Supplementary Material for Unified Local-Cloud Decision-Making via Reinforcement Learning

Kathakoli Sengupta, Zhongkai Shangguan, Sandesh Bharadwaj, Sanjay Arora[†], Eshed Ohn-Bar, and Renato Mancuso

Boston University [†]Red Hat

Abstract. This supplementary provides additional implementation and ablative details. Specifically, we discuss additional details regarding the introduced cloud and energy-aware environment as well as additional ablative analysis. Our supplementary video depicts qualitative results, comparing roll-outs generated by the proposed model to baseline models.

1 Environment Implementation

Energy Cost Model: To quantify energy consumption, we assume a standard cost per floating point operation model [2, 4]. For instance, a 675 Kb $480 \times 480 \times 3$ image and a 1.37 M parameters MobileVIT [7] model would result in energy consumption on a local GPU that is 0.15 J (based on a factor of 0.095 J per flop [4]). We further add an energy cost for every communication, i.e., transmitting raw data or an embedding to the cloud [3]. Here, energy consumed for local-cloud communication is computed as 6.94×10^{-5} J per byte [4]. When transmitting only an embedding (a 24×24 array) from the local policy as an intermediate input to the cloud policy, the communication energy consumption decreases, e.g., from the original image which requires 1.55 J to 25.18 mJ.

Cloud Communication Model: Cloud transmission involves a latency, which the model should optimize over, e.g., in safety-critical scenarios based on the task. Our communication model resembles the one used by Kang etal [4], where latency is modeled as a Gaussian with an average time of 0.5 s and variance of 0.1 s. To extend our research, we also experimented with modeling latency using a Pareto distribution. Under the Pareto distribution, we observed a 7% reduction in ENS, with an ENS value of 80.45, infractions at 0.03 /m, and energy consumption at 2.76 J/m. While this is a conservative estimate, we will release our simulation, code, and models to facilitate the analysis of RL agents across diverse computational and latency configurations.

1.1 Reward Design

Reward Progression: Fig. 1 shows our reward analysis for RL baselines and UniLCD models. The reward progression indicates that the baselines fail to learn any meaningful information throughout the training and continue to oscillate within a very narrow

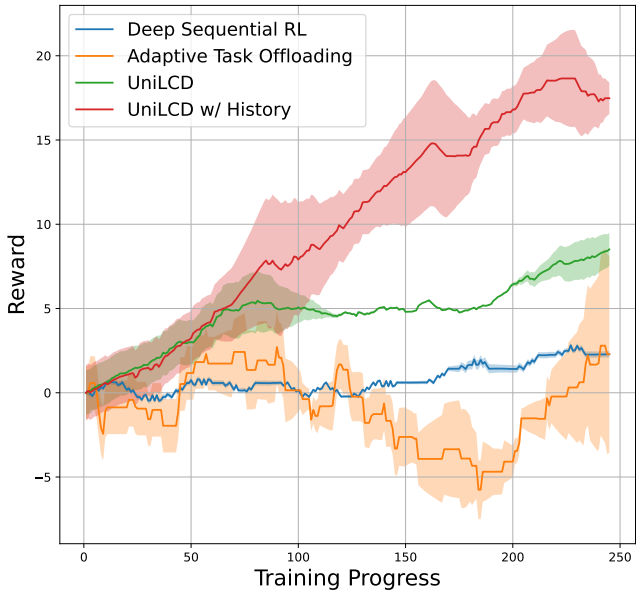

Fig. 1: Training Progression. We evaluate the rewards for our overall framework throughout the training process against baseline models. We demonstrate high sample-efficiency compared to prior methods, particularly when the routing module is given a history input, i.e., prior actions and their source (cloud or local decision). Despite common instabilities in training reinforcement learning models, UniLCD is shown to achieve significantly higher reward early in the training process.

range. Our UniLCD demonstrates a slight improvement in the initial episodes but exhibits delayed convergence due to its slower learning rate. Remarkably, the UniLCD with history model displays a distinct pattern of rapid advancement, characterized by a steep rise in the initial episodes. Consequently, it is clear that this configuration learns and adapts to the desired behavior more efficiently than the comparative approaches, even within a few iterations. The primary benefits in sample efficiency stem from the definition of the reward function definition and the careful design of the routing module and its inputs.

Comparison with Standard Additive Reward: In the main paper, we find it crucial to carefully shape the reward function in order to effectively optimize for several different types of rewards. Our reward is contrasted with current standard approaches that linearly combine the rewards [1, 5, 13], i.e.,

$$r_{standard} = \alpha_g r_{geo} + \alpha_s r_{speed} + \alpha_e r_{energy} + \alpha_c r_{collision} \tag{1}$$

adding the geodesic, speed, energy, and collision rewards and α_g, α_s, α_e, α_c, are tuned scalar hyperparameters (we perform careful tuning of these with a grid search).

Impact of Reward Components: Table 1 analyzes the impact of various reward terms on task performance. We show each term to contribute to the overall task performance.

Table 1: Reward Component Ablation. We investigate the impact of each reward component in our reward design for UniLCD. ENS is Ecological Navigation Score (%), NS is Navigation Score (%), SR is Success Rate (%), RC is Route Completion (%), Infract. is Infraction Rate (/m), Energy is measured in Joules per meter (J/m), and FPS is Frames Per Second (empirically measured and averaged throughout the trials).

Reward	ENS↑	NS↑	SR↑	RC↑	Infract.↓	Energy↓	FPS↑
All Terms	**85.97**	**94.58**	**93.33**	**95.90**	**0.02**	**2.90**	**26.49**
w/o r_{geo}	67.04	74.65	0.00	76.22	0.03	7.42	58.82
w/o r_{speed}	66.05	72.66	0.00	77.34	0.09	6.61	65.40
w/o r_{energy}	0.00	93.53	90.00	95.50	0.03	43.75	6.42

We note that we do not multiply the collision reward with the other terms in order to emphasize it during training. This provides a **major negative disadvantage to avoid collisions**, as it dominates the overall reward once it occurs (and results in episode termination). Moreover, we show that without penalizing energy consumption of communication to and processing in the cloud, the model effectively learns to rely on the more powerful model. Finally, while the imitation-learned models take as input the goal, adding a geodesic reward term is found to help ensure the routing module and final action prediction do not result in a policy that deviates from the planned path.

Real-world Validation: Our simulation experiments are based on real-world network latency calculations, as detailed in Sec 1. To extend our research into real-world validations, we conducted a small-scale experiment using an RC vehicle platform (Traxxas X-Maxx) equipped with a Jetson Nano. In this setup, reward terms were computed in a self-supervised manner using an EAI XL2 LiDAR with a minimum range of 0.1 meters. Given the need for high sample efficiency in real-world reinforcement learning (RL), we performed an ablation study comparing Proximal Policy Optimization (PPO) and CrossQ. In Line 16 of Algorithm 1, CrossQ learns a policy in approximately five minutes, demonstrating a 21% improvement over [31].

Further Work: While our work takes a step towards developing vision-based systems and effectively address multifaceted objectives, several simplifications were necessary. Due to the difficulty and sample inefficiency associated with the optimization of RL agents for energy, latency, and safety constraints – all are crucial in real-world settings – we adopted a step-by-step training approach. Specifically, the local and cloud policies were trained using imitation learning by assuming expert trajectories. Our UniLCD pipeline can readily support additional model fine-tuning and incorporation of consumption-based reward terms, e.g., both the cloud and local models can be fine-tuned using RL, yet this requires additional study in the future.

1.2 Performance Over Task Difficulty

The results in the main paper are shown for the high (30 pedestrians) density settings of our environment, spawned along the route (Fig. 2). For completeness, we report in Table 2 the model performance across additional pedestrian density settings along the

Fig. 2: Examples of Different Environmental Settings in Our Robot Navigation Environment. We vary the pedestrian density in order to stress-test the proposed UniLCD method. Pedestrian count along the path ranges from 5 (Low), 15 (Medium), 30 (High), and 70 (Crowd).

path, starting from a low-density scenario with 5 pedestrians to a crowded scenario with 70 pedestrians. As shown in the table, several models degrade in performance, particularly in crowded settings. Remarkably, our final UniLCD model exhibits only slight degradation (1%), in terms of ENS, in the most challenging and crowded settings.

Table 2: Pedestrian Density Ablation. Model performance is shown for different pedestrian densities along the path, from Low (mostly empty, 5 pedestrians), Medium (15 pedestrians), High (30 pedestrians), and up to Crowded (70 pedestrians). † denotes methods transmitting to the cloud raw input data instead of an embedding. ENS is Ecological Navigation Score (%), NS is Navigation Score (%), SR is Success Rate (%), RC is Route Completion (%), Infract. is Infraction Rate (/m), Energy is measured in Joules per meter (J/m), and FPS is Frames Per Second.

Method	Setting	ENS↑	NS↑	SR↑	RC↑	Infract.↓	Energy↓	FPS↑
† Cloud-Only [8]	Low	0.00	98.41	95.00	99.10	0.01	33.93	8.91
	Medium	0.00	97.14	93.33	98.50	0.02	35.11	8.74
	High	0.00	96.47	93.33	98.50	0.03	36.49	7.11
	Crowd	0.00	88.24	92.00	95.24	0.11	48.93	6.52
Local-Only [9]	Low	66.18	69.70	0.00	75.23	0.11	3.78	78.19
	Medium	65.99	69.70	0.00	75.23	0.11	3.98	73.67
	High	63.43	67.33	0.00	75.23	0.16	4.33	68.40
	Crowd	35.47	37.55	0.00	63.16	0.75	4.94	65.40
Baseline Methods:								
SPINN [6]	Low	43.71	85.13	70.00	95.12	0.16	16.88	23.84
	Medium	39.94	80.60	66.66	93.24	0.21	16.74	24.21
	High	36.31	72.75	60.00	92.73	0.35	18.94	20.37
	Crowd	32.07	53.22	60.00	90.76	0.77	17.16	21.84
Compressive Offloading [12]	Low	11.81	80.43	0.00	80.43	0.00	88.23	2.46
	Medium	12.38	80.16	0.00	80.16	0.00	89.12	2.17
	High	13.98	80.16	0.00	80.16	0.00	90.66	1.82
	Crowd	10.06	75.66	0.00	75.66	0.00	92.66	1.19
† Selective Query [3]	Low	27.87	64.78	0.00	82.68	0.03	42.64	21.57
	Medium	25.40	60.63	0.00	80.12	0.08	44.87	19.39
	High	24.14	61.28	0.00	82.68	0.11	45.35	18.14
	Crowd	14.90	44.27	0.00	79.36	0.52	51.72	14.38
Neurosurgeon [4]	Low	43.79	67.62	0.00	85.12	0.01	25.62	12.53
	Medium	42.12	65.12	0.00	82.54	0.02	26.48	13.21
	High	39.85	63.10	0.00	80.54	0.03	28.31	12.53
	Crowd	32.94	51.19	0.00	75.58	0.24	29.18	9.61
† Adaptive Offloading [10]	Low	89.38	97.15	90.0	98.51	0.02	5.02	34.91
	Medium	51.87	55.56	80.00	97.42	0.81	4.22	33.39
	High	37.42	40.37	70.00	94.05	1.22	4.80	30.14
	Crowd	33.96	37.67	80.00	95.38	1.34	6.39	28.71
Deep Sequential RL [11]	Low	70.15	73.02	0.00	79.36	0.12	3.07	78.10
	Medium	66.66	69.56	0.00	79.36	0.19	3.25	78.03
	High	58.84	61.83	0.00	79.36	0.36	3.77	77.94
	Crowd	26.47	27.93	0.00	75.28	1.43	4.30	76.12
UniLCD Variations:								
† Standard Reward	Low	49.37	56.15	0.00	75.23	0.10	2.47	52.63
	Medium	49.37	56.15	0.00	75.23	0.10	3.12	51.18
	High	48.35	54.99	0.00	75.23	0.13	3.57	50.20
	Crowd	30.60	34.80	0.00	75.23	0.79	4.86	47.74
† Standard Reward w/ History	Low	51.44	58.81	10.00	77.71	0.08	7.02	51.42
	Medium	50.38	57.60	10.00	77.71	0.11	7.57	50.20
	High	50.04	57.20	10.00	77.71	0.12	8.38	49.07
	Crowd	35.22	40.26	10.00	75.23	0.58	10.57	43.16
† Our Reward	Low	51.82	85.73	60.00	90.00	0.07	18.49	20.82
	Medium	51.20	84.71	60.00	92.70	0.13	20.57	17.14
	High	48.30	79.90	56.66	91.15	0.19	21.72	16.05
	Crowd	38.79	64.16	60.00	86.45	0.43	24.57	11.20
† Our Reward w/ History	Low	73.72	90.17	83.33	94.66	0.07	5.78	36.61
	Medium	72.20	88.32	83.33	94.66	0.10	6.51	34.73
	High	71.70	87.71	83.33	94.66	0.11	7.83	33.98
	Crowd	59.04	72.22	83.33	93.33	0.37	10.83	30.62
Our Reward	Low	61.88	94.54	60.00	95.20	0.01	5.51	**14.18**
	Medium	60.68	92.71	60.00	94.66	0.03	5.88	**13.72**
	High	57.20	87.39	60.00	91.10	0.06	6.60	**12.49**
	Crowd	50.22	76.74	60.00	90.00	0.23	8.43	**10.73**
Our Reward w/ History	Low	**88.10**	**96.92**	**93.33**	**97.60**	**0.01**	**2.58**	30.02
	Medium	**87.06**	**95.77**	**90.00**	**96.44**	**0.01**	**2.63**	29.12
	High	**85.97**	**94.58**	**93.33**	**95.90**	**0.02**	**2.90**	26.49
	Crowd	**83.04**	**91.36**	**93.33**	**95.90**	**0.07**	**3.12**	22.84

References

1. Booth, S., Knox, W.B., Shah, J., Niekum, S., Stone, P., Allievi, A.: The perils of trial-and-error reward design: misdesign through overfitting and invalid task specifications. In: AAAI (2023)
2. Hadidi, R., Cao, J., Xie, Y., Asgari, B., Krishna, T., Kim, H.: Characterizing the deployment of deep neural networks on commercial edge devices. In: IISWC (2019)
3. Kag, A., Fedorov, I., Gangrade, A., Whatmough, P., Saligrama, V.: Efficient edge inference by selective query. In: ICLR (2022)
4. Kang, Y., Hauswald, J., Gao, C., Rovinski, A., Mudge, T., Mars, J., Tang, L.: Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. SIGARCH (2017)
5. Knox, W.B., Allievi, A., Banzhaf, H., Schmitt, F., Stone, P.: Reward (mis) design for autonomous driving. J. Artif. Intell. (2023)
6. Laskaridis, S., Venieris, S.I., Almeida, M., Leontiadis, I., Lane, N.D.: Spinn: synergistic progressive inference of neural networks over device and cloud. In: MobiCom (2020)
7. Mehta, S., Rastegari, M.: Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. arXiv (2021)
8. Radosavovic, I., Kosaraju, R.P., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: CVPR (2020)
9. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: CVPR (2018)
10. Van Tam, N., Hieu, N.Q., Van, N.T.T., Luong, N.C., Niyato, D., Kim, D.I.: Adaptive task offloading in coded edge computing: A deep reinforcement learning approach. COMML (2021)
11. Wang, J., Hu, J., Min, G., Zhan, W., Ni, Q., Georgalas, N.: Computation offloading in multi-access edge computing using a deep sequential model based on reinforcement learning. Commun. Mag. (2019)
12. Yao, S., Li, J., Liu, D., Wang, T., Liu, S., Shao, H., Abdelzaher, T.: Deep compressive offloading: Speeding up neural network inference by trading edge computation for network latency. In: SenSys (2020)
13. Zhuang, Z., Fu, Z., Wang, J., Atkeson, C., Schwertfeger, S., Finn, C., Zhao, H.: Robot parkour learning. In: CoRL (2023)