNN-R layer sizes with the 264,445 parameter count.
7. Figure 3 caption: 8 heads or 4?
8. §2.2: write out the reward, γ , ϵ schedule, buffer size, and target update period; §2.3.2: the number of LSTM layers.
9. §4, claims about computational time versus GA and Monte Carlo: either add measured run times for the baselines under comparable hardware, or remove the comparison.
10. §4, Figure 7 interpretation: check the colour scale against the "blue-green = strong" reading, and relate the head patterns to actual contacts before calling them secondary-structure detectors.

11. The phrase "state-of-the-art results" in §4.1 is not supported by Table 3, where the BKV column is better on five of nine 3d sequences; I would drop it.
12. A small point in §2.1: on a cubic lattice with unit steps and self-avoidance, the $90^\circ/180^\circ$ bond-angle condition is satisfied automatically, so it can be stated as a property rather than a constraint.

5. Recommendation

Major revisions. The method is sensible, the benchmark is standard, and the honest reading of Table 3 already contains a useful result: a learned DQN policy that recovers optimal folds on short chains and beats a GA on most long ones. The revisions I ask for are mostly about matching the claims to that table, adding seed statistics and the missing baseline, and making the architecture description internally consistent. None of it requires a new idea, but the efficiency claim and the variance claim need data that are not in the current version.

Declarations

Potential competing interests: No potential competing interests to declare.