**Discussion 3.1 – Architecture, Units, and Sequential Modeling in RNNs**

1. How does the hidden state in an RNN differ conceptually from weights in a feedforward neural network, and why is this difference critical for modeling sequential dependencies?
2. LSTM and GRU units introduce gating mechanisms. What does each gate (input, forget, output in LSTMs; update and reset in GRUs) *mean* in terms of information flow, and how do they mitigate the vanishing gradient problem described by Bengio, Simard, and Frasconi (1994)?
3. Cho et al. (2014) proposed GRUs as a simplification of LSTMs. In what types of problems might the reduced complexity of a GRU be an advantage, and when might it be a disadvantage?
4. The recurrent update equation ht=σ(Wxhxt+Whhht−1+bh) extends the equation of a line to handle sequential data. How does this linear foundation support the nonlinear, memory-retaining behavior of RNNs?
5. Sequential data often contain long-range dependencies, but RNNs struggle to capture them. Beyond LSTM/GRU modifications, what other strategies (architectural, preprocessing, or training-based) could address these limitations?
6. In sequential modeling, how do you balance the trade-off between capturing long-term dependencies (larger sequence lengths) and avoiding overfitting or excessive computational cost?

**Required:** Read all your peers' posts, then comment meaningfully on two or more.

References

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks, 5*(2), 157–166. <https://doi.org/10.1109/72.279181>

Cho, K., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. *arXiv*. <https://arxiv.org/abs/1406.1078>