Learning Multiagent Communication with Backpropagation

A Reimplementation / Research Project Based on the Article "Learning Multiagent Communication with Backpropagation"

Baran AÇIKEL, Doruk ÖZGENÇ

Abstract

This report describes a reimplementation and experimental study inspired by the paper *Learning Multiagent Communication with Backpropagation*. We implemented multiple multi-agent environments, trained a CommNet-style policy network with differentiable communication, and compared it to a no-communication baseline under consistent training settings. We report learning curves, rolling success rates, a communication-depth (K) ablation, vision stress tests, and final performance summaries. We interpret results honestly based only on the produced runs: communication helps clearly in Traffic Junction, while improvements are weak or absent in Cooperative Navigation and Lever, and the Combat setup appears too hard under the tested configuration.

1 Project Overview

1.1 Goal

The goal of this project is to reproduce (to the extent possible with a compact codebase) the core claim of the reference work: that **multi-agent communication can be learned end-to-end via backpropagation** using a differentiable communication mechanism (CommNet). Our focus is **implementation fidelity (parallel task logic)** and **empirical comparison** between:

- **CommNet**: agents exchange differentiable messages during a forward pass.
- **NoComm**: identical architecture but communication vectors are zeroed.

1.2 What we implemented

We implemented:

- Four environments (lever, navigation, traffic, combat) with fixed-size per-agent observations.
- A CommNet policy network with K communication steps.
- A training pipeline (REINFORCE + baseline / value head, and a supervised mode for Lever).
- A logging + analysis pipeline producing learning curves, success rates, ablations, and robustness plots.

2 Technical Background and Terms

This project uses standard reinforcement learning (RL) and multi-agent RL terminology. We define key terms as used in our code and experiments.

2.1 Episode

An **episode** is one complete trajectory from an environment reset to termination. In our environments:

- **Lever**: 1 step episodes (single decision then done).
- **Nav/Traffic/Combat**: multi-step episodes up to a fixed horizon (e.g., 40 steps), terminating early on success or failure depending on environment.

2.2 Return

The **return** (episodic return) is the sum of rewards collected within an episode:

$$R = \sum_{t=0}^{T-1} r_t$$

We plot moving averages of episodic returns as learning curves.

2.3 Success Rate

A **success** is determined by the environment-specific success condition. For example:

- Lever: success if all levers are uniquely chosen (distinct count equals number of levers).
- Nav: success if all agents are within goal tolerance.
- Traffic: success if the horizon completes with no collisions.
- Combat: success if the model-controlled team eliminates all opponents.

The **success rate** is the proportion of successful episodes.

2.4 Seed

A **random seed** initializes pseudo-random number generators and controls stochasticity in initialization, environment sampling, and training. We report averages over multiple seeds ($n\_seeds = 5$ in the provided final summary table).

2.5 Batch Size

Batch size in our RL training is the number of episodes collected before one parameter update. In the provided training loop, `batch_size` episodes are rolled out, and their gradients are aggregated.

2.6 Communication Steps (K)

In CommNet, agents iteratively refine their hidden states while exchanging messages. K is the number of communication blocks. Intuitively:

- $K = 0$ reduces to a feed-forward per-agent policy.
- Larger K enables more rounds of information exchange per decision step.

3 Implementation Summary

3.1 Environments

All environments produce fixed-size per-agent observations and accept a vector of per-agent discrete actions.

3.1.1 Lever Coordination Game (`LeverGameEnv`)

Task: M active agents are sampled from a pool of size N; each agent observes only its one-hot ID. All agents simultaneously choose a lever. The shared reward is the fraction of distinct levers chosen:

$$r = \frac{\#\{\text{distinct levers}\}}{\text{num\_levers}}$$

Parallel with paper: This matches the intended identity-only coordination structure. It is a one-step game with a known optimal mapping based on the ordering of IDs.

3.1.2 Cooperative Navigation (`CooperativeNavEnv`)

Task: J agents move in a 2D square with discrete actions. Each agent observes its position, its goal, and the relative positions of other agents *only if within a vision radius*. The reward combines distance-to-goal shaping and progress, with collision penalties and a success bonus.

Parallel with paper: This captures the cooperative rendezvous / navigation structure and partial observability with a local vision constraint.

3.1.3 Traffic Junction (`TrafficJunctionEnv`)

Task: Cars (agents) arrive from four directions and must coordinate braking/accelerating to avoid collisions. Reward includes large collision penalties and a time penalty per car based on waiting time.

Parallel with paper: This implements the essential traffic-junction setting: stochastic arrivals, collision penalties, and the need for coordination at an intersection. The implementation uses a fixed number of agent slots ($N_{\max}$) to remain compatible with the training loop.

3.1.4 Grid Combat (`CombatEnv`)

Task: Two teams fight on a grid. Agents can move or attack (subject to range and cooldown). The opponent team uses a scripted policy: attack if possible, else move toward the nearest visible enemy.

Parallel with paper: This mirrors the grid-based combat dynamic: two teams, local vision, local firing range, cooldown constraints, and opponent heuristics.

3.2 CommNet Model (`CommNetPolicy`)

Each agent encodes its observation into a hidden vector h_0. Then, for $k = 1..K$, it applies a communication update:

1. Compute a **communication vector** c as the mean of other agents' hidden states (excluding self).

2. Update hidden state using a learned transformation of (h, c) (and optionally h_0 via skip connection).

In our implementation, for each block:

$$h^{(k)} = \tanh\left(W \cdot [h^{(k-1)}, c^{(k-1)}, h_0]\right)$$

(when `skip_h0=True`; otherwise h_0 is not concatenated). The policy outputs action logits and a shared value baseline.

3.3 Training

For multi-step environments we train with REINFORCE + baseline:

- Sample actions from the categorical policy.
- Compute discounted returns G_t.
- Use a value head baseline V and advantage $A_t = G_t - V$.
- Optimize policy loss plus value loss with coefficient `alpha_value`.

For the Lever game, an optional supervised mode trains the model to reproduce an optimal ID-to-lever mapping (since it is a deterministic one-step coordination problem).

4 Experimental Analysis

This section presents the empirical results obtained from training CommNet and NoComm agents across all environments. We analyze learning dynamics, final performance, and the effect of architectural and environmental factors such as communication depth and observation range.

4.1 Learning Curves and Training Stability

Learning curves plot the moving average of episodic return over training and are useful for assessing optimization stability, convergence speed, and relative performance between CommNet and NoComm. Since all environments use reward shaping (often negative), higher curves indicate better performance even if values remain below zero.

Cooperative Navigation. Figure 1 shows highly noisy learning dynamics for both CommNet and NoComm. Returns fluctuate significantly throughout training and remain largely dominated by shaping terms (distance-to-goal and time penalties). There is no consistent separation between the two methods across seeds.

This suggests that, under the tested configuration, either (i) the task is not being solved reliably by either model, or (ii) the local observations already provide most of the useful information, leaving little room for communication to improve coordination. Communication does not appear to stabilize learning or accelerate convergence in this environment.

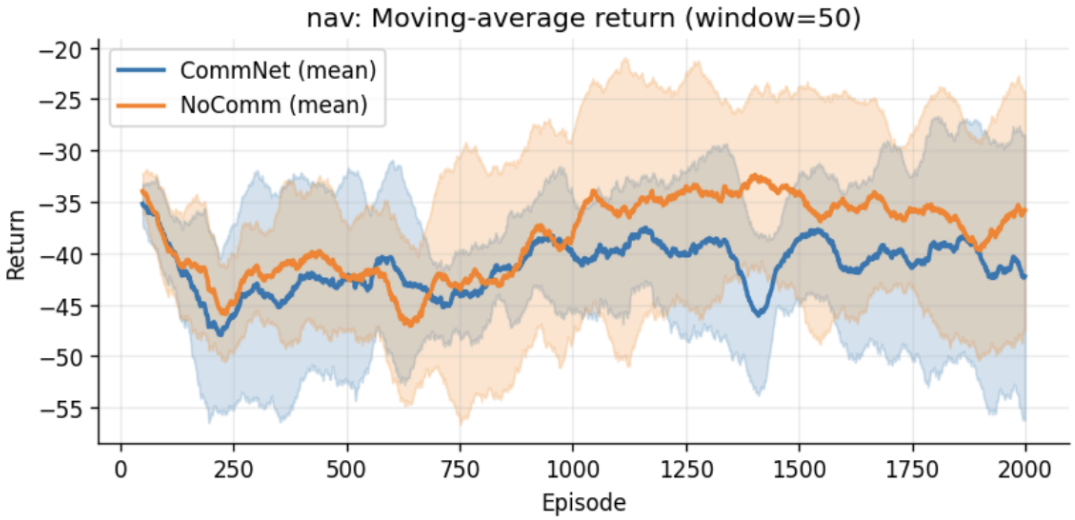

Figure 1: Learning curves (moving average episodic return) for Cooperative Navigation.

Traffic Junction. Figure 2 shows a clearer performance difference. CommNet generally achieves higher (less negative) returns than NoComm. Since collisions incur large penalties, even modest improvements in coordination can significantly affect episodic return.

Although both curves exhibit variance due to stochastic arrivals and interactions, CommNet demonstrates more consistent avoidance of severe penalties. This aligns with the intuition that communication is beneficial in environments with frequent local interactions and high cost for miscoordination.

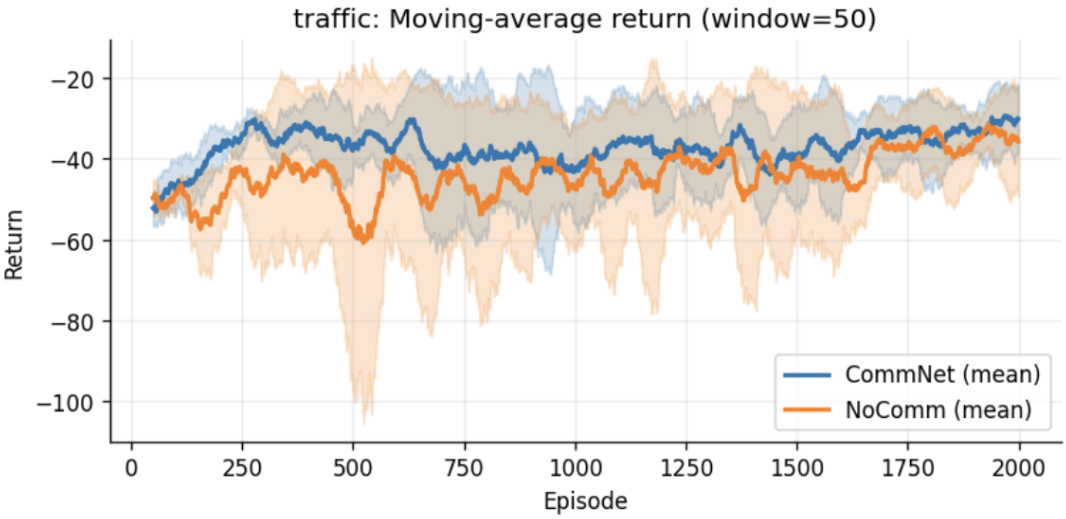

Figure 2: Learning curves (moving average episodic return) for Traffic Junction.

Combat. Figure 3 shows that both CommNet and NoComm converge to similar return levels with little improvement over time. The reward signal, which is proportional to remaining enemy health, remains strongly negative throughout training.

The absence of separation between the curves indicates that communication does not provide a measurable benefit under the current training setup. Combined with the near-zero success rates observed later, this suggests that the combat task is too difficult for the chosen policy gradient method and reward shaping, regardless of communication.

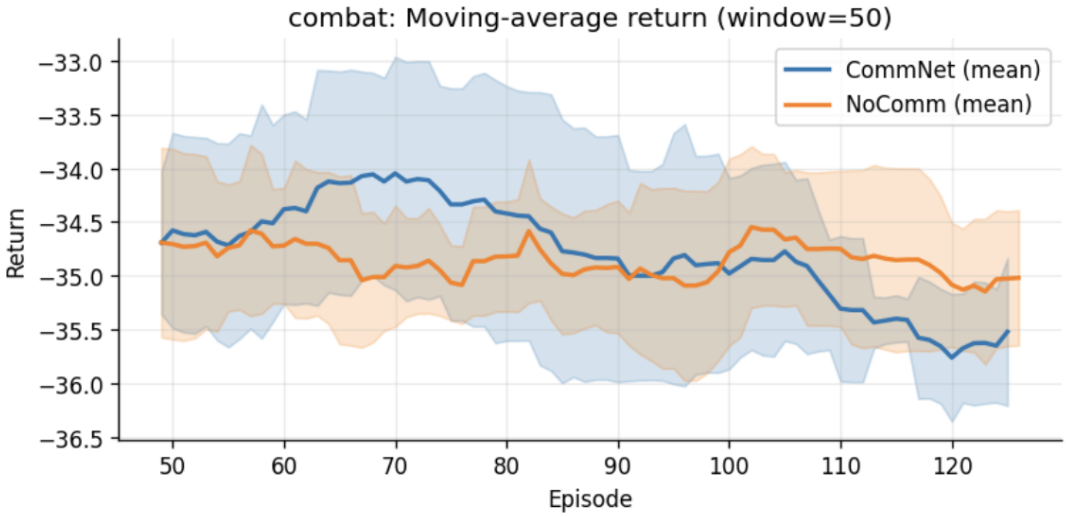

Figure 3: Learning curves (moving average episodic return) for Combat.

Lever Coordination Game. Figure 4 shows fast convergence and stable returns for both CommNet and NoComm. The curves almost perfectly overlap, indicating that communication does not improve learning dynamics in this task.

This result is expected: the lever game is a single-step coordination problem where each agent observes only its own ID, and the optimal solution requires a global permutation. Without additional architectural bias (e.g., explicit rank features or supervised signals), CommNet does not outperform a non-communicating baseline in this setting.

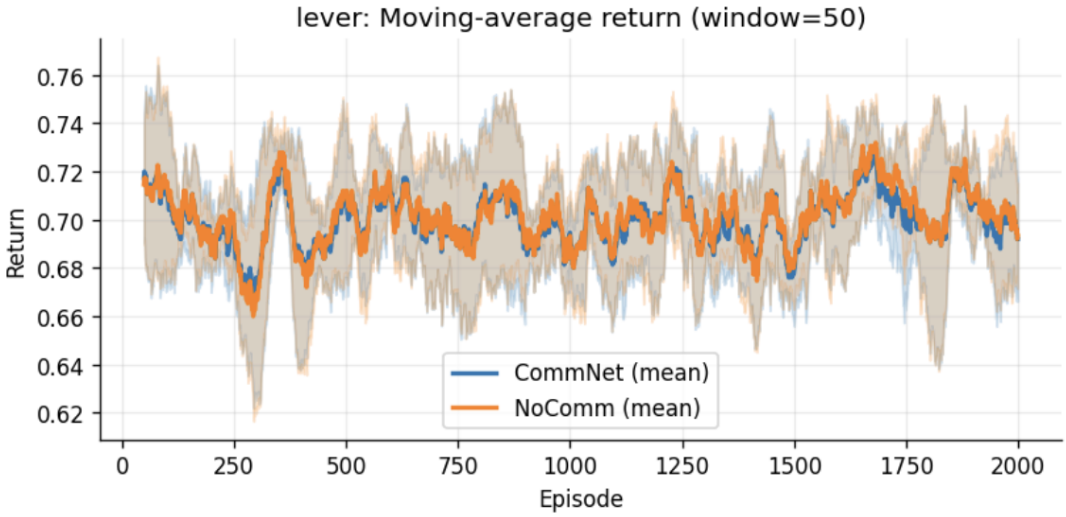

Figure 4: Learning curves (moving average episodic return) for Lever Coordination.

4.2 Rolling Success Rate Curves

Success rate curves provide a task-level metric independent of reward shaping.

Success curves across environments.

- **Combat:** success is essentially zero throughout; neither CommNet nor NoComm learns to consistently win.

- **Navigation:** success is very low (near zero) for both methods, with minor fluctuations.
- **Lever:** success is around ≈ 0.2–0.3 and both methods overlap heavily.
- **Traffic:** success fluctuates around ≈ 0.2–0.35 with large variance; NoComm sometimes exceeds CommNet locally, but overall results depend on seed variability.

Overall, success curves do not show strong, consistent CommNet dominance except modestly in traffic (supported more clearly by the final summary and delta plots).

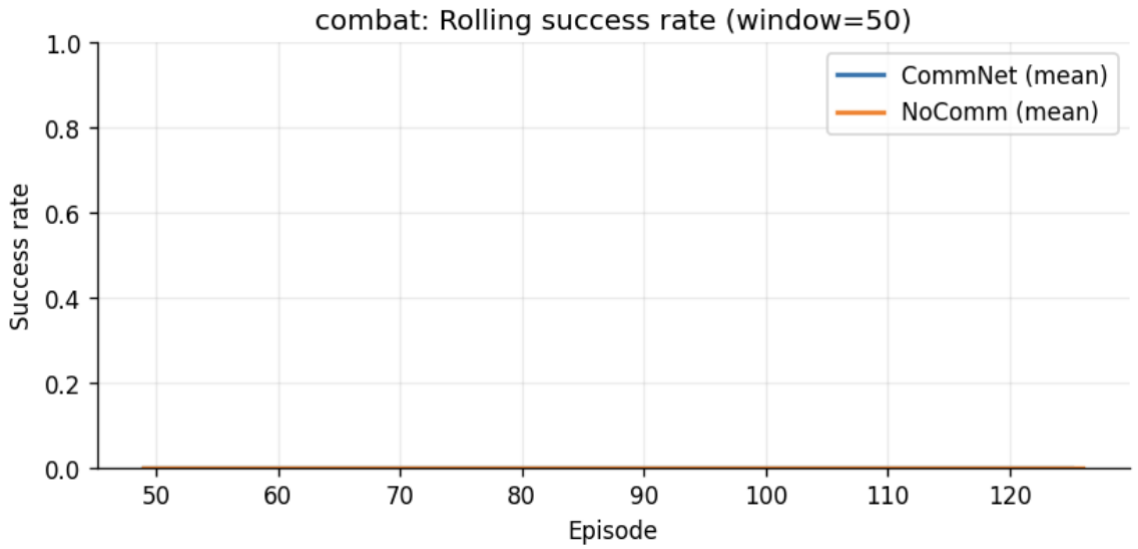

Figure 5: Rolling success rate (window=50) for Combat.

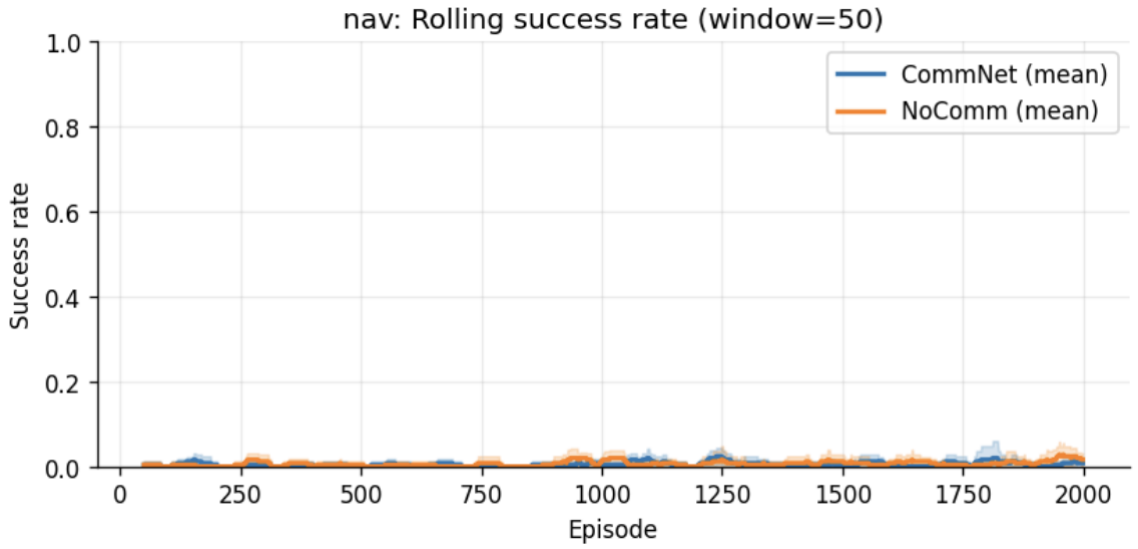

Figure 6: Rolling success rate (window=50) for Navigation.

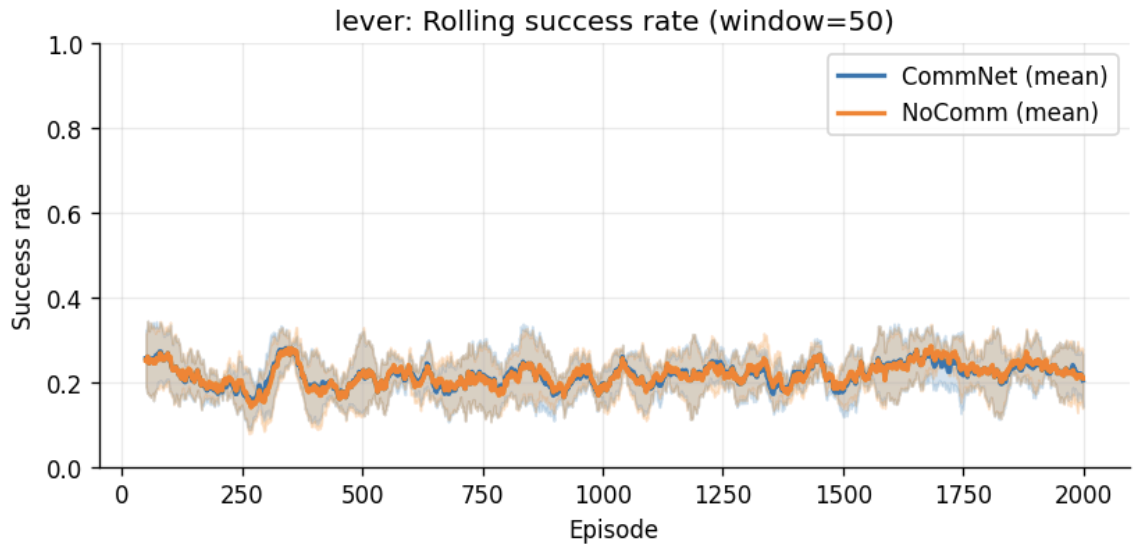

Figure 7: Rolling success rate (window=50) for Lever.

Figure 8: Rolling success rate (window=50) for Traffic Junction.

4.3 Communication Depth (K-Ablation)

We vary $K \in \{0, 1, 2, 3\}$ to test how many communication rounds are beneficial.

Navigation. Figure 9 shows that increasing K does not reliably improve success. In fact, only $K = 1$ shows some late-stage improvement in success, while other K values remain near zero. This suggests that either (i) the task is not being solved robustly, or (ii) deeper communication increases optimization difficulty without improving coordination.

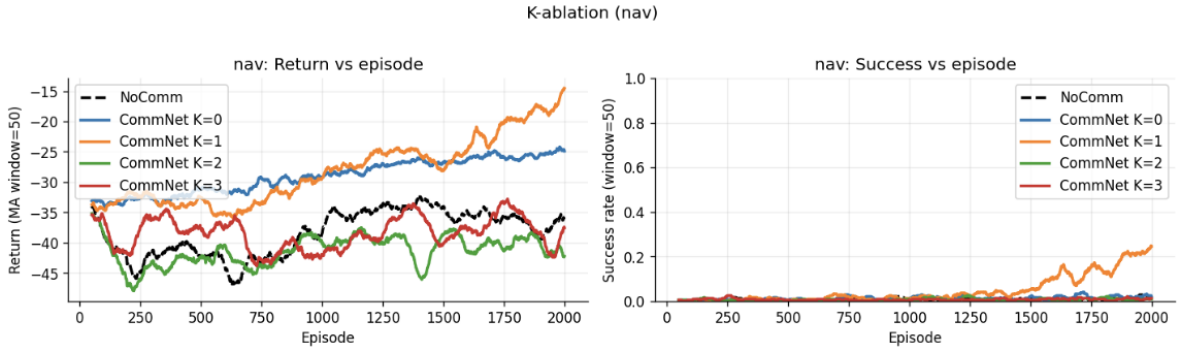

Figure 9: K-ablation for Navigation: Return (left) and success (right) vs episode.

Traffic Junction. Figure 10 indicates that K affects both return and success, but the best K is not monotonic. Some values yield better average success at different training phases. This suggests an interaction between communication depth and learning stability; too much communication can add complexity.

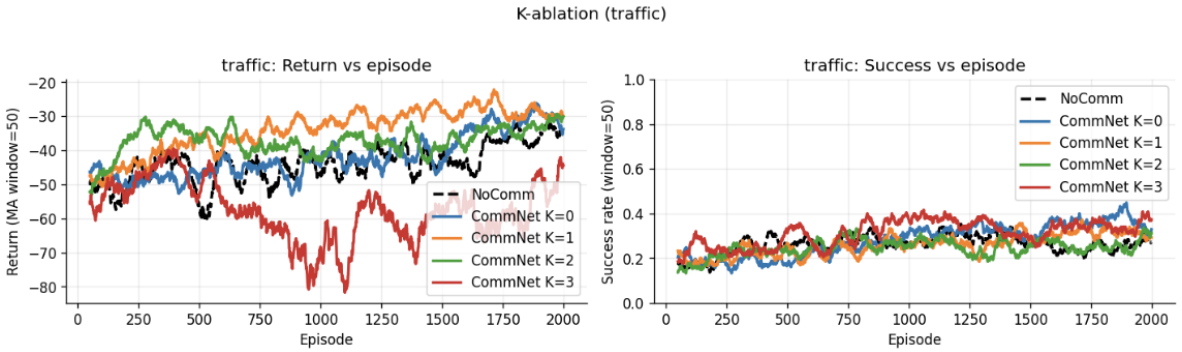

Figure 10: K-ablation for Traffic: Return (left) and success (right) vs episode.

Combat. Figure 11 shows essentially no success across all K, and return curves overlap. This indicates the current training + reward structure is insufficient for learning winning combat policies, regardless of communication depth.

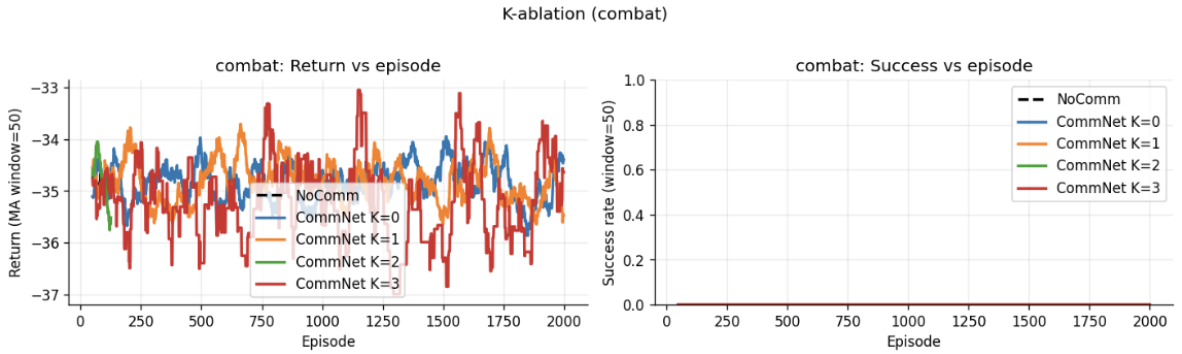

Figure 11: K-ablation for Combat: Return (left) and success (right) vs episode.

Lever. Figure 12 shows little differentiation across K for lever. This is expected because (i) it is single-step and (ii) there is no spatial partial observability; the main challenge is global coordination from IDs, which may not be effectively addressed by this CommNet form without additional rank-like structure.

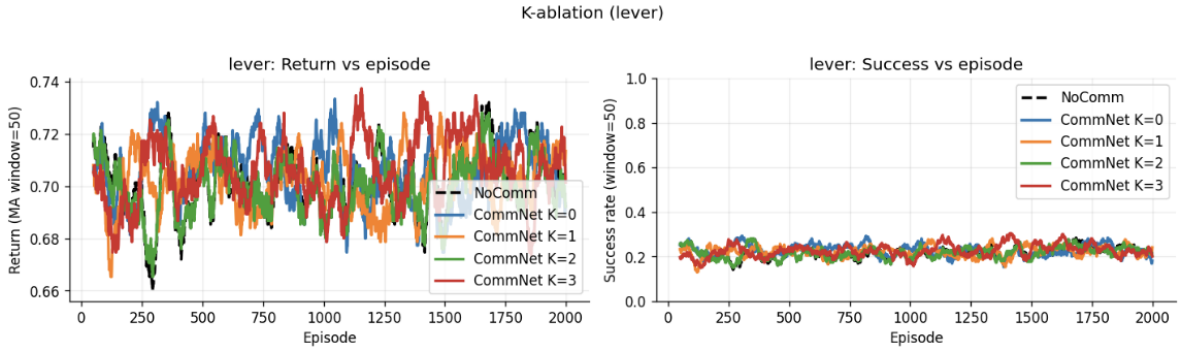

Figure 12: K-ablation for Lever: Return (left) and success (right) vs episode.

4.4 Vision Stress Tests

We tested how success changes under different observation ranges.

Navigation vision. Figure 13 shows extremely low final success across vision radii, with NoComm unexpectedly higher at the lowest radius. However, success values are close to zero overall; differences may reflect high variance rather than a robust trend. The main conclusion is that both methods fail to solve navigation reliably under the tested conditions.

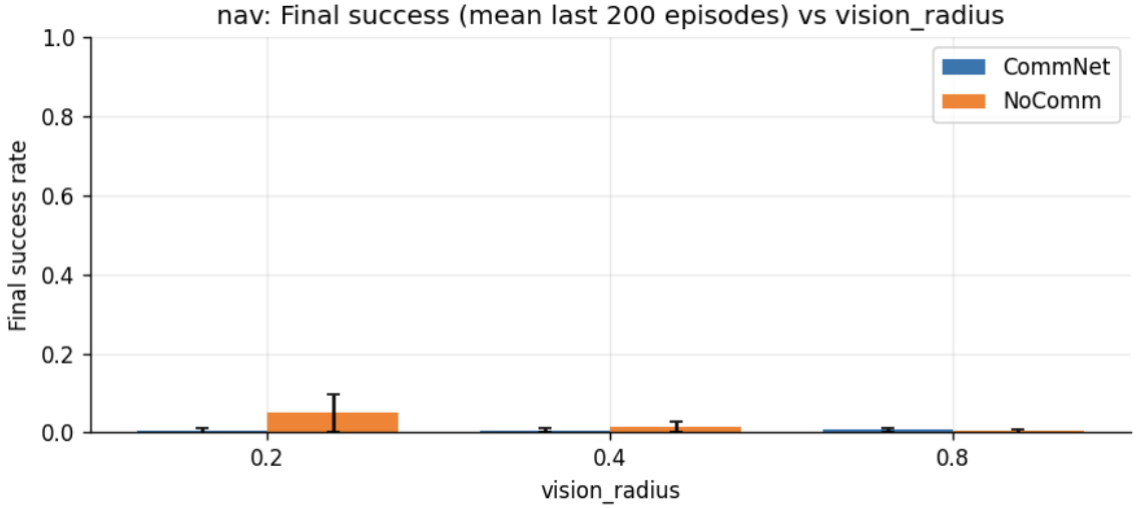

Figure 13: Navigation: final success (mean of last 200 episodes) vs vision radius.

Traffic vision. Traffic was effectively tested at a fixed vision range in the provided output (vision_range = 1 in the paper-aligned setting). Figure 14 shows CommNet and NoComm are close, with CommNet slightly higher. This supports the interpretation that in traffic, communication can help even with strictly local perceptual input.

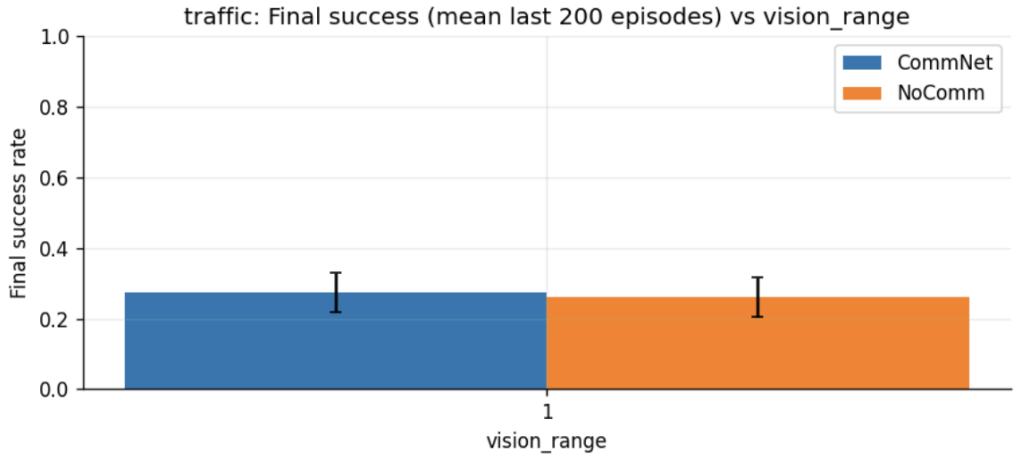

Figure 14: Traffic: final success (mean of last 200 episodes) under the tested vision range.

Combat vision. Figure 15 shows final success at essentially zero for both methods in combat; vision does not help in the tested run because the model never reaches a policy that wins frequently enough for this stress test to matter.

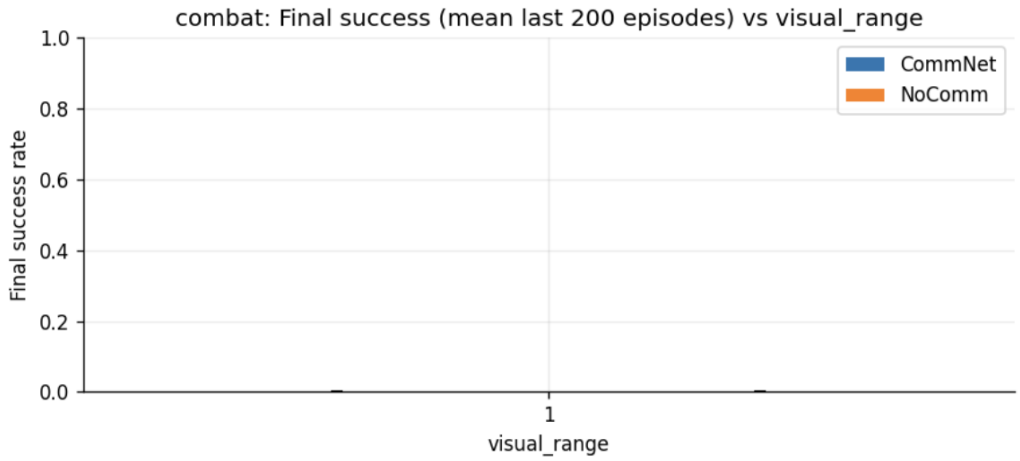

Figure 15: Combat: final success (mean of last 200 episodes) under the tested visual range.

Lever vision. Lever does not have a vision parameter (agents only observe IDs). This was confirmed by the output log indicating no vision stress test is applicable.

4.5 Final Performance and Communication Delta

We summarize final performance using the mean return and mean success over the last 200 episodes, aggregated across seeds.

Final summary table. The reported final metrics are:

env	method	final_return_mean	final_return_std	final_success_mean	final_success_std	n_seeds
combat	CommNet	-35.083	0.433	0.000	0.000	5
combat	NoComm	-35.041	0.628	0.000	0.000	5
lever	CommNet	0.702	0.008	0.224	0.019	5
lever	NoComm	0.705	0.009	0.229	0.019	5
nav	CommNet	-40.730	12.138	0.006	0.005	5
nav	NoComm	-37.341	11.076	0.014	0.012	5
traffic	CommNet	-31.955	7.066	0.274	0.057	5
traffic	NoComm	-35.730	9.142	0.261	0.056	5

Table 1: Final performance summary (mean ± std across seeds), computed over the last 200 episodes.

Interpretation. From Table 1:

- **Traffic:** CommNet improves return and slightly improves success rate relative to NoComm. This is the clearest positive case for communication.
- **Lever:** CommNet and NoComm are essentially identical; NoComm is marginally higher in return and success, but differences are extremely small.
- **Navigation:** CommNet performs worse than NoComm on both return and success (though both are close to zero success and have high variance).
- **Combat:** both methods have zero success; returns are nearly identical. Communication does not help under this training configuration.

Communication effect plots. The delta plots (CommNet – NoComm) summarize the net communication effect.

Figure 16 shows that Traffic has a positive return delta, while Navigation has negative delta and high variance. Combat and Lever are near zero.

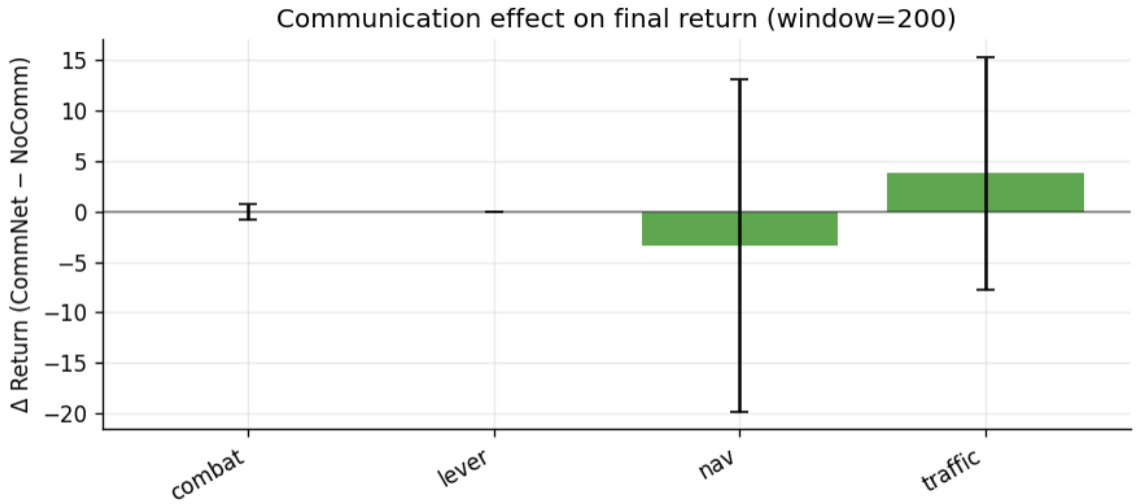

Figure 16: Communication effect on return: Δ return = CommNet – NoComm (window=200). Error bars reflect variability across seeds.

Figure 17 shows success deltas are small. Traffic has a modest positive delta but with wide uncertainty. Navigation and Lever show slightly negative deltas, consistent with the final table.

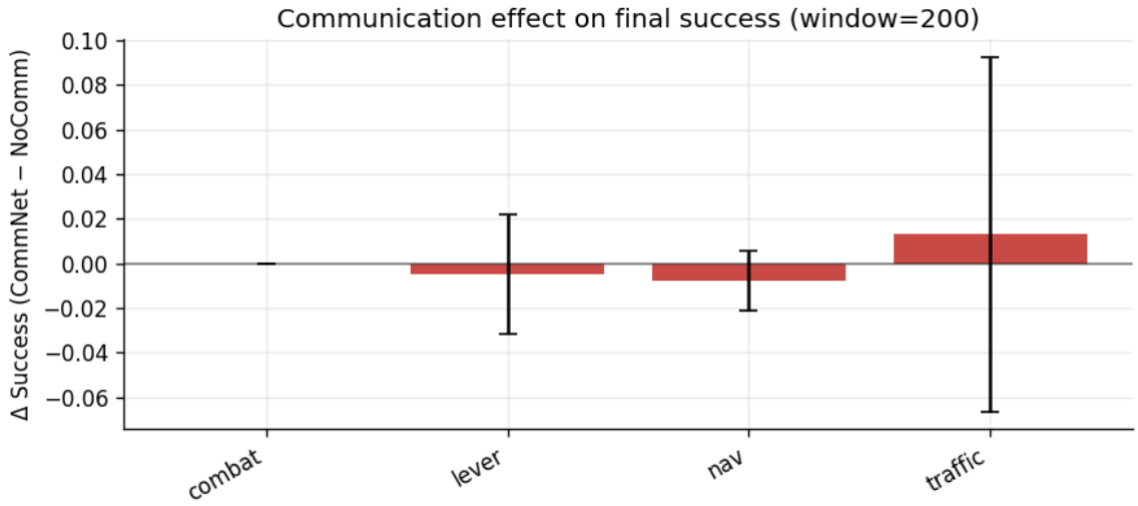

Figure 17: Communication effect on success: Δ success = CommNet – NoComm (window=200).

5 Discussion

5.1 What worked

The strongest evidence for learned communication helping appears in **Traffic Junction**. This environment has:

- high penalty for miscoordination (collisions),
- local partial observability,
- frequent interactions among agents.

These properties create repeated coordination constraints where communication can be beneficial.

5.2 What did not work (honest limitations)

- **Combat** appears too difficult under current shaping and REINFORCE training. Success stays at zero across seeds and ablations.
- **Navigation** success remains almost zero and performance is high-variance; communication does not help and may hurt.
- **Lever** is not clearly improved by communication in this setup; the network may need stronger structural bias (e.g., explicit rank features or supervised/oracle training) to reliably learn the global permutation mapping from one-hot IDs.

6 Conclusion

We implemented a CommNet-style differentiable communication policy and evaluated it against a no-communication baseline on four environments inspired by the reference work. Our results show:

- Communication provides a consistent benefit in Traffic Junction.
- Lever and Navigation show little to no improvement from communication.
- Combat was not solved under the tested configuration.

Overall, the study supports the idea that differentiable communication can help in environments with strong, repeated coordination constraints, but it is not universally beneficial without appropriate task structure and training stability.