Language Grounded Multi-agent Reinforcement Learning with Human-interpretable Communication

Huao Li[*]
University of Pittsburgh
hul52@pitt.edu

Hossein Nourkhiz Mahjoub
Honda Research Institute USA, Inc.
hossein_nourkhizmahjoub@honda-ri.com

Behdad Chalaki
Honda Research Institute USA, Inc.
behdad_chalaki@honda-ri.com

Vaishnav Tadiparthi
Honda Research Institute USA, Inc.
vaishnav_tadiparthi@honda-ri.com

Kwonjoon Lee
Honda Research Institute USA, Inc.
kwonjoon_lee@honda-ri.com

Ehsan Moradi-Pari
Honda Research Institute USA, Inc.
emoradipari@honda-ri.com

Michael Lewis
University of Pittsburgh
ml@sis.pitt.edu

Katia Sycara
Carnegie Mellon University
sycara@andrew.cmu.edu

Abstract

Multi-Agent Reinforcement Learning (MARL) methods have shown promise in enabling agents to learn a shared communication protocol from scratch and accomplish challenging team tasks. However, the learned language is usually not interpretable to humans or other agents not co-trained together, limiting its applicability in ad-hoc teamwork scenarios. In this work, we propose a novel computational pipeline that aligns the communication space between MARL agents with an embedding space of human natural language by grounding agent communications on synthetic data generated by embodied Large Language Models (LLMs) in interactive teamwork scenarios. Our results demonstrate that introducing language grounding not only maintains task performance but also accelerates the emergence of communication. Furthermore, the learned communication protocols exhibit zero-shot generalization capabilities in ad-hoc teamwork scenarios with unseen teammates and novel task states. This work presents a significant step toward enabling effective communication and collaboration between artificial agents and humans in real-world teamwork settings.

1 Introduction

Effective communication is crucial for multiple agents to collaborate and solve team tasks. Especially in ad-hoc teamwork scenarios where agents do not coordinate a priori, the ability to share information via communication is the keystone for successful team coordination and good team performance [30]. Multi-Agent Reinforcement Learning (MARL) methods have shown promise in allowing agents to learn a shared communication protocol by maximizing the task reward [38, 37]. However, merely

[*]Work done while interning at Honda Research Institute USA. Code available at https://romanlee6.github.io/langground_web/.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-2790

maximizing the utility of communication in specific task goals might compromise task-agnostic objectives such as optimizing language complexity and informativeness [40, 16], making the learned communication protocols 1) hard to interpret for humans or other agents that are not co-trained together [18, 20, 6] and 2) highly data-inefficient [9]. In addition, most previous works learn communication protocols with atomic symbols or a combination of them leaving the relation between symbols unexplored [50]. Only a few researches attempt to learn a semantic space for zero-shot communication of unseen states [42, 17]. One of the popular directions for interpretable communication is to regulate the learning process with external knowledge from human languages [41, 22, 28, 1]. However, this process is challenging due to the divergent characteristics between human and machine languages by nature [5]. For example, most agent training methods (e.g., deep reinforcement learning) require a huge amount of data that is impractical for human-in-the-loop training or even collecting from humans [28].

The rise of Large Language Models (LLMs) provides new opportunities in grounding agent communication with human languages. Recent generative models fine-tuned with human instructions (e.g., GPT-4, Llama 3) show reasonable capabilities in completing team tasks and communicating in a human-like fashion via embodied interaction [36, 25]. Essentially, LLMs encapsulate a highly trained model of human language patterns in teamwork, allowing them to generate descriptions and responses that are well-grounded in natural language. For the purpose of guiding multi-agent communication, they represent the most generally available reference based on a vast corpus of human language data that would be infeasible to collect through other means. However, LLMs are known to suffer from a lack of grounding with the task environments (a.k.a. hallucinations), which prevents embodied agents from generating actionable plans [29]. While attempts have been made to ground LLMs with reinforcement learning or interactive data collected from environments [47, 4, 39], none of them involve teamwork nor communication among multiple embodied agents.

In this paper, we propose LangGround, a novel computational pipeline for artificial agents to learn human-interpretable communication for ad-hoc human-agent teamwork. Specifically, we use synthetic data generated by LLM agents in interactive teamwork scenarios to align the communication space between MARL agents with human natural language. Learning signals from both language grounding and environment reinforcement regulate the emergence of a communication protocol to optimize both team performance and alignment with human language. We have also evaluated the learned communication protocol in ad-hoc teamwork scenarios with unseen teammates and novel task states. The aligned communication space enables translation between high-dimensional embeddings and natural language sentences, which facilitates ad-hoc teamwork. The proposed computational pipeline does not depend on specific MARL architecture or LLMs and should be generally compatible. We have sought to minimize the influence of prompt engineering to ensure the seamless applicability of our approach in diverse environments. To the best of our knowledge, this work is among the very first attempts at training MARL agents with human-interpretable natural language communication and evaluating them in ad-hoc teamwork experiments.

2 Related Work

2.1 Multi-Agent Communication

Reinforcement learning has been used to coordinate the teamwork and communication among multiple agents in partially observable environments. In earlier works such as DIAL [12], CommNet [38], and IC3Net [37], agents learn differentiable communication in an end-to-end fashion under the pressure of maximizing task reward. Other works use shared parameters [17] or a centralized controller [33] to stabilize the non-stationary learning process of multi-agent communication. More recently, representation learning methods such as autoencoder [26] and contrastive learning [27] are used to ground an agent's communication on individual observations. However, comm-MARL methods usually suffer from overfitting to specific interlocutors trained together [19]. The learned communication protocols can not be understood by unseen teammates in ad-hoc teams, let alone another human.

Another relevant trend of research is Emergent Communication (EC), where researchers focus more on simulating the development of natural (i.e., symbolic) language with artificial agents [19, 15]. The most common task scenarios used in the EC community are reference games or Lewis signaling games [24], in which a speaker must describe an object to a listener, who must then recognize

it among a set of distractions [42]. However, previous research has shown that learning EC in more complicated, scaled-up, and multi-round interactive task scenarios can be challenging or even infeasible [9, 7]. Even in situations where agents can learn to communicate, the learned protocols are usually either not human-interpretable [18] or semantically drifting from human language [23].

2.2 Human-Interpretable Communication

To address the above-mentioned challenges, several recent works propose human-interpretable communication in RL settings. Lazaridou et al.[22] leverage pre-trained task-specific language models to provide high-level guidance for natural language communication. A few other works align low-level communication tokens with human language[21, 41], or learn discrete prototype communication in a semantically meaningful space [42, 17]. But as pointed out in several studies [8, 49], the low mutual intelligibility between human language and neural agent communication makes the alignment process non-trivial. Our work is closest to [28], in which researchers alternate imitating human data via supervised learning and self-play to maximize reward in a reference game. The differences are that in [28] authors try to train neural agents for reference games in an end-to-end fashion with backpropagation, while we train MARL agents in interactive team tasks. The exploration of this research direction is still very limited, as no previous work has ever evaluated natural language communication agents within interactive task environments and for ad-hoc human-agent teams.

2.3 Language-Grounded Reinforcement Learning

Reinforcement learning is known to struggle with long-horizon problems with sparse reward signals [31]. Natural language guidance has been used to provide auxiliary rewards to improve the data efficiency and learning robustness [44]. Goyal et al.[13] use step-by-step natural language instructions provided by human annotators to construct auxiliary reward-learning modules, encouraging agents to learn from expert trajectories. Narasimhan et al.[32] research the impact of language grounding on representation learning and transfer learning of RL agents in a 2D game environment. Additional work has explored grounding RL with other formats of materials such as game manuals [14, 46] and human commands [48]. However, none of those works has ever used the communication messages as the language ground nor guided the information-sharing process in multi-agent teamwork.

The most relevant research to our proposed method is CICERO, which empirically evaluates the proposed AI agent in the complex natural language strategy game Diplomacy with human players [11]. CICERO has an RL module for strategic reasoning and a language model for generating messages. The two modules are trained separately on different datasets, namely self-play trajectories and conversation data collected from human players. The two modules in CICERO function independently, with the only connection being that the language model takes intention estimation from the planning module as input. While in our work, both action and communication are generated by individual RL agents that are trained end-to-end with a combination of RL loss and supervised learning loss.

3 Preliminaries

We formulate the problem as a decentralized partially observable Markov Decision Process [35], which can be formally defined by the tuple $(\mathcal{I}, \mathcal{S}, \mathcal{A}, \mathcal{C}, \mathcal{T}, \Omega, \mathcal{O}, \mathcal{R}, \gamma)$, where $\mathcal{I}$ is the finite set of n agents, $s \in \mathcal{S}$ is the global state space, $\mathcal{A} = \times_{i \in \mathcal{I}} \mathcal{A}_i$ is the set of actions, and $\mathcal{C} = \times_{i \in \mathcal{I}} \mathcal{C}_i$ is the set of communication messages for each of n agents. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition function that maps the current state s_t into next state s_{t+1} given the joint agent action. In our partially observable environments, each agent receives a local observation $o^i \in \Omega$ according to observation function $\mathcal{O} : \mathcal{S} \times \mathcal{C} \times \mathcal{I} \to \Omega$. Finally, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, while $\gamma \in [0, 1)$ is the discount factor. At each time stamp t, each agent i takes an action a_t^i and sends out a communication message c_t^i after receiving the partial observation of task state s_t along with all messages sent by other agents from last time stamp c_{t-1}. Each agent then receives the individual reward $r_t^i \in \mathcal{R}(s_t, a_t)$. We consider fully cooperative settings in which the reinforcement learning objective is to maximize the total expected return of all agents:

$$\max_{\pi^i : \Omega \to \mathcal{A} \times \mathcal{C}} \mathbb{E}\left[\sum_{t \in T} \sum_{i \in \mathcal{I}} \gamma^t \mathcal{R}(s_t^i, a_t^i) | a_t^i \sim \pi^i, o_t^i \sim \Omega\right] \tag{1}$$

https://doi.org/10.52202/079017-2790

We borrow the definition of language learning from [28]. We define a target language $L* \in \mathcal{L}$ that we want the agents to learn, assuming $\mathcal{L}$ is the set of natural language and $L*$ is the optimal communication language for achieving a good team performance in the specific task. Specifically, we consider a language $L \in \mathcal{L}$ to be a set of communication messages $\mathcal{C}$ which are mapped from agent observations to communication messages defined as $L : \Omega \rightarrow \mathcal{C}$. In typical RL settings, this can be thought of as the mapping between input observation vectors and English descriptions of the observation. We consider a dataset $\mathcal{D}$ consisting of $|\mathcal{D}|$ (observation, action) pairs, which comes from expert trajectories generated by LLM embodied agents using the target language $L*$. The language learning objective is to train agents to speak language $L*$ in order to collaborate with experts in ad-hoc teamwork. It is worth noting that we want the learned language to generalize to unseen examples that are not contained in $\mathcal{D}$.

To train agents that perform well on the team task and speak human-interpretable language, we need to solve a multi-objective learning problem by combining the learning signals from both environment reward and supervised dataset $\mathcal{D}$. The process of training an optimal communication-action policy can be defined as solving an optimization problem with two constraints 1) agents must learn to communicate effectively to maximize team performance and 2) agents must learn to use similar language as in the supervised dataset $\mathcal{D}$.

4 Language Grounded Multi-agent Communication

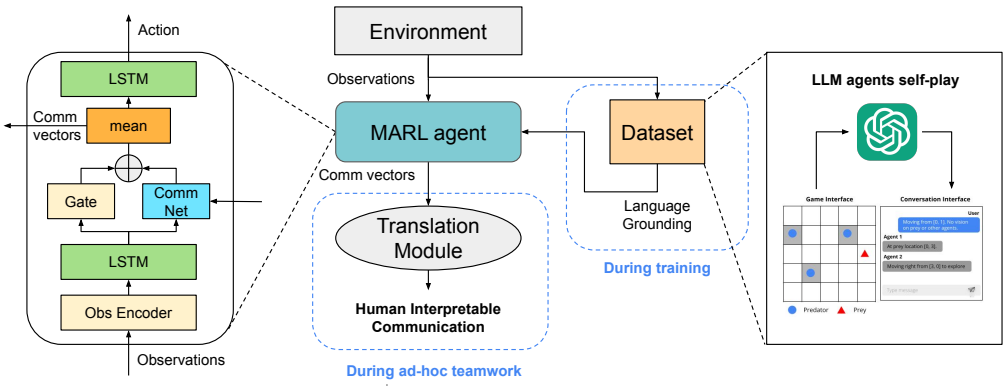

Figure 1: Illustrations of our proposed computational pipeline , LangGround. The framework consists of three modules: 1) collecting grounded communication from LLM agents, 2) aligning MARL communication with language grounds, 2) translating aligned communication vectors into natural language messages via cosine similarity matching.

In this section, we propose a computational pipeline for learning language-grounded multi-agent communication (LangGround). The general framework of LangGround is illustrated in Figure 1 consisting of two parts: collecting grounded communication from LLM agents and aligning MARL communication with language grounds.

4.1 Grounded communication from LLM agents

We use embodied LLM agents to collect samples of target language $L*$. To allow LLM agents to interact with the task environments, a text interface I is implemented following [25] to translate between abstract representations and English descriptions. Essentially, each of the n LLM agents is provided with general prompts about the team task and instructed to collaborate with others to achieve the common goal. At each time stamp t, the LLM agent i receives English descriptions of its own observation of the environment $I(o_t^i)$ which also includes communication messages from teammates in the last time stamp C_{t-1}.The LLM agent is prompted to output its next action and communication message which are then encoded into abstract $\mathcal{A}_t^i$ and $\mathcal{C}_t^i$ and used to update the task environment.

Theoretically, we construct a text environment in parallel with the actual task environment to ensure that embodied LLM agents are exposed to the equivalent information as RL agents, albeit in a different

format. The action and communication policy of LLM agents emerge from the backbone LLM, since the provided prompts do not include any explicit instructions on team coordination or communication strategy. In the results section we will confirm findings from previous literature that modern LLMs (e.g., GPT-4) is able to perform reasonably well and communicate effectively in collaborative tasks. The expert trajectories generated by LLMs are used to construct the supervised dataset $\mathcal{D}$ which maps individual agent's (observation, action) pairs to natural language communication messages. $\mathcal{D}$ is used during the training of MARL to provide supervised learning signals to align the learned communication protocols with human language. More implementation details of LLM agents and data collection can be found in the Appendix.

4.2 Multi-agent Reinforcement Learning with aligned communication

The MARL with communication pipeline is similar to IC3Net [37] in which each agent has an independent controller model to learn how and when to communicate. During each time-step, input observation o_t^i is encoded and passed into each agent's individual LSTM. The hidden state of LSTM h_t^i is then passed to the probabilistic gating function to decide whether to pass a message to other agents. A single-layer communication network transforms h_t^i into communication vector c_t^i. The mean communication vector of all agents is finally used by each agent's LSTM to produce the action a_t^i.

To shape the learned communication protocol toward human language, we introduce an additional supervised learning loss during the training of MARL. Specifically, at each time step, we sample a reference communication from the dataset based on each agent's observation and action, $D(o_t^i, a_t^i)$, representing how a human (LLM) would communicate in the same situation. We then calculate the cosine similarity between the agent communication vector c_t^i and the word embedding of the reference communication c_h. To align the communication space learned by MARL with the high-dimensional embedding space of natural language, we construct the supervised loss function as follows:

$$L_{sup} = \sum_{t \in T} \sum_{i \in \mathcal{I}} \left[1 - cos(c_t^i, D(o_t^i, a_t^i))\right] \tag{2}$$

$$D(o_t^i, a_t^i) = \begin{cases} c_h & \text{if } (o_t^i, a_t^i) \in \mathcal{D} \\ \mathbf{0} & \text{otherwise} \end{cases} \tag{3}$$

The construction of the communication message is shaped by two learning signals: 1) the reinforcement learning objective which determines useful information to share with other agents based on the policy loss gradient, and 2) the supervised learning objective which imitates the communication messages used by LLM agents in dataset $\mathcal{D}$. The total loss function is formulated as follows:

$$L = L_{RL} + \lambda L_{sup} \tag{4}$$

The hyperparameter λ is used to scale the supervised loss. Each agent's policy is optimized with backpropagation to minimize the joint loss.

5 Experiments

5.1 Environments

In this section, we evaluate our proposed method in two multi-agent collaborative tasks with varied setups and different characteristics. The first environment, Predator Prey [37], is widely used in comm-MARL research as a benchmark. We include this environment to represent team tasks that require all agents to share their partial observations for the team to form a complete picture of the task state. The second environment, Urban Search & Rescue (USAR) [25, 34], presents a more demanding challenge due to the inclusion of heterogeneous team members and the temporal dependence between their behaviors. Here, agents must communicate not only their observations but also their intentions and requests to effectively coordinate. Illustrations of the two environments are shown in Figure 1, and more details are provided in the Appendix.

 https://doi.org/10.52202/079017-2790

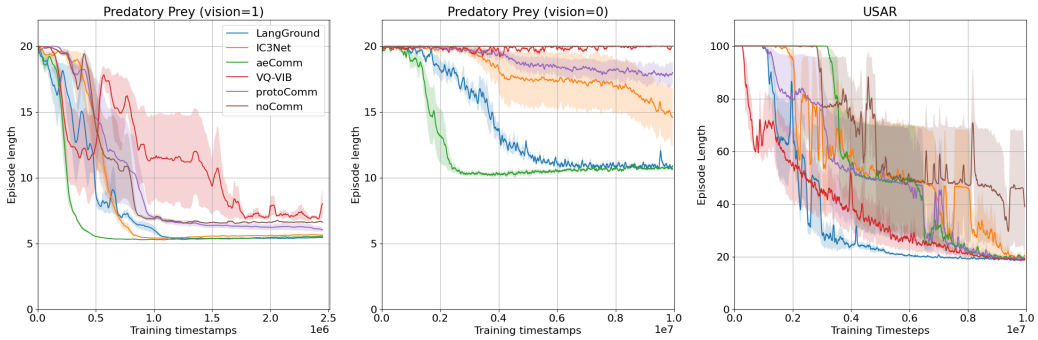

Figure 2: Learning curves of LangGround in comparison with baseline methods. The y-axis is task performance measured by the episode length until task completion, which is lower the better. The x-axis is the number of training timestamps. Shaded areas are standard errors over three random seeds.

5.2 Experiment setups

We compare our proposed pipeline LangGround against previous methods, including IC3Net [37], autoencoded communication (aeComm) [26], Vector-Quantized Variational Information Bottleneck (VQ-VIB) [40], prototype communication (protoComm) [42], and a control baseline of independent agents without communication (noComm). aeComm represents the state-of-the-art multi-agent communication methods that grounds communication by reconstructing encoded observations. It has been shown to outperform end-to-end RL methods and inductive biased methods in independent, decentralized settings. VQ-VIB and prototype-based method are representative solutions for human-interpretable communication, which learn a semantic space for discrete communication tokens and perform reasonably well in human-agent teams. Finally, IC3Net has a similar architecture to our proposed pipeline representing an ablating baseline without language grounding.

All methods are implemented with the same centralized training decentralized execution (CTDE) architecture for a fair comparison. Each agent has an observation encoder, an LSTM for action policy, a single-layer fully-connected neural network for transferring hidden states into communication messages, and a gate function for selectively sharing messages. The parameters of the action policy and obs encoder are shared during training for a more stable learning process. We use REINFORCE [45] to train both the gating function and policy network. The communication messages are continuous vectors of dimension $D = 256$.

6 Results

As for the evaluation matrices, we first consider if LangGround allows MARL agents to complete collaborative tasks successfully (i.e., task utility) and converge to a shared communication protocol quickly (i.e., data-efficiency), in comparison with other state-of-the-art methods as the baselines. Then we analyze the properties of aligned communication space such as human interpretability, topographic similarity, and zero-shot generalizability, to show how close the learned language is to the target human natural language. Finally, we evaluate the ad-hoc teamwork performance in which MARL agents must communicate and collaborate with unseen LLM teammates via natural language.

6.1 Task performance

In Figure 2, we compare the task performance of multi-agent teams using different communication methods by plotting out the average episode length during training over 3 random seeds with standard errors.

In Predator Prey vision = 1 (i.e., pp_{v1}), our method LangGround achieves a similar final performance with IC3Net and aeComm, outperforming other baselines. However, the improvement is not outstanding due to the simplicity of the task environment. In Predator Prey vision = 0 (i.e., pp_{v0}), the predator's vision range is limited to their own location making effective information sharing more important in solving this search task. As shown in the middle figure, LangGround has a

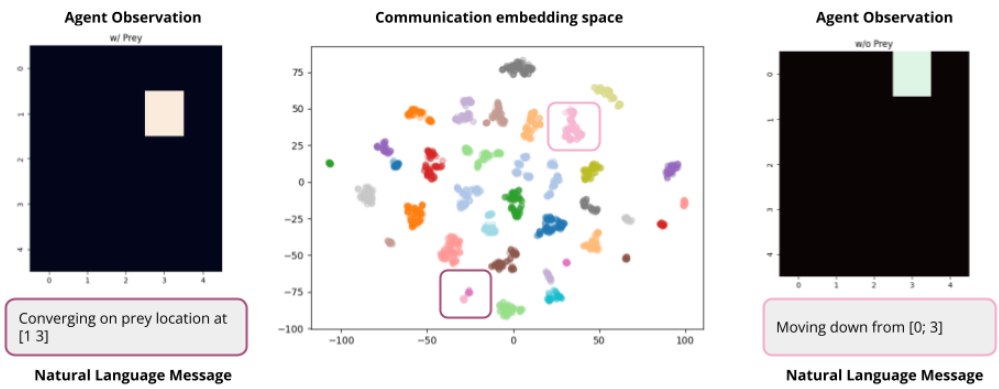

Figure 3: Learned communication embedding space. Communication vectors between agents in pp_{v0} are visualized with t-SNE and clustered with DBSCAN. Two semantically meaningful clusters are identified as examples, each corresponding to a specific agent observation. We also present the most similar reference message from dataset $\mathcal{D}$ to illustrate the alignment between the agent communication space and the human language embedding space.

comparable final performance with aeComm and outperforms other baselines. Finally, in the most challenging USAR environment, LangGround outperforms all baselines in solving the task in fewer steps with the same amount of training time-steps. In addition, language grounding also stabilizes the communication learning process such that the variance of LangGround is much smaller than other methods.

To test the performance of proposed method in scaled environments, we ran additional experiments in Predator Prey with a larger map size (i.e., pp_{v1} (10 by 10)). The learning curves of LangGround and baselines are presented in Fig 4. As shown in the figure, our method outperforms ablating baselines without language grounding (i.e., IC3Net) or without communication (i.e., noComm). This result demonstrates the benefit of introducing LangGround in stabilizing the learning process of emergent communication of MARL agents in scaled environments.

In summary, LangGround enables multi-agent teams to achieve on-par performance in comparison with SOTA multi-agent communication methods. Introducing language grounds as an auxiliary learning objective does not compromise the task utility of learned communication protocols while providing interpretability.

6.2 Aligned communication space

In addition to task utility, we are also interested in other properties of the learned communication space, such as human interpretability, topographic similarity, and zero-shot generalizability.

6.2.1 Semantically meaningful space

To evaluate whether the learned communication space is semantically meaningful, we visualize the learned communication embedding space by clustering message vectors sent by agents over 100 evaluation episodes in pp_{v0}, following [26]. The high dimensional vectors are reduced to a two dimension space via t-SNE [43], and clustered with DBSCAN [10]. As shown in Figure 3, agent communication messages can be clustered into several classifications with explicit meanings associated with agent observation from the environment. For example, the pink cluster on the right side corresponds to the situation where the agent locates in coordinates (0, 3) without vision of prey. We can look up from dataset $\mathcal{D}$ for the reference message that has the most similar word embedding with agent communication vectors in the pink cluster. The reference message (i.e., "moving down from (0, 3)") accurately refers to the agent observation, indicating the learned communication space is semantically meaningful and highly aligned with natural language embedding space.

https://doi.org/10.52202/079017-2790

Env	Δ Cos sim	Δ Bleu score
pp_{v1}	0.82±0.02	0.52±0.03
pp_{v0}	0.81±0.03	0.45±0.12
$USAR$	0.79±0.12	0.42±0.04

Table 1: Similarity gain w/ LangGround

Grounding	Cos sim	Bleu score
100%	0.775	0.633
75%	0.667	0.498
50%	0.304	0.308
25%	0.188	0.224

Table 2: Zero-shot generalization evaluation results on pp_{v1} (10 by 10).

6.2.2 Human interpretability

Given the goal of aligning agent communication with human language, it is intuitive to evaluate the human interpretability of language-grounded agent communication. We use the offline dataset $\mathcal{D}$ as the reference to calculate the similarity between communication messages shared by LangGround agents and LLM agents in same situations. Given 100 evaluation episodes in pp_{v0}, we calculate 1) the cosine similarity between word embedding and agent communication vectors, and 2) BLEU score between natural language messages and reference messages in $\mathcal{D}$ with the most similar word embedding as agent communication vectors. The results are shown in Table 1, demonstrating that LangGround achieves significant gains in both metrics, cosine similarity and BLEU score, compared to baselines without alignment. These measures for other baseline methods would be equivalent to random chance because none of them are grounded with language during training, and hence we do not compute their performance on these metrics.

6.3 Ad-hoc Teamwork

The ultimate goal of our proposed pipeline is to facilitate ad-hoc teamwork between unseen agents without pre-coordination. Here, we propose two experiments to evaluate the zero-shot generalizability and ad-hoc collaboration capability of our trained agents.

6.3.1 Zero-shot generalizability

One prerequisite of ad-hoc teamwork is the ability to communicate about unseen states to their teammates. We evaluate this capability by removing a subset of prey spawn locations from environment initialization of pp_{v0} and training LangGround agents from scratch. In this condition, agents would neither be exposed to nor receive any language grounding for those situations during training. During the evaluation, we record the communication messages used by agents in those unseen situations and compare them with ground truth communication from dataset $\mathcal{D}$. Results show that agents are still able to complete tasks when the prey spawns in those 4 unseen locations. As shown in Table 3, the communication messages agents used to refer to those unseen locations are similar to natural language sentences generated by LLMs.

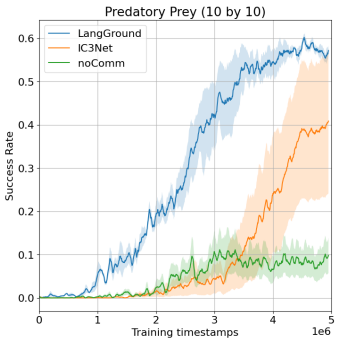

Figure 4: Team performance of LangGround and baselines on Predator Prey with 10 by 10 map and vision range of 1.

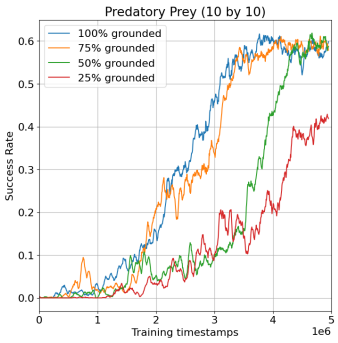

Figure 5: Team performance of LangGround agent with different levels of language grounding on pp_{v1} (10 by 10).

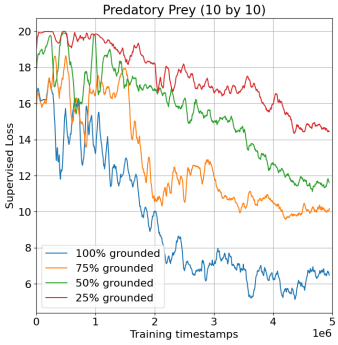

Figure 6: Communication alignment of LangGround agent with different levels of language grounding on pp_{v1} (10 by 10).

Table 3: Zero-shot generalizability in pp_{v0}

Prey Loc	Cos sim	Bleu score	Example message
(1,1)	0.81	0.41	Moving up to converge on prey location at (1,0) for capture
(1,3)	0.81	0.27	Converging on prey location at (1,3)
(3,1)	0.82	0.51	Moving up toward prey location at (3,1)
(3,3)	0.78	0.72	Converging on prey location at (3,3)

Additionally, we train LangGround agents on Predator Prey (10 by 10) but only provide language grounding in a subset of states (i.e., 25%, 50%, 75%, 100%). The learning curves of LangGround agents with different levels of language grounding are presented in Fig. 5 and Fig. 6. As shown in the figures, the more grounded states, the better the team performance, as well as the better the communication alignment. Table 2 shows similar results of communication alignment in un-grounded states during evaluation. The first column refers to percentage of grounding data used during training. The other two columns are alignment measurements when agents encounter un-grounded states during evaluation.

To summarize, these findings confirm the alignment between the agent communication space and the human language word embedding space in zero-shot conditions. More importantly, we show that LangGround is not merely a memorization of one-to-one mapping between observations and communications but also shapes the continuous communication space in a semantically meaningful way. We could ground the agent communication space with the word embedding space of human language on a limited number of instances, and expect LangGround to output interpretable messages in un-grounded states via topographic similarity (discussed more in the Appendix) of the aligned space. In practice, this assumption depends on many factors such as the coverage of offline dataset, number of grounded states, dimension of the communication space, scale of the problem, etc. The above experiments illustrate the impact of language grounding percentage, and we leave further investigation to future work.

6.3.2 Ad-hoc teamwork between MARL and LLM agents

Finally, we evaluate the performance of our agents to work with unseen teammates in ad-hoc teamwork settings. Ad-hoc teamwork refers to situations where agents collaborate with unseen teammates without pre-coordination. In this work, we use embodied LLM agents to emulate human behaviors in human-agent ad-hoc teams. Teams with 2 MARL agents and 1 LLM agent were evaluated on 8 episodes over 3 random seeds, resulting in a total of 24 evaluation episodes per condition. Ad-hoc teamwork performance is measured by the number of steps taken to complete the task; therefore, lower is better. Means and standard deviations of each condition are reported in Table 4.

Table 4: Ad-hoc teamwork performance (lower is better)

Team composition	pp_{v1}	pp_{v0}	$USAR$
LangGround	4.3 ± 1.20	10.9 ± 4.53	22.0 ± 4.24
LLM	6.8 ± 5.20	11.6 ± 5.30	15.9 ± 3.37
LangGround + LLM	$\mathbf{8.5 \pm 5.76}$	$\mathbf{15.5 \pm 4.80}$	23.2 ± 10.61
aeComm + LLM	10.3 ± 6.46	17.5 ± 4.60	20.3 ± 9.07
noComm + LLM	10.6 ± 5.73	20.0 ± 0.00	32.4 ± 13.47

We find that 1) homogeneous teams (i.e., LangGround and LLM) achieve better performance than ad-hoc teams because of their common understanding of both action and communication. As those agents are either co-trained together or duplicates of the same network, they form a stable strategy for team coordination and information sharing. Since ad-hoc teams (e.g., LangGround + LLM) were not trained together nor speak the same language, their decreased performance is expected. 2) The ad-hoc team performance of LangGround agents is better than noComm and aeComm agents in at least two out of three evaluation scenarios. Because aeComm is not aligned with human language, it serves as a baseline with a coordinated action policy and a random communication policy. The advantage of our method over aeComm and noComm merely comes from effective information sharing via the

 https://doi.org/10.52202/079017-2790

aligned language with unseen teammates. The empirical evidence presented in this section confirms the application of our method in ad-hoc teamwork.

7 Discussion

In this work, we developed a novel computational pipeline to enhance the capabilities of MARL agents to interact with unseen teammates in ad-hoc teamwork scenarios. Our approach aligns the communication space of MARL agents with an embedding space of human natural language by grounding agent communications on synthetic data generated by embodied LLMs in interactive teamwork scenarios.

Through extensive evaluations of the learned communication protocols, we observed a trade-off between utility and informativeness. According to the Information Bottleneck principle [40], informativeness corresponds to how well a language can be understood in task-agnostic situations, while utility corresponds to the degree to which a language is optimized for solving a specific task. By introducing the additional supervised learning signal, our method pushes the trade-off toward informativeness compared to traditional comm-MARL methods that merely optimize for utility. This partially explains the "inconsistent" patterns we observe in Figure 2 and Table 4 across different task scenarios. In relatively easy tasks such as predator-prey, the learned communication is more optimized for informativeness, aligning better with human language and generalizing better in ad-hoc teamwork. In more challenging tasks such as USAR, the learned communication is more shaped toward task utility, resulting in faster convergence but less interpretability to unseen teammates.

Additionally, we found that introducing language grounding does not compromise task performance but even accelerates the emergence of communication, unlike the results reported in previous literature, where jointly optimizing communication reconstruction loss and RL loss leads to a drop in task performance [26, 27]. The main reason for this contradiction is that our dataset $\mathcal{D}$ consists of expert trajectories from LLM embodied agents with a well-established grounding on the task. Therefore, the language grounding loss not only shapes the communication but also guides the action policy of MARL agents by rewarding behavior cloning and providing semantic representations of input observations.

It worth noting that LangGround is an extremely flexible pipeline with most of the modules being interchangeable, such as the word embedding model, base MARL-comm model, message dataset source, and the translation module. This allows us to allocate appropriate tasks to LLMs and RL respectively, considering LLMs are known to have good linguistic capabilities (e.g., describing) while struggle with formal reasoning (e.g., planning). While we use embodied LLM agents to collect grounding communication datasets for LangGround as explained in Section 4, These datasets can also come from rule-based agents or human participants, as long as they show effective communication in solving collaborative tasks. Because only communication messages are used, LLM agents' hallucinations and does not impact MARL's task performance directly. In more complicated scenarios in which communication is beyond describing observation and action, we could still expect LLMs to generate reasonable outputs. Compared to alternative methods in embodied agents where LLMs must make correct action planning at every timestamp, it is more feasible to collect semantically meaningful messages from either LLMs or any other sources.

We believe our work would benefit the broader society for the following reasons. This research provides empirical evidence of linguistic principles during language evolution among neural agents, which might provide insights for broader research communities, including computational linguistics, cognitive science, and social psychology. The usage of embodied LLM agents as interactive simulacra of human team behaviors has a broad impact since it has potential applications in social science and may deepen our understanding of modern LLMs. Most importantly, our proposed pipeline takes initial steps in enabling artificial agents to communicate and collaborate with humans via natural language, shedding light on the broad research direction of Human-centered AI.

As for future directions, we plan to evaluate our proposed pipeline in more complicated task environments at scale and experiment with different selections of MARL algorithms, backbone LLMs, and word embeddings. Particularly, we plan to replace the use of a static dataset $\mathcal{D}$ by querying LLMs online during the training of MARL. This may allow us to capture complex information exchanged among team members in addition to individual observations, such as beliefs, intents, and requests.

Acknowledgments and Disclosure of Funding

We would like to thank Mycal Tucker, Seth Karten and the original authors of IC3Net for providing public access to their code base that were instrumental in this research. We also appreciate the helpful discussions with Lu Wen, Yikang Gui, and Vasanth Reddy that contributed to the development of this work.

References

[1] Akshat Agarwal, Swaminathan Gurumurthy, Vasu Sharma, Mike Lewis, and Katia Sycara. Community regularization of visually-grounded dialog. In Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, pages 1042–1050, 2019.

[2] Henry Brighton and Simon Kirby. Understanding linguistic evolution by visualizing the emergence of topographic mappings. Artificial life, 12(2):229–242, 2006.

[3] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[4] Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning. In International Conference on Machine Learning, pages 3676–3713. PMLR, 2023.

[5] Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. Anti-efficient encoding in emergent communication. Advances in Neural Information Processing Systems, 32, 2019.

[6] Rahma Chaabouni, Eugene Kharitonov, Emmanuel Dupoux, and Marco Baroni. Communicating artificial neural networks develop efficient color-naming systems. Proceedings of the National Academy of Sciences, 118(12):e2016569118, 2021.

[7] Rahma Chaabouni, Florian Strub, Florent Altché, Eugene Tarassov, Corentin Tallec, Elnaz Davoodi, Kory Wallace Mathewson, Olivier Tieleman, Angeliki Lazaridou, and Bilal Piot. Emergent communication at scale. In International conference on learning representations, 2021.

[8] Roberto Dessì, Eleonora Gualdoni, Francesca Franzon, Gemma Boleda, and Marco Baroni. Communication breakdown: On the low mutual intelligibility between human and neural captioning. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 7998–8007, 2022.

[9] Tom Eccles, Yoram Bachrach, Guy Lever, Angeliki Lazaridou, and Thore Graepel. Biases for emergent communication in multi-agent reinforcement learning. Advances in neural information processing systems, 32, 2019.

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd, volume 96, pages 226–231, 1996.

[11] Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. Science, 378(6624):1067–1074, 2022.

[12] Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. Advances in neural information processing systems, 29, 2016.

[13] Prasoon Goyal, Scott Niekum, and Raymond J Mooney. Using natural language for reward shaping in reinforcement learning. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, pages 2385–2391, 2019.

 https://doi.org/10.52202/079017-2790

[14] Austin W Hanjie, Victor Y Zhong, and Karthik Narasimhan. Grounding language to entities and dynamics for generalization in reinforcement learning. In International Conference on Machine Learning, pages 4051–4062. PMLR, 2021.

[15] Seth Karten, Siva Kailas, Huao Li, and Katia Sycara. On the role of emergent communication for social learning in multi-agent reinforcement learning. arXiv preprint arXiv:2302.14276, 2023.

[16] Seth Karten and Katia Sycara. Intent-grounded compositional communication through mutual information in multi-agent teams. In Workshop on Decision Making in Multi-Agent Systems at International Conference on Intelligent Robots and Systems (IROS), 2022.

[17] Seth Karten, Mycal Tucker, Huao Li, Siva Kailas, Michael Lewis, and Katia Sycara. Interpretable learned emergent communication for human-agent teams. IEEE Transactions on Cognitive and Developmental Systems, 2023.

[18] Satwik Kottur, José Moura, Stefan Lee, and Dhruv Batra. Natural language does not emerge 'naturally'in multi-agent dialog. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2962–2967, 2017.

[19] Angeliki Lazaridou and Marco Baroni. Emergent multi-agent communication in the deep learning era. arXiv preprint arXiv:2006.02419, 2020.

[20] Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. Emergence of linguistic communication from referential games with symbolic and pixel input. In International Conference on Learning Representations, 2018.

[21] Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. arXiv preprint arXiv:1612.07182, 2016.

[22] Angeliki Lazaridou, Anna Potapenko, and Olivier Tieleman. Multi-agent communication meets natural language: Synergies between functional and structural language learning. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7663–7674, 2020.

[23] Jason Lee, Kyunghyun Cho, and Douwe Kiela. Countering language drift via visual grounding. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4385–4395, 2019.

[24] David Lewis. Convention: A philosophical study. John Wiley & Sons, 2008.

[25] Huao Li, Yu Chong, Simon Stepputtis, Joseph P Campbell, Dana Hughes, Charles Lewis, and Katia Sycara. Theory of mind for multi-agent collaboration via large language models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 180–192, 2023.

[26] Toru Lin, Jacob Huh, Christopher Stauffer, Ser Nam Lim, and Phillip Isola. Learning to ground multi-agent communication with autoencoders. Advances in Neural Information Processing Systems, 34:15230–15242, 2021.

[27] Yat Long Lo, Biswa Sengupta, Jakob Nicolaus Foerster, and Michael Noukhovitch. Learning multi-agent communication with contrastive learning. In The Twelfth International Conference on Learning Representations, 2023.

[28] Ryan Lowe, Abhinav Gupta, Jakob Foerster, Douwe Kiela, and Joelle Pineau. On the interaction between supervision and self-play in emergent communication. In International Conference on Learning Representations, 2019.

[29] Kyle Mahowald, Anna A Ivanova, Idan A Blank, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. Dissociating language and thought in large language models: a cognitive perspective. arXiv preprint arXiv:2301.06627, 2023.

[30] Reuth Mirsky, William Macke, Andy Wang, Harel Yedidsion, and Peter Stone. A penny for your thoughts: The value of communication in ad hoc teamwork. International Joint Conference on Artificial Intelligence, 2020.

[31] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. nature, 518(7540):529–533, 2015.

[32] Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. Grounding language for transfer in deep reinforcement learning. Journal of Artificial Intelligence Research, 63:849–874, 2018.

[33] Yaru Niu, Rohan R Paleja, and Matthew C Gombolay. Multi-agent graph-attention communication and teaming. In AAMAS, volume 21, page 20th, 2021.

[34] Ini Oguntola, Dana Hughes, and Katia Sycara. Deep interpretable models of theory of mind. In 2021 30th IEEE international conference on robot & human interactive communication (RO-MAN), pages 657–664. IEEE, 2021.

[35] Frans A Oliehoek, Christopher Amato, et al. A concise introduction to decentralized POMDPs, volume 1. Springer, 2016.

[36] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, pages 1–22, 2023.

[37] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In International Conference on Learning Representations, 2018.

[38] Sainbayar Sukhbaatar, Rob Fergus, et al. Learning multiagent communication with backpropagation. Advances in neural information processing systems, 29, 2016.

[39] Weihao Tan, Wentao Zhang, Shanqi Liu, Longtao Zheng, Xinrun Wang, and Bo An. True knowledge comes from practice: Aligning llms with embodied environments via reinforcement learning. arXiv preprint arXiv:2401.14151, 2024.

[40] Mycal Tucker, Roger Levy, Julie A Shah, and Noga Zaslavsky. Trading off utility, informativeness, and complexity in emergent communication. Advances in neural information processing systems, 35:22214–22228, 2022.

[41] Mycal Tucker, Roger P Levy, Julie Shah, and Noga Zaslavsky. Generalization and translatability in emergent communication via informational constraints. In NeurIPS 2022 Workshop on Information-Theoretic Principles in Cognitive Systems, 2022.

[42] Mycal Tucker, Huao Li, Siddharth Agrawal, Dana Hughes, Katia Sycara, Michael Lewis, and Julie A Shah. Emergent discrete communication in semantic spaces. Advances in Neural Information Processing Systems, 34:10574–10586, 2021.

[43] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. Journal of machine learning research, 9(11), 2008.

[44] Nicholas Waytowich, Sean L Barton, Vernon Lawhern, Ethan Stump, and Garrett Warnell. Grounding natural language commands to starcraft ii game states for narration-guided reinforcement learning. In Artificial intelligence and machine learning for multi-domain operations applications, volume 11006, pages 267–276. SPIE, 2019.

[45] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3):229–256, 1992.

[46] Yue Wu, Yewen Fan, Paul Pu Liang, Amos Azaria, Yuanzhi Li, and Tom M Mitchell. Read and reap the rewards: Learning to play atari with the help of instruction manuals. Advances in Neural Information Processing Systems, 36, 2024.

 https://doi.org/10.52202/079017-2790

[47] Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. Language models meet world models: Embodied experiences enhance language models. Advances in neural information processing systems, 36, 2024.

[48] Shusheng Xu, Huaijie Wang, and Yi Wu. Grounded reinforcement learning: Learning to win the game under human commands. Advances in Neural Information Processing Systems, 35:7504–7519, 2022.

[49] Shunyu Yao, Mo Yu, Yang Zhang, Karthik R Narasimhan, Joshua B Tenenbaum, and Chuang Gan. Linking emergent and natural languages via corpus transfer. arXiv preprint arXiv:2203.13344, 2022.

[50] Changxi Zhu, Mehdi Dastani, and Shihan Wang. A survey of multi-agent deep reinforcement learning with communication. Autonomous Agents and Multi-Agent Systems, 38(1):4, 2024.

A Implementation details

A.1 Embodied LLM agents

Large language models are prompted to interact with the task environments in team tasks. We implement embodied LLM agents based on the pipeline proposed in [25], where agents are augmented with explicit belief state and communication for better team collaboration capability. Each agent keeps a memory of his own observations from the environment and communication messages from other team members. Exact prompts can be found in the code within the supplementary materials. The design principles are that we only provide general rules about the task environments without explicitly instructing them on any coordination or communication strategy. We attempt to minimize the influence of prompt engineering to ensure the seamless applicability of our approach in diverse environments. In our language grounding data collection and ad-hoc teamwork experiments, we use OpenAI's API to call gpt-4-0125-preview as the backbone pre-trained model and set the temperature parameter to 0 to ensure consistent outputs.

Example prompts for LLM agents in the $USAR$ environment are provided below:

> Welcome to our interactive text game! In this game, you'll assume the role of a specialist on a search and rescue team. Alongside two other players, you'll navigate a five-room environment with a mission to defuse five hidden bombs.
>
> **The Map**: Imagine a network of rooms represented by a connected graph where each node corresponds to a room, and the edges between nodes depict hallways. The rooms are numbered 0, 3, 6, 5, and 8. Room 0 is connected to all other rooms. Room 5 shares a hallway with room 6. Room 3 is linked to room 8. And room 8 is also connected with room 6. You can only travel to adjacent, directly connected rooms at each turn.
>
> **The Challenge**: Scattered among these rooms are five bombs, each coded with different phases represented by colors. To defuse them, you'll need to use the correct wire-cutting tools in the correct sequence. There are one-phase, two-phase, and three-phase bombs, needing 1, 2, or 3 color-coded tool applications in sequence to disarm. For instance, a bomb with a red-green phase sequence requires the red tool first, then the green one. Points are awarded based on the number of tools used for defusing a bomb, with each tool use worth 10 points. Your task is to maximize the team score as soon as possible. The challenge is that the bomb locations and sequences are unknown to players at the start.
>
> **Tools**: Each player is equipped with two color-coded wire cutters. As player Alpha, you have red and green tools, player Bravo wields green and blue, and player Charlie possesses blue and red.
>
> **Actions**: Each round, you can opt to do one of the following: 1) Move to an adjacent room, 2) Inspect a bomb's phase sequence in your current room, or 3) Apply your wire cutters to a bomb in the current room.
>
> **Communications**: In addition to selecting an action to take from the above list, you can also send communication message texts to both of your teammates in each round. The message text you sent will be shared with both of your teammates in their observation in the next round.
>
> **Observation**: While you can only see what's in your current room and read text messages from teammates. You'll also be informed of the current round number, team score and the current location of your teammates. Your teammates have the same observability as you. They will not be able to know your action and its consequences unless you explicitly communicate.
>
> To facilitate our interaction, reply your action selection and communication messages in this fixed format: Action selection: Your action. Message to Team: "Your Message". To move to an adjacent room, say: 'Move to Room X'. To inspect the sequence of a bomb in your current room, say: 'Inspect Bomb'. To apply a wire cutter tool, say: 'Apply X Tool'. Remember, your replies must adhere strictly to these rules. Feel free to ask clarifying questions if needed. I'll supply the necessary information as we progress. Are you ready to take on this explosive challenge?

https://doi.org/10.52202/079017-2790

Example interactions between LLM agents and environments are provided below:

> **Env**: You are now in Room 3 with Bomb 5. What is your next action?
> **Alpha**: Action selection: Move to Room 5. Message to Team: "I am moving to Room 5 t help Charlie with Bomb 3.
> **Env**: You can not directly move to Room 5 because it is not adjacent to your current location; Room 3. Consider taking a detour to another room first and then move to your destination.

Example communication among LLM agents in $USAR$:

> **Alpha**: I am inspecting Bomb 1 in Room 0. Bravo; please move to Room 3. Charlie; please move to Room 5.
> **Bravo**: I am moving to Room 3 as instructed. Alpha; please defuse Bomb 1 in Room 0 using the red tool.
> **Charlie**: I am moving to Room 5 to search for more bombs.

A.2 LangGround and baseline MARL agents

The implementation of LangGround and baseline MARL methods is based on open-sourced codebase provided by the original paper authors [40, 16, 42, 37, 34, 26]. Agents were trained with the same setup and hyperparameters after empirical searching. The batch size is 500, and the number of update iterations in an epoch is 10. Training on pp_{v0} and $USAR$ takes 2000 epochs and $1e7$ timestamps, which takes about 4 hours to complete. Training on pp_{v1} takes 500 epochs and $2.5e6$ timestamps, which takes about 1.5 hours to complete. We use a learning rate of 0.0001 for $USAR$ and 0.001 for pp. MARL agent's action policy is an LSTM with a hidden layer of size 256. Communication vectors are exchanged one round at each timestamp. The supervised learning weight λ is 1 in pp and 10 in $USAR$. VQ-VIB and prototype communication agents were allowed to use 58 different discrete tokens to share information during teamwork. All experiments were conducted on a machine with a 14-core Intel(R) Core(TM) i9-12900H CPU and 64GB memory.

B Environment details

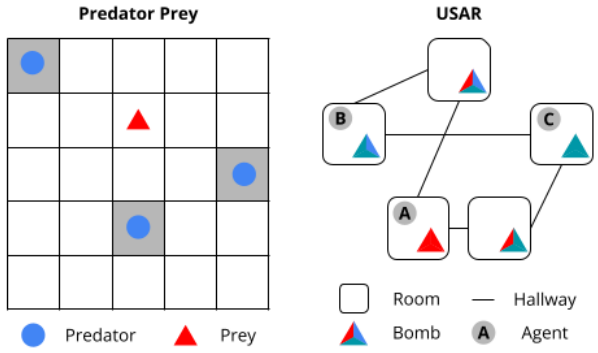

Figure 7: Illustrations of the evaluation environments. Predator Prey is a gridworld environment conceptualizing a team of predators with partial observation trying to search for a static prey. The task goal is for all predators to reach the prey location within the time limit. USAR simulates a team of specialists searching for and defusing bombs in an unknown environment. Because each specialist has the unique capability of defusing bombs in different colors, the team must coordinate to complete the task efficiently.

B.1 Predator Prey

In this task, n predators with a limited range of vision v need to search for stationary prey on an x by x grid-world environment. Each predator receives a positive reward upon reaching the prey location.

Each episode ends when all predators reach the prey or exceed the maximum number of steps T. The initial locations of predators and the prey might spawn anywhere on the map. At each timestamp, the predator agent receives a partial observation of v by v grids around its own location and may select a movement action to navigate through the map. Since this is a collaborative task, predators must learn to communicate their partial observations to allow for optimal navigation of the team. We consider Predator Prey to be a more challenging task since it has a higher-dimensional observation space and a more complex action space [26]. Through this environment, we aim to demonstrate that aligning agents' communication with human language is a straightforward yet effective method of grounding their communication with task observations.

B.2 USAR

The USAR task environment is designed to simulate the collaborative and problem-solving dynamics of a search and rescue mission. Three agents (i.e., Alpha, Bravo, and Charlie) need to collaborate in locating and defusing color-coded bombs hidden in an unexplored environment. Each bomb has unique phase sequences in m colors, which are not revealed until inspected by agents. Agents start with different colored cutters and must use them in the correct sequence to defuse bombs. The environment is represented as a graph, where each of the n rooms is a node, and the hallways connecting them are edges. At each timestamp, every agent can choose from three different types of actions: moving to one of the n rooms, inspecting a bomb's sequence in the current room, or utilizing one of the m wire-cutters. The size of the action space depends on the problem scale (i.e., $n + m + 1$). Agents' observations are limited to their current room's contents and agent status. The team is rewarded $10*x$ points when an x-phase bomb is successfully defused. An episode ends when the team has defused all bombs or exceeded the time limit. This task is designed to force team coordination since each team member has unique observations and capabilities. For example, each agent only has a subset of wire cutters and must coordinate with other teammates to defuse bombs with multiple phases. Therefore, an effective communication protocol is required for efficient information sharing and team synchronization.

B.3 Environment configurations

For the predator and prey environment, we use a map size of 5 by 5 with 3 predators and 1 prey. The predator's range of vision is manipulated to be either 0 or 1 to create two variants of the task environment. In the situation of vision = 0, predators cannot observe the prey until they jump onto the same location. We set the maximum episode length to 20 based on previous research [17]. The USAR environment comprises five rooms ($n = 5$) and five bombs, including two single-phase, two double-phase, and one triple-phase bombs. The bomb phase might have three different colors ($m = 3$). Each of the 3 agents spawns with 2 different wire cutters, forcing the team to collaborate in defusing bombs with multiple phases. Each successfully defused bomb awards the team 10 points per processed phase, resulting in 90 as the maximum score per mission. We set the maximum episode length to 100 based on previous research [25].

B.4 Text game interface

The initial task environments of Predator Prey and USAR are implemented for MARL agents based on Gym API [3]. To facilitate interaction between LLM-based agents with the environment, we implement a rule-based text interface for each task. At each timestamp, LLM agents sequentially interact with the environment, receiving observations and performing actions via natural language interaction. Additionally, they are allowed to broadcast communication messages in natural language which are appended with the observation text and sent to all team members in the next round. It is worth noting that LLM agents receive equivalent information as MARL agents, limited to individual agent's partial observation.

The text interface facilitates communication between the game engine and the language model agents by converting game state observations into natural language descriptions and mapping agent responses back to valid game actions. To generate observations, the interface extracts relevant state features from the game engine, such as the current round number, cumulative team score, action feedback, visible objects, and communication messages from other agents. It then populates predefined sentence templates with these extracted features to produce a structured natural language

 https://doi.org/10.52202/079017-2790

description of the current game state. Action encoding relies on keyword matching, as the language models are instructed to format their responses using specific keywords and structures. The interface scans the agent's response for these predefined keywords and maps them to corresponding game actions. In cases where an agent's response is invalid or ambiguous, such as attempting to perform an action in an incorrect location, the interface generates an informative error message based on predefined rules and templates. For instance, if an agent attempts to inspect a non-existent bomb, the interface might respond with the following error message: "There is no bomb in the current location, Room X, for you to inspect." This targeted feedback helps the agents refine their actions to comply with the game's rules and constraints.

C Data collection details

C.1 LangGround dataset

In order to construct dataset $\mathcal{D}$, we collected expert trajectories from embodied LLM agents powered by GPT-4 in interactive task scenarios. As shown in Table 4, teams consisting of pure LLM agents perform reasonably well in comparison to MARL methods. Therefore, we believe their action and communication policy can be used in guiding MARL agents. In $USAR$, we collected 50 episodes resulting in 2550 pairs of (observation, action) and communication messages of individual agents. The number of data pairs is 1893 for pp_{v0} and 2493 for pp_{v1}, respectively. To facilitate the alignment of agent communication space and natural language, we use OpenAI's word embedding api (i.e., text-embedding-3-large) to translate each natural language message into a high-dimensional vector with the same dimension (i.e., $D = 256$) as agent communication vectors.

C.2 Ad-hoc teamwork

Due to the restrictions of resources and time, we use embodied LLM agents to emulate human behaviors in human-agent teams. We match 2 MARL agents with 1 unseen LLM agent in a team and ask them to complete the collaborative task in predator-prey and USAR environments. The LLM agent is powered by GPT-4-turbo and prompted to output action selection and communication messages given observation inputs, similar to the data collection process introduced in Section4. The Gym environment is wrapped with a text interface to decode observations into English descriptions and encode LLM agent's output into concrete action selection. Both MARL agents and LLM agents interact with the same task environment in sequence. Natural language communication messages from LLMs are embedded using OpenAI's word embedding API and sent to MARL agents. The communication vectors from MARL agents are translated to English sentences via cosine similarity matching in dataset $\mathcal{D}$.

D Additional Experiment Details

D.1 Topographic similarity

The topographic similarity is defined by the correlation between object distances in the observation space and their associated signal distances in the communication space [2]. This property is usually associated with language compositionality and ease of generalization. The intuition behind this measure is that agents should emit similar communication messages given semantically similar observations. We calculate this measure following [20], based on agent trajectories collected from 100 evaluation episodes in pp_{v0}. We first calculate 1) the cosine similarity between all pairs of communication vectors, and 2) the Euclidean distance between all pairs of agent locations. Then, we calculate the negative Spearman correlation ρ as the measure of topographic similarity. Table 5 indicates that our method (i.e., LangGround) results in the highest topographic similarity $\rho = 0.67$ among all other baselines, exhibiting a relatively more similar property as human language.

D.2 Additional ablation study

Here we run additional ablation study to attribute reinforcement learning signal and supervised learning signal to the agent's action and communication output in the multi-objective optimization problem.

Methods	Topo Sim ρ
LangGround	**0.67±0.07**
IC3Net	0.54±0.14
aeComm	0.37±0.05
protoComm	0.35±0.35

Table 5: Topographic similarity in pp_{v0}

Because the LangGround agent is trained end-to-end with a combination of RL and SL loss and uses the intermediate hidden state of its policy as the communication vector, it is very hard to separate the reasoning processes of action and communication. However, we could provide indirect evidence to prove that RL and SL jointly contribute to both the agent's action and communication.

The MARL-comm agent uses a gating function to learn whether to communicate at specific timestamps. We could ablate this function to see its impact on team performance. As shown in the Fig. 8, removing the gating function harms the performance of LangGround more than IC3Net. This means both RL and SL signals influence the content and timing of LangGround communication.

In addition, we could manipulate the weight of supervised learning loss, i.e. λ in function 4, to illustrate the contribution of RL and SL signals. As shown in Fig. 9 and 10 in the PDF, λ matters for both task performance and supervised loss. If the SL loss is weighted too high, the LangGround agent cannot optimize its policy in completing the task. While if the RL loss is weighted too high, the LangGround agent cannot align its communication with human language. This result aligns with our claim that RL optimizes the communication for task utility and SL optimizes the communication for alignment.

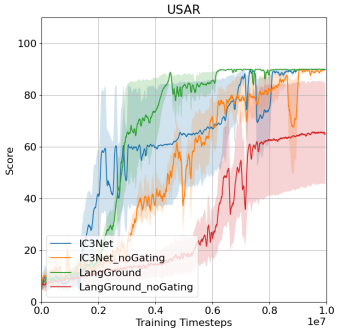

Figure 8: Ablation study of gating function and language grounding.

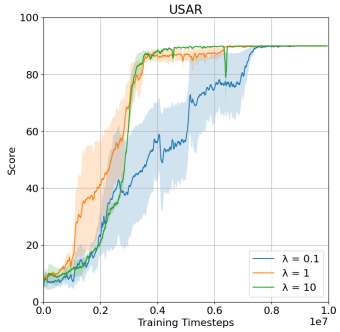

Figure 9: Impact of supervised learning loss weight λ on team performance.

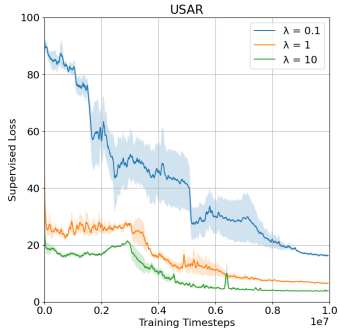

Figure 10: Impact of SL loss weight λ on communication alignment.

https://doi.org/10.52202/079017-2790

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We make reasonable claims based on the empirical results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in the discussion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe our proposed pipeline clearly in Section 4.In the Appendix, we provide implementation details of MARL agents and reproducibility details (e.g. model API, prompts, parameters) for LLM agents.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

https://doi.org/10.52202/079017-2790

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We submit our code in the supplementary materials for the review purpose. We will make the code public available upon paper acceptance, following the NeurIPS code and data submission guidelines.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide those details in both the main text and appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We provide error bars over evaluation episodes and random seeds in the main text for all tables and figures.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We reveal the computational resource used for experiments in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No Code of Ethics are violated.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the potential societal impact of our work in the discussion section.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

https://doi.org/10.52202/079017-2790

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not release any data or models that have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original papers that produced the open-source code for baseline methods.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide detailed instructions and documentations with the code.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

https://doi.org/10.52202/079017-2790

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.