universität freiburg

Kolmogorov-Arnold Networks for Q-Learning

Aaryan Bhandari, Shanmugapriya Kanagasabapathi, Tanya Dora
University of Freiburg
Baohe Zhang

GOAL- Compare the generalizability of KAN, MLP and LSTM in Q-learning

Introduction

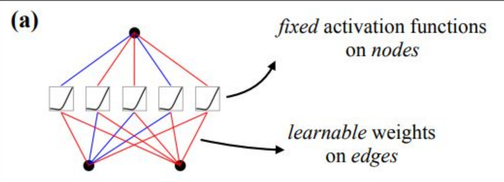
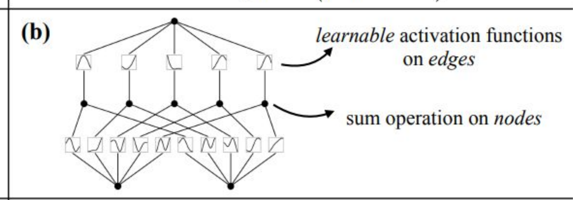
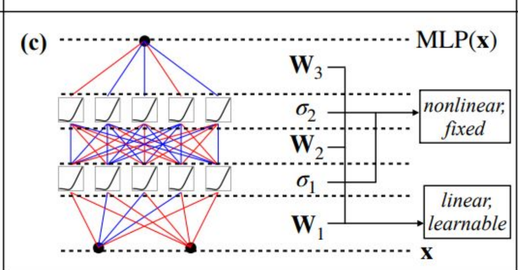
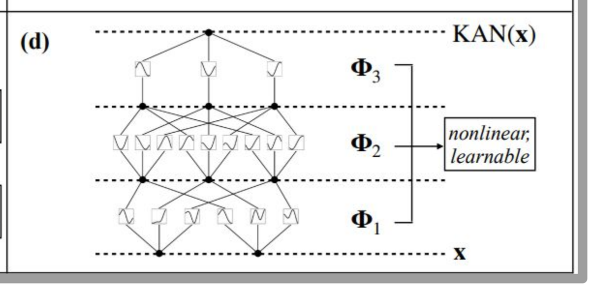
Model	Multi-Layer Perceptron (MLP)	Kolmogorov-Arnold Network (KAN)
Theorem	Universal Approximation Theorem	Kolmogorov-Arnold Representation Theorem
Formula (Shallow)	$f(\mathbf{x}) \approx \sum_{i=1}^{N(\epsilon)} a_i \sigma(\mathbf{w}_i \cdot \mathbf{x} + b_i)$	$f(\mathbf{x}) = \sum_{q=1}^{2n+1} \Phi_q \left(\sum_{p=1}^{n} \phi_{q,p}(x_p) \right)$
Model (Shallow)	(a) 	(b) 
Formula (Deep)	$\mathrm{MLP}(\mathbf{x}) = (\mathbf{W}_3 \circ \sigma_2 \circ \mathbf{W}_2 \circ \sigma_1 \circ \mathbf{W}_1)(\mathbf{x})$	$\mathrm{KAN}(\mathbf{x}) = (\mathbf{\Phi}_3 \circ \mathbf{\Phi}_2 \circ \mathbf{\Phi}_1)(\mathbf{x})$
Model (Deep)	(c) 	(d) 

Table 1: Compare MLP and KAN

Method

Component	LSTM Equation	KAN-LSTM Equation
Input gate i_t	$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$	$i_t = \sigma(\Phi_{ii}x_t + \Phi_{hi}h_{t-1})$
Forget gate f_t	$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf})$	$f_t = \sigma(\Phi_{if}x_t + \Phi_{hf}h_{t-1})$
Cell state candidate g_t	$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg})$	$g_t = \tanh(\Phi_{ig}x_t + \Phi_{hg}h_{t-1})$
Output gate o_t	$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho})$	$o_t = \sigma(\Phi_{io}x_t + \Phi_{ho}h_{t-1})$
Cell state c_t	$c_t = f_t \odot c_{t-1} + i_t \odot g_t$	$c_t = f_t \odot c_{t-1} + i_t \odot g_t$
Hidden state h_t	$h_t = o_t \odot \tanh(c_t)$	$h_t = o_t \odot \tanh(c_t)$

Table 2 : Compare LSTM and KAN based LSTM

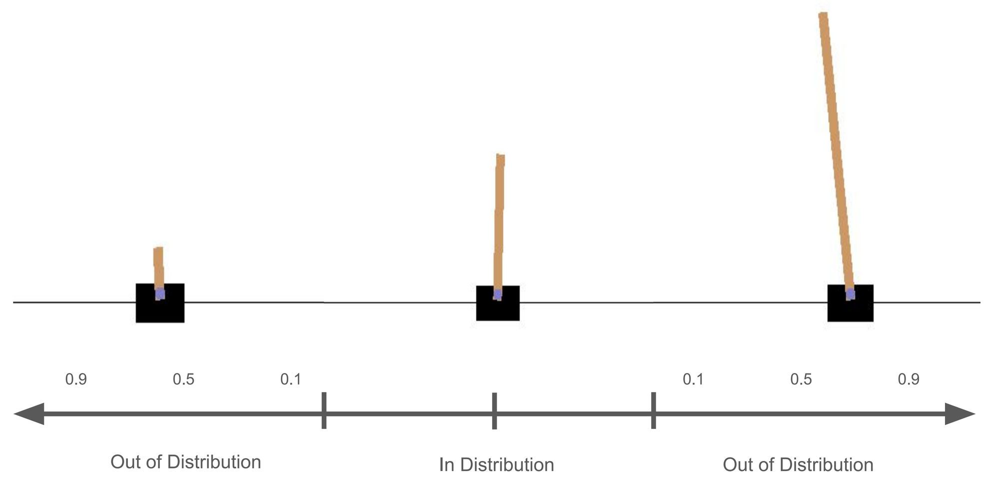


Figure 1: Cartpole distributions

Reinforcement Learning Algorithms Applied

- DQN (Deep Q Network)
- ADRQN (Action Specific Deep Recurrent Q Network)
- PPO (Proximal Policy Optimization)

This experiment evaluates a model's performance in the CartPole environment under **In-Distribution** and **Out-of-Distribution** scenarios.

- In-Distribution (Default Settings): Gravity: 9.8, Cart Mass: 1.0, Pole Mass: 0.1, Pole Length: 0.5, Force: 10.0, Time Step: 0.02
- Out-of-Distribution: ±20% changes in all parameters.

Experiments and Results

- **KAN vs. MLP**: KAN outperforms MLP in the default CartPole environment with DQN.
- **Performance with Environment Changes**: KAN and MLP show comparable performance when sampled with 20% parameter changes.
- **KAN-LSTM vs. LSTM (ADRQN)**: KAN-LSTM outperforms LSTM for CartPole environment.
- **KAN-LSTM vs. LSTM (PPO)**: KAN-LSTM underperforms compared to LSTM. For PPO, the model was trained with 100,000 steps and tested for 1,000 steps in the CartPole environment.
- **Model Evaluation**: Best models selected at regular intervals (every 20 of 10,000 episodes for DQN, every 5 of 500 for ADRQN).

Evaluation of MLP vs KAN for Default Environment using DQN

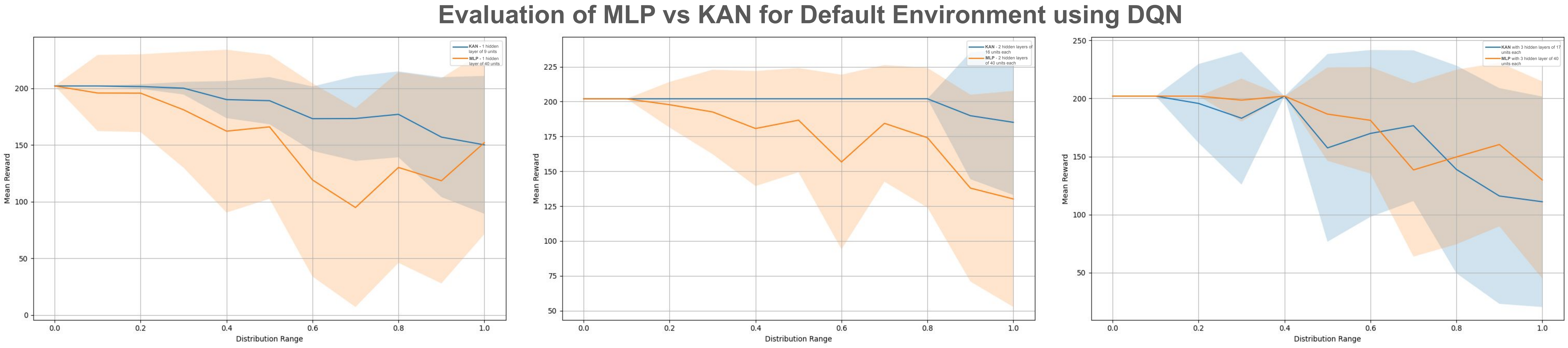


Figure 2: In distribution and out of distribution performance of cartpole trained on default environment

Evaluation of MLP vs KAN for Changing Environment using DQN

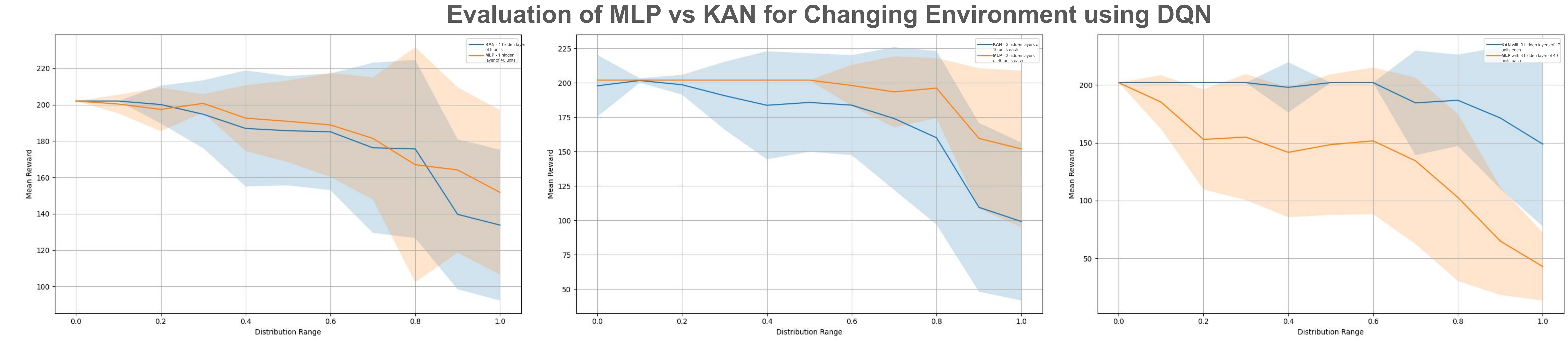


Figure 3: In distribution and out of distribution performance of cartpole trained on changing environment

Evaluation of LSTM vs KAN-based LSTM using ADRQN

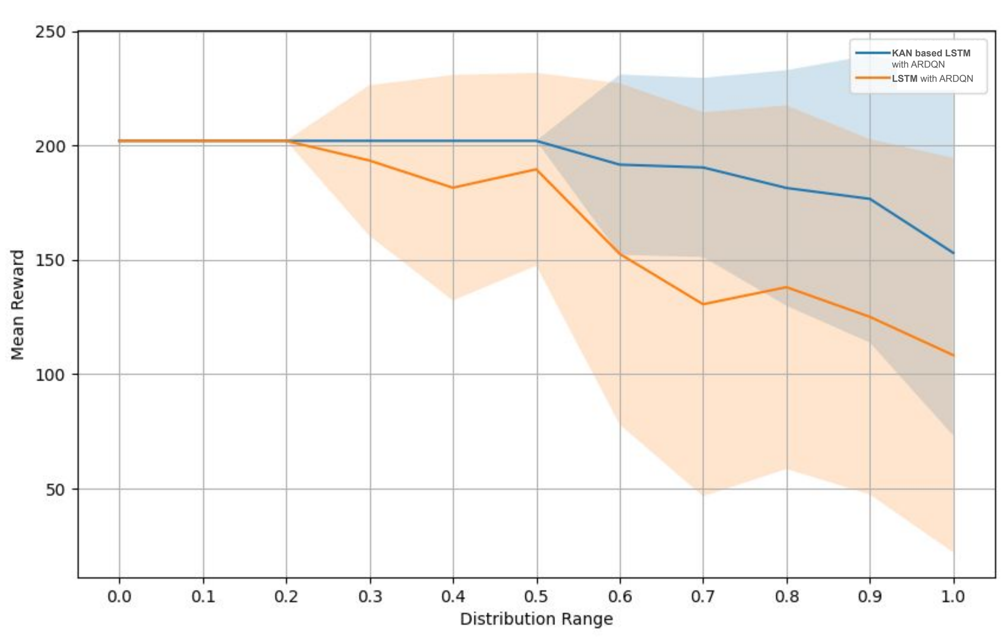


Figure 4: Mean reward of LSTM and KAN based LSTM for cartpole using ADRQN for increasingly out of distribution environments

Evaluation of LSTM vs KAN-based LSTM using PPO

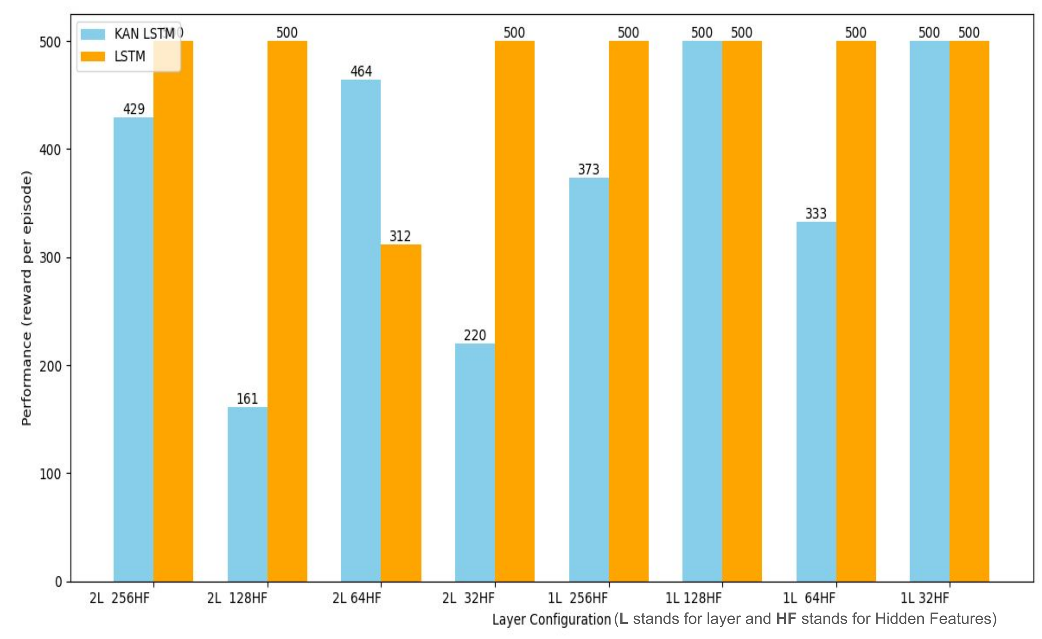


Figure 5: Comparison of LSTM and KAN based LSTM on Cartpole using Proximal Policy Optimization

Future Work and Conclusion

Our study shows that KAN networks offer better generalization than MLPs in default CartPole environment with DQN for similar number of learnable parameters, and perform similarly when environment parameters change. While KAN-LSTM surpasses LSTM with ADRQN, it lags behind LSTM with PPO. Future work should focus on optimizing KAN-LSTM with different algorithms and exploring its effectiveness in varied environments. Overall, KAN's strong performance in dynamic settings highlights its potential for further research and improvement.

References

[1] Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljacic, M., Hou, T.Y., & Tegmark, M. (2024). KAN: Kolmogorov-Arnold Networks. *ArXiv, abs/2404.19756*.
[2] Li, Z. (2024). Kolmogorov-Arnold Networks are Radial Basis Function Networks. *ArXiv, abs/2405.06721*.
[3] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M.A. (2013). Playing Atari with Deep Reinforcement Learning. *ArXiv, abs/1312.5602*.
[4] Zhu, P., Li, X., & Poupart, P. (2017). On Improving Deep Reinforcement Learning for POMDPs. *ArXiv, abs/1704.07978*.
[5] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *ArXiv, abs/1707.06347*.