

# RLupus

*Cooperation through emergent communication in The Werewolf social deduction game*

Nicolo' Brandizzi<sup>a</sup>, Davide Grossi<sup>b</sup>, Luca Iocchi<sup>a</sup>

<sup>a</sup> *Dipartimento di Ingegneria Informatica, Automatica e Gestionale,  
Sapienza University of Rome, Italy*

*E-mail: {brandizzi, iocchi}@diag.uniroma1.it*

<sup>b</sup> *Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence,  
University of Groningen, The Netherlands.*

*Amsterdam Center for Law and Economics and Institute for Logic, Language and Computation,  
University of Amsterdam, The Netherlands.*

*E-mail: d.grossi@rug.nl*

**Abstract.** This paper focuses on the emergence of communication to support cooperation in environments modeled as social deduction games (SDG), that are games where players communicate freely to deduce each others' hidden intentions. We first state the problem by giving a general formalization of SDG and a possible solution framework based on reinforcement learning. Next, we focus on a specific SDG, known as *The Werewolf*, and study if and how various forms of communication influence the outcome of the game. Experimental results show that introducing a communication signal greatly increases the winning chances of a class of players. We also study the effect of the signal's length and range on the overall performance showing a non-linear relationship.

**Keywords:** Multi-agent systems, Social deduction games, Deep reinforcement learning, Emergent communication

## 1. Introduction

Social deduction games (SGDs) are games characterized by the confrontation between two or more parties, one of which is usually seen as an evil faction. The other parties must deduce the real intention of the latter, seeing through lies and deceptions. While the details of these games may change, free communication—i.e., players can communicate with each other with no limitations—is a common aspect for them all.

In artificial settings, communication would lead to increased complexity in the environment, both on the user side, where engineers are tasked to design an expressive and robust syntax [Genesereth et al., 1992, Finin et al., 1994, O’Brien and Nicol, 1998], and on the artificial players who have to learn the syntax and the meaning of the available words. For shallow players, this task quickly becomes unfeasible. A common solution is to define a game-specific language containing communication semantics. This language is devised by usually injecting some expert knowledge into the system. The language becomes then part of the game, providing new available communication actions augmenting the agents’ choices in the decision-making process. Instead of developing a language for the agents, in this article, we propose another approach enabling players to use a free communication mechanism.

*Context of the paper* Games arising from social interaction have been extensively studied within the multi-agent reinforcement learning (MARL) literature. Indeed, MARL systems have been successfully used to model a wide variety of social systems found in nature and modern society, from food-collective ants systems to complex sharing of information in social networks [Buşoniu et al., 2010]. In these settings, the agents’ behavior can be cooperative, competitive or a mix of the two. In [Buşoniu et al., 2010] a study of these different kinds of settings and the algorithms associated with them is pursued while [Tan, 1993, Liang et al., 2020] focuses on the difference between competitive and cooperative agents.

Although MARL has been used for a wide variety of applications, the reinforcement learning paradigm may not scale up easily in complex multi-agent systems. For this reason, RL has been integrated with deep neural networks (DeepRL,

DRL) [Schmidhuber, 2015]. This allows RL to scale to problems that were previously intractable, such as playing video games using pixels as input [Hosu and Rebedea, 2016]. Shortly after [Tampuu et al., 2017], this approach has been applied to multi-agent systems to study the complex emergent behavior of multiple agents interacting with each other to reach a goal. Although DRL is a well-established field of study, agent interaction via actions or communication remains a challenging problem.

This research area is strictly tied to the study of complex behavior arising from the interaction of simple agents; these aspects are mainly studied through the usage of structured game systems. In their work, Baker et al. [Baker et al., 2020] showed how a few simple rules from the *Hide’n Seek* game can generate complicated behaviors to the point of exploiting environmental errors to their advantages. Along the same line, Leibo et al. [Leibo et al., 2019] carried out extensive analysis on the problem of autotutorials and non-stationary learning in multi-agent deep reinforcement learning (MADRL); they pointed out how the interaction of competitive agents can culminate in an endless cycle of counter-strategies due to the non-stationarity of the environment.

In particular, one such complex behavior consists in the emergence of communication between cooperative agents. Recent works investigate this aspect in various environments varying from joined image captioning [Graesser et al., 2019], to negotiation [Cao et al., 2018] and simulated pointing games [Mordatch and Abbeel, 2018]. Our paper is a contribution to this line of research, focusing on communication in social deduction games.

*Paper contribution and outline* The paper makes two main contributions:

- a formal description of social deduction games coupled with a general reinforcement learning solution framework, which allows for free communication among agents without requiring to provide game-specific knowledge;
- an analysis of the performance obtained through learned communication behaviors in an instance of the above class of games: *The Werewolf*<sup>1</sup> social deduction game.

---

<sup>1</sup>Also known as *Lupus in Fabula*.

The paper is organized as follows. An overview of the related work is provided in Section 2. In Section 3, the problem statement is formalized together with the general RL framework. Section 5 describes the *Werewolf* game instance in detail, defining both the game logic and the actual implementation. This game provides a fit ground to study language emergence since its whole system is based on communication, indeed it is the subject of an annual *AiWolf* contest in Japan. The experimental settings and results are reported and commented in Section 6 together with the comparison between our work and [Kajiwara et al., 2014]. Finally, a discussion is provided in Section 7.

The code is available at [https://github.com/nicofirst1/rl\\_werewolf](https://github.com/nicofirst1/rl_werewolf).

## 2. Related Work

Our work relies on the findings coming from the social interaction field of psychology, coupled with the multi-agent systems and reinforcement learning.

*Social Deduction Games.* Social deduction games have been studied in the broader context of social interactions [Eger and Martens, 2018]. In particular, they have been used to study the role of rationality in inter-personal interaction [Colman, 2003], analyze the different forms of social mechanics [Consalvo, 2011], and research the role of communal topology [Abramson and Kuperman, 2001], however, the deduction part of these games has been neglected.

Indeed, in their work [Chan et al., 2009], the authors give a mathematical formulation for a general social game in order to simplify the way to design such games; however, no specific formulation is given for deduction games.

On the other hand, [Wiseman and Lewis, 2019] study the most influential information source in social deduction games and concludes that the interactions that occurred prior to the game are regarded as most important to the player. Although this approach is reasonable in the context of acquaintances, no result is given for games in which the playing parties do not know each other.

In all these works, the goal is centered around the social interaction between players. In our work, we shift the focus to finding an optimal policy to improve the performances of a party.

*Multi-agent Deep RL* Applying RL paradigms to find an optimal policy for multi-agent systems is a well-established line of work [Busoniu et al., 2008, Tan, 1993, Shoham et al., 2003]. In recent years deep neural networks (DNN) have been used to solve complex tasks such as playing Atari games [Hosu and Rebedea, 2016], cooperating in Hide-and-seek [Baker et al., 2020] and competing with humans on strategic games [Vinyals et al., 2019, Silver et al., 2016].

A common paradigm for training RL agents in a deep setting is policy gradient methods, [Sutton et al., 1999, Silver et al., 2014] where the gradient of a parametrized policy is used to guide the agent in the direction of maximal expected reward. In our work, we leverage a particular instance of these methods called Proximal Policy Optimization [Schulman et al., 2017].

In all such cases, the complexity of the environment coupled with a DNN generated unexpected behaviors that are usually counter-intuitive for humans but achieve greater performances in the task at hand. In our work, we leverage this aspect and further study how the coordination between agents varies under different communication instances.

*Emergent Communication.* This phenomenon has been exploited in the newly born field of emergent communication [Wagner et al., 2003], where agents are given the choice to use a communication channel to achieve a common goal.

The workshop on Emergent Communication (Emecom) includes many publications in the field of natural language processing [Li et al., 2020] strictly tied to MADRL [Lazaridou and Baroni, 2020, Liang et al., 2020, Foerster et al., 2016] and social deduction games as an environment. Standard games for this line of research are the Task & Talk [Kottur et al., 2017], which is centered around dialogue, on the other hand, the Pointing Game [Mordatch and Abbeel, 2018] grounds the communication into natural image processing.

Another instance of these settings is *The Werewolf* game, where the players find themselves split into two opposite groups in a partially known environment. This game has gained increased popularity in the field of cooperation through emergent communication, especially in Japan, where the annual *AiWolf* contest [Bi and Tanaka, 2016, Nakamura et al., 2016, Hirata et al., 2016, Katagami et al., 2014] sees artificial agents competing with

and against human players to win the game with fixed language syntax. In particular, [Wang and Kaneko, 2018] set up a 5-player game with additional roles and use a Deep Q-Network to determine who to trust or kill.

In our work, we choose *The Werewolf* as an instance of the general SDG framework. However, our implementation differs from the ones in the *AiWolf contest*, the closest being [Kajiwara et al., 2014], where the authors use Q-learning to study the winning chances of the villagers in a game with 16 players, divided into 14 villagers and 2 werewolves. Indeed we drop the hand-coded syntax and let the players develop their own communication by defining some general attributes of the channel.

### 3. Problem Statement

Social deduction games (SDG) are characterized by the presence of a number of opposing parties

$$P^{(1)}, P^{(2)}, \dots, P^{(m)}$$

(typically  $m = 2$ ), each containing a finite number  $n(k)$  of players, which may differ per party. For the sake of conciseness, we drop the  $k$  dependency and denote the number of players in a party as simply  $n$ :

$$P^{(k)} = \{p_1^{(k)}, p_2^{(k)}, \dots, p_n^{(k)}\}$$

with  $1 \leq k \leq m$ . We denote by  $N = \bigcup_{1 \leq k \leq m} P^{(k)}$  the set of all players.

The game evolves as a sequence of actions performed by the players of the parties, typically in turns. The effect of these actions contributes to the definition of the game score.

The goal of each party is to prevail over the others by performing suitable social behaviors, including, for example, leveraging other players by means of bluffs and lies. These deceitful methods are the base for any SDG and force every player to perform a deductive analysis on the member of the other parties. In general we can identify two types of goals in SDGs:

- An agent-based *micro-goal*, which is the main factor steering the agent’s behavior, either in isolation (in competitive environments) or together with other agents’ goals (in cooperative environments).

- A party-based *macro-goal*, expressing the aligned interests of the members of the same party that, combined, make up the party goal.

During the execution of the game, agents can communicate among them, either implicitly or explicitly. We define explicit communication as the act of sharing information for the sole purpose of affecting other agents’ mental states. On the other hand, implicit communication regards all those actions that carry more than one meaning, e.g., guiding someone to a goal. Decisions about how, when, and what to communicate are critical choices for the game’s success.

We thus distinguish two categories of actions performed by the players: 1) *game actions*, that are actions affecting the evolution of the game, 2) *communication actions* that are actions affecting only the mental state (i.e., the knowledge state) of the players. In this formalization, we consider only forms of explicit communication, while studying forms of implicit communication is left as future work.

With the previous assumptions, in this paper, we consider the formalization of an SGD with the following elements:

1. the action set  $\mathcal{A} = \mathcal{G} \cup \mathcal{C}$  is made of two separate components:
  - a finite set of possible game actions  $\mathcal{G}$  the elements of which we will denote with  $g$ .
  - a set of unidirectional communication actions  $C_{i,j}(b)$  intended to convey some information  $b \in \mathcal{B}$ <sup>2</sup> between two players:

$$C_{i,j}(b) : p_j \rightarrow p_i \quad \forall p_j, p_i \quad j \neq i \in N$$

2. the state set  $\mathcal{O} = \mathcal{E}^N \times \mathcal{W}^N \times \mathcal{V}$ , with  $N$  being the total number of players, built out of three elements<sup>3</sup>:
  - a set of agent’s features  $\mathcal{E}$  representing the game situation of each agent (typically visible to all other agents),
  - a set of agent’s internal states  $\mathcal{W}$  (e.g., representations of beliefs not visible to other agents) .

<sup>2</sup>A possibly infinite set of all possible signals.

<sup>3</sup>We consider a setup where all the agents  $N$  choose an action simultaneously.

- a set of environment states  $\mathcal{V}$  that are common for all the agents (i.e., independent from the agent states).
3. an environment  $\mathcal{S}$  implementing the game logic.  $\mathcal{S}$  can be seen as a function taking as input the agents actions and yielding a new state obtained as the result of execution of such actions

$$\mathcal{S} : \mathcal{O} \times \mathcal{A}^N \rightarrow \mathcal{O} \quad (1)$$

Notice that, while it may be relatively easy to formalize the specifications of game actions, for example, in terms of pre-conditions and post-conditions using action representation formalisms, it is less clear how to formalize communication actions since it would require an explicit model of agents' knowledge. For modeling this kind of communication actions, the use of typical action formalisms is not straightforward. For example, they may need to be extended with epistemic operators [Fagin et al., 1995].

#### 4. General solution framework

This article studies social deduction games that can be formalized as a multi-agent (deep) reinforcement learning (RL) scenario. In such scenarios, each party has to choose an optimal strategy or policy <sup>4</sup> (i.e., an optimal assignment from states to actions) to maximize the game score. As already mentioned, a particular feature of SDGs is the presence of communication actions and the need to choose optimal communications among the players within a party.

The problem of learning optimal policies in multi-agent games is indeed well known, and many solutions are available. However, when communication actions are involved, using artificial intelligence techniques to make optimal decisions about how, what, and when to communicate is still a challenging problem under investigation, and fewer research works are available.

The advantage in defining a solution based on RL is that it does not require an explicit model

of the transition function for communication actions. In other words, optimal behaviors can be computed without associating a semantic meaning to the communication actions. While this feature can be considered not desirable for some kinds of applications (e.g., for mixed human-AI teams), it is very convenient for AI teams based on RL that can learn their own communication language to win the game. Indeed AI agents can effectively learn a communication language without making the semantics of communication explicit. The explainability of learned communication actions is left as future work.

*Action Policy* We consider a single game turn  $t$  as a sequence of  $k+1$  steps, where the first  $k$  steps are associated to only communication actions  $c \in \mathcal{C}$  while the last step is a game action  $g \in \mathcal{G}$ . We denote each time step as  $\mathcal{O}_t^j \quad \forall j \in [0, k+1]$ .

Now, we can define a policy  $\pi$  that uses the information from  $\mathcal{O}_t^j$  in order to choose an action as follows:

$$\begin{aligned} \pi(\mathcal{O}_t^j) &\in \mathcal{C} \quad \forall j \leq k \\ \pi(\mathcal{O}_t^{k+1}) &\in \mathcal{G} \end{aligned}$$

Note that in some cases it can be useful to compute two distinct policies  $\pi_C ; \pi_G$ , one for communication and one for the game actions.

#### 5. R-Lupus framework

In this section, we present the *RLupus* framework for the Werewolf social deduction game in which we apply the above formalization. In this specific context, we aim to study if and how various forms of communication can influence the outcome of the game, in which only one party is able to learn while the other one has a fixed, hand-coded policy.

Under the same circumstances, the hypothesis is that the agents who can communicate will perform much better than those not allowed to exchange information. Moreover, we speculate that different communication settings will have diverse influences on the amount of coordination among the agents and the outcome of the game.

In the following sub-sections, we give a brief introduction on *The Werewolf* game logic, a description of the RL environment with all its compo-

<sup>4</sup>Although the coordinated behavior favors the parties' macro-goal, the actual policy works on the players' micro-goal level.

nents and the policies used for the players in the game.

### 5.1. The Werewolf Game

Werewolf is a social deduction game modeling conflicts between two groups in a partially known environment. In its easiest version, the game sees two groups ( $M = 2$ ), villagers  $P^{(v)}$  and werewolves  $P^{(w)}$  where  $P^{(v)} > P^{(w)} + 1$ . The wolves know exactly the identity of each player, while the villagers are certain exclusively about their role and the number of werewolves. In an open setup, an additional moderator is needed to coordinate the players.

The game is divided into two phases: night and day, interleaving each other.

The game ends either when the villagers execute the last werewolf, or there is an even number of both roles. The latter case implies the wolves winning since the execution phase can be stalled, thus taking away the only possibility for villagers to kill the wolves.

Table 1

RLupus: Multi-channel metrics. The first column reports the type of **comm**(unication)channel regarding the **SignalLength** and the **SignalRange**. The next four show the metrics values for villagers winning rate, suicide rare, number of days elapsed and accordance rate

| Comm           | Win Vil     | Suicide      | Days       | Accord      |
|----------------|-------------|--------------|------------|-------------|
| <b>0SL</b>     | 0.044       | 0.086        | 1.55       | 0.47        |
| <b>1SL-2SR</b> | 0.19        | 0.078        | 1.58       | 0.47        |
| <b>1SL-9SR</b> | 0.21        | 0.078        | 1.58       | 0.47        |
| <b>9SL-2SR</b> | <b>0.45</b> | <b>0.067</b> | <b>1.9</b> | <b>0.47</b> |
| <b>9SL-9SR</b> | 0.19        | 0.077        | 1.58       | 0.46        |

### 5.2. Reinforcement Learning

Before describing the *RLupus* environment, a brief description of the Reinforcement Learning paradigm must be introduced.

In the canonical RL environment, the problem satisfies the Markov property<sup>5</sup> so it can be formulated as a Markov decision process (MDP). An MDP is defined by a tuple of five elements  $(\mathcal{O}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$  defined as follows:

- A finite set of observations called the observation space  $\mathcal{O}$
- A finite set of actions, the action space  $\mathcal{A}$
- A transition model  $\mathcal{T}_a(m_t, m_{t+1})$  defining the probability to transition from state  $m_t$  to a new state  $m_{t+1}$  given action  $a$ .
- $\mathcal{R}(m_t, m_{t+1})$  the reward associated to the previous transition.

At each timestep  $t$ , an RL agents receives a state  $m_t$  from a dynamic environment  $\mathcal{O}$ . The agent then selects an action  $a_t$  from the action space  $\mathcal{A}$  following a policy  $\pi(a_t|m_t)$ . The action is processed by  $\mathcal{S}$ , equation 1, which transitions to the next state  $m_{t+1}$  and yields a reward  $r_t$  to the agent. This feedback loop continues until the agent reaches a terminal state and restarts.

The agent aims to maximize the expected discounted reward given by:

$$R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$$

Where  $\gamma \in (0, 1]$  is the discount factor governing the importance of future rewards.

### 5.3. RLupus Environment

Dealing with RL implies the presence of an action and state set; the latter are referred to as action space and observation space, which are presented in the following section<sup>6</sup>.

*Action Space* As mentioned in Section 5.4, the Werewolf uses a policy  $\pi(\mathcal{O}_t^j)$  in order to choose both the communication and game actions. In a way, the agent can be seen as performing a game  $g_t$  and communication  $c_t$  action simultaneously.

Indeed the action space is divided into two parts:

- *Target*: The target  $g_t$  is an integer in range  $g_t \in [0, N - 1]$ , where  $N$  is the number of players. Its intended usage is to allow players to vote for other players during the game. The range of possible values never changes during the execution; instead, illegal actions, such as voting for dead players, are filtered out later in the model.

<sup>6</sup>For the sake of conciseness some formulations are omitted from the body of the paper. The interested reader can refer to the Appendix A.

<sup>5</sup>The future depends only on the current state and action.



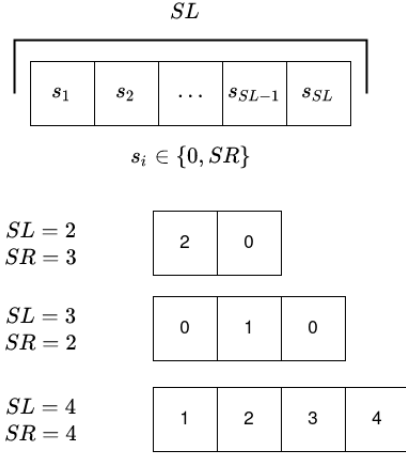


Fig. 1. Signal vector with general formulation as well as specific values for  $SL$  and  $SR$ .

- *Signal*: The signal vector  $c_t$  (see Figure 1) is defined by two integer values: its length  $SL \in [0, \infty[$  which can be any value from zero (no communication) to an arbitrary large integer; its range  $SR \in [2, N]$  that defines the number of possible values it can have.  $SR$  bounded below by 2, since a signal with only one possible value would be considered a static vector carrying no information; the upper bound  $N$  comes from the necessity for the signal to be embedded with the target. Both are used to define the valid space for communication before training.

*Observation Space* The observation space characterizes what the agents perceive in the environment. This space includes both other agents’ actions and information about the environment<sup>7</sup>.

*Transition Model* The transition model fully depends on the target action; that is, the environment is independent of the communication between agents. Indeed  $\mathcal{S}$  switches between night and day at every timestep  $t$  and removes dead players from the environment.

*Rewards* The rewards, or penalties, are the core of the environment and determine how the players interact, learn, and develop new strategies; an agent’s main goal is to take actions that will max-

imize the expected reward.

Following our formalization, the environment is responsible for delivering a reward to each player<sup>8</sup>.

*Metrics* To measure the changes in the agent behavior, the following normalized metrics are logged:

- *Suicide*: the number of times an agent votes for itself during an execution phase.
- *Wins*: the villagers’ wins are plotted in the normalized range of values.
- *Average days*: average number of days before a match ends.
- *Accord*: This value represents, on average, the percentage of agents that vote for the same target during the two execution phases.

#### 5.4. Policies

An agent’s policy defines the behavior of a player during the game. In this environment, there are two kinds of policies: *trainable* policies use custom algorithms to collect experience and learn to maximize the reward; *static* policies are hard-coded behaviors that are used to guarantee a fixed baseline trough out the evaluation.

In this work, we assign static policies to opponent players (werewolves) and training policies to AI agents (villagers) learning how to win the game.

*Static policies for werewolves.* Static policies are reserved for the werewolf agents; their aim is to allow a baseline evaluation of the villager learning. Since wolves are more likely to win in a completely random environment, applying such policies is enough to prove the development of new strategies for the villagers if the winning rates are to change significantly.

Three policy are implemented:

- *Random Target*: chooses a random non-dead player among the villagers during the execution phase.
- *Random Target Unite*: this policy targets the same player for every wolf, both during day and night execution; this allows the werewolf to dominate the day execution phase with random villagers.

<sup>7</sup>The complete observation space is described in Appendix A.1.1.

<sup>8</sup>The complete set of rewards is described in Appendix A.1.2.

Table 2

RLupus: single/no channel metrics. The Design Choice part of the table shows which kind of policy Random, Unite or Revenge has been used in relation to the communication, while the Results half present the metric' values

| Design Choice  |     |     |     | Results     |              |              |             |
|----------------|-----|-----|-----|-------------|--------------|--------------|-------------|
| Comm           | Rnd | Unt | Rvg | Vil Win     | Accord       | Suicide      | Days        |
| <b>OSL</b>     | X   |     |     | 0.044       | 0.478        | 0.086        | 1.55        |
| <b>OSL</b>     |     | X   |     | 0.03        | <b>0.695</b> | <b>0.059</b> | <b>1.5</b>  |
| <b>OSL</b>     |     |     | X   | <b>0.12</b> | 0.482        | 0.078        | 1.64        |
| <b>1SL-2SR</b> | X   |     |     | 0.19        | 0.47         | 0.078        | 1.58        |
| <b>1SL-2SR</b> |     | X   |     | 0.08        | <b>0.685</b> | <b>0.055</b> | <b>1.52</b> |
| <b>1SL-2SR</b> |     |     | X   | <b>0.4</b>  | 0.479        | 0.065        | 1.9         |

- *Revenge Target*: with this policy, the wolves will either vote randomly or target a villager who previously voted for a wolf.

*Trainable policies for villagers.* In our work, we chose Proximal Policy Optimization (PPO) as a training algorithm for its widespread success in multi-agent environment [Wei et al., 2019, Guan et al., 2020], but future research could also focus on other algorithms such as TRPO [Schulman et al., 2015] or MADDPG [Lowe et al., 2017]. PPO uses a surrogate loss function to keep the difference between the old and the new policy within a safe range<sup>9</sup>. The model is a simple fully connected network with an LSTM cell<sup>10</sup>.

*Learning Werewolves.* Up until now, only villager agents have been able to learn from experience, while the werewolf behave according to a static policy. Here, we address the possibility of an environment where both villagers and werewolves are able to learn. Having multiple agents learning simultaneously void the stationary assumption that is necessary for optimality in RL. The presence of multiple villagers already invalidates this assumption, but the coordination between them alleviates the problem. Introducing an adversarial set of learning agents would cause an increased complexity that would cloud the goal of this paper, which is to study the emergence of a language between agents.

<sup>9</sup>More information on the policy loss are available in Appendix A.1.3.

<sup>10</sup>Given the partial observability of the problem, we found the LSTM as a suitable element to approximate the state space.

## 6. Results

Following, an analysis of the results for both nine (Section 6.1) and twenty-one (Section 6.2) players is given. In both cases, a baseline setting with no communication is compared with multi-channel communication to show the increased performance reported using the metrics in Section 5.3.

Moreover, for the nine players instance, results for the additional *revenge* and *unite* policies are reported against the *random* one.

Finally, in Section 6.3, a summary for a setting with 16 players is given for both the *AiWolf* environment [Kajiwara et al., 2014] and the *RLupus* one.

### 6.1. Nine Players

A game with nine players is relatively short but not trivial for the villagers. Indeed, in a completely random environment, they have a probability of 3.12% to win<sup>11</sup>.

Since the random policy is believed to hold the least expert knowledge about the game environment, we present the results for this policy only in the next section. On the other hand, Section 6.1.2 also shows the results for the *revenge* and *unite* policies.

#### 6.1.1. Multi-channel communication

Table 1 presents the results (columns) obtained with different forms of communication (rows), with different communication settings (SL=signal length, SR = signal range). From the table, it is clear that any form of communication improves

<sup>11</sup>See Appendix A.2 for the complete estimation.



over the non-communication setting (0SL). Moreover, it shows how the bit communication (2SR) performs much better than the full ranged one (9SR). Indeed, with the addition of just one bit of communication, the villagers can perform as well as the full extended communication (9SL-9SR). In fact, for the settings where  $SR = 9$ , increasing the channel length  $SL$  has the only effect of speeding up the convergence by a factor of circa 25% for each increase.

On the other hand, the bit communication, independently from the channel length, provides generally better results. This leads to the conclusion that the two parameters dictating the change in the communication channel are not equally influential when it comes to the agent’s learning.

On a final note, the average time to train the agents was 6 hours on a single GPU<sup>12</sup> machine.

### 6.1.2. Policies comparison

Table 2 reports results also related to the werewolves policies (Rnd = Random, Unt = Unite, Rvg = Revenge).

When no communication is available (0SL), the accord value is almost identical for both the revenge and the random policy and much higher for the unite one, as previously anticipated.

The number of days reports a similar trend being smaller in the unite settings where the game ends sooner. Finally, the revenge winning rate reaches a much greater value than the other policies, 12%. The motivation being the simplicity of the policy itself, i.e., the wolves can not hide behind purely random action anymore and are not strong enough to drive the majority of the votes toward a villager.

Following the considerations for the three policies and bit communication ( $SL = 1$   $SR = 2$ ) referred to Table 2:

- *Random*: the villager’s winning rate reaches 20% ; 4.5 times more than the previous condition and 6.5 times the theoretical winning rate.
- *Unite*: as in the previous case, the coordination is much greater. Indeed the villagers are able to increase their winning rate by a factor of  $\times 3$ . This result alone proves that introducing a limited communication channel can

greatly favor the outcome even in the most disadvantageous setting.

- *Revenge*: again, the revenge policy is the easiest to spot for a trained agent reaching a winning rate of 40%.

## 6.2. Twenty-one players

Unlike what was studied in the previous section, the twenty-one players (21P) environment has more room for the villagers to win in a completely random setting. Indeed, by building a tabular representation of the expanded tree (shown in Table 3), the reader can see how the total villagers’ winning possibilities increased to 11.62% in a completely random environment.

Table 3  
Mapping between outcomes and probabilities.

| Outcome      | Prob. % | Leaves | Total % |
|--------------|---------|--------|---------|
| 0-12         | 0.029   | 1      | 11.62   |
| 0-10         | 0.143   | 4      |         |
| 0-8          | 0.447   | 10     |         |
| 0-6          | 1.162   | 20     |         |
| 0-4          | 2.819   | 35     |         |
| 0-2          | 7.02    | 56     |         |
| 1-1          | 21.06   | 56     | 88.38   |
| 2-2          | 27.965  | 21     |         |
| 3-3          | 26.017  | 6      |         |
| 4-4          | 26.017  | 1      |         |
| <b>Total</b> | 1.0     | 210    | 1.0     |

### 6.2.1. Multi channel communication

A comparison among all the multi-channel settings is reported in Table 4.

According to the previous section’s findings, the bit communication (9SL-2SR) seems to perform better than the full-ranged one (\*SL-21SR) both in terms of winning ratio and suicides.

On the other hand, the 1SL-2SR instance performs worse than the no communication one; this anomaly can be attributed to an incorrect environment exploration. As mentioned in Table 3, the expanded tree size is greater than the instance with nine players; thus, the algorithm can get more easily stuck in a local minimum. However, every other setting with a communication channel has a higher winning rate than the one without; thus, one could confidently say that the communication

<sup>12</sup>Nvidia Geforce GTX 1080 Ti.

signal is indeed helping the agents exploring the possible branches better than in other cases. For these reasons, we believe this outcome shows how the shape of the communication signal can improve the agents’ capacity to explore the environment and thus should be further investigated in future work.

Table 4  
RLupus: 21 Player metrics.

| Comm             | Win         | Suicide     | Days       | Accord      |
|------------------|-------------|-------------|------------|-------------|
| <b>OSL</b>       | 0.42        | 0.072       | 7.84       | 0.56        |
| <b>1SL-2SR</b>   | 0.25        | 0.075       | 7.74       | <b>0.57</b> |
| <b>9SL-2SR</b>   | <b>0.98</b> | <b>0.04</b> | <b>7.3</b> | 0.56        |
| <b>21SL-2SR</b>  | 0.94        | 0.05        | 7.6        | 0.56        |
| <b>1SL-21SR</b>  | 0.72        | 0.062       | 8          | 0.56        |
| <b>21SL-21SR</b> | 0.61        | 0.066       | 7.9        | 0.56        |

### 6.3. Sixteen Players

As previously mentioned, Kajiwara et al. [Kajiwara et al., 2014] presented the AiWolf framework, implementing a well-defined semantic<sup>13</sup> constrained to be both easy to use for the artificial players and understandable by humans. They extended the agents’ adaptive capabilities by adopting a Deep Q-network architecture and reported an increase in the villagers’ winning rate when they are the only ones capable of learning. Although their methodology is similar to ours, their work is not directly comparable to RLupus because the two frameworks represent the game in different ways. The different set-ups lead to related but orthogonal findings, which we discuss in this section.

We trained an environment with 16 players (2 werewolves and 14 villagers) and reported our results in Table 5, together with the results of the AiWolf framework.

It is interesting to observe the increase in performance due to the learning process. Indeed, the table shows baseline results without learning in the two frameworks and results obtained with RL.

In comparison with AiWolf, it can be seen how the OSL setting in RLupus already improves the winning chances by 33% reaching 90.2% and

Table 5

Winning rate for [Kajiwara et al., 2014] and our method in both baseline and active learning.

| Design      |                  | Win Vil |        |
|-------------|------------------|---------|--------|
|             |                  | AiWolf  | RLupus |
| No Learning | AiWolf baseline  | 38.6    | /      |
|             | RLupus baseline  | /       | 56.9   |
| Learning    | DQN / hand-coded | 52.9    | /      |
|             | PPO / OSL        | /       | 90.2   |
|             | PPO / 1SL-2SR    | /       | 98.4   |

adding a single bit communication results in almost perfect victory on the villagers’ side (98.4%).

## 7. Conclusions and Future Work

In this paper, we have defined a formalism for social deduction games, in which communication is an essential part of the game together with the allowed actions. On top of that, a general resolution framework based on reinforcement learning was defined and applied to the *The Werewolf* SDG by studying how various forms of communication influenced the outcomes of a match.

As shown by the experimental results, the introduction of different forms of communication greatly increases the agents’ performance (villagers). In particular, we observed that a Boolean signal range is preferred to an integer one. The reason for this is unclear. We speculate it may lay in the duality of the roles of the game.

Moreover, we found how much of the villagers’ winning rate is determined by the agents not voting for themselves while keeping the accord value to a maximum. This translates into better coordination, which is possible only when there is a sufficient amount of communication present in the environment.

Also, we noticed that there is no linear map between the amount of communication permitted, i.e., *SL* and *SR*, and the overall performance. Indeed there seems to be an optimal combination of the two, which depends mainly on the signal length.

We conclude by identifying three directions in which our work could be extended.

<sup>13</sup>More details about the AiWolf framework are available in Appendix A.3.

*Model of SDGs* One possible line of research consists of studying an instance of an SDG where the communication cannot be intrinsically tied to the action space, i.e., multi-step communication games such as Task & Talk [Kottur et al., 2017]. Alternatively, one could decide to depart from the deep part of the RL resolution framework and choose an SDG instance whose environment can be optically solved with standard reinforcement methods, e.g., Pointing game [Mordatch and Abbeel, 2018].

*The Werewolf* On the other hand, various possibilities arise from the study of The Werewolf game. One such could be the analysis of the language used for communication to highlight potential patterns, given that performance alone should not be considered a valid metric for the study of emergent communication [Lowe et al., 2019]. Moreover, one could extend the RLupus environment either by adding other roles, for example, the medium and the witch, or using normalized continuous vectors for the communication channel, which allow for the backpropagation of errors directly through the communication channel [Foerster et al., 2016]<sup>14</sup>. Finally, a deeper analysis could be applied to understand the impact of the communication parameters (SR and SL) on the metrics of the system and the overall performance.

*Human-machine coordination* Coordination among artificial agents is a key engineering challenge: from task allocation [Vytelingum et al., 2010, de Weerd et al., 2012], knowledge management [Wu, 2001], distributed constraint optimization problems [Modi et al., 2005] to multi-robot SLAM [Zhou and Roumeliotis, 2006, Thrun and Liu, 2005] and language models [Bengio et al., 2003, Vaswani et al., 2017]. And even more challenging is coordination between artificial and human agents. The study of emergent communication in SDGs could provide useful novel insights for the development of more efficient coordination among artificial agents and between artificial and human agents.

<sup>14</sup>On the other hand, [Jang et al., 2017] and [Maddison et al., 2017] use a Gumbel approximation to backpropagate the error through a discrete distribution.

## References

- [Abramson and Kuperman, 2001] Abramson, G. and Kuperman, M. (2001). Social games in a social network. *Physical Review E*, 63(3):030901.
- [Ahmed et al., 2019] Ahmed, Z., Roux, N. L., Norouzi, M., and Schuurmans, D. (2019). Understanding the impact of entropy on policy optimization. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 151–160. PMLR.
- [Baker et al., 2020] Baker, B., Kanitscheider, I., Markov, T. M., Wu, Y., Powell, G., McGrew, B., and Mordatch, I. (2020). Emergent tool use from multi-agent autocurricula. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- [Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *The journal of machine learning research*, 3:1137–1155.
- [Bi and Tanaka, 2016] Bi, X. and Tanaka, T. (2016). Human-side strategies in the werewolf game against the stealth werewolf strategy. In *International Conference on Computers and Games*, pages 93–102. Springer.
- [Busoniu et al., 2008] Busoniu, L., Babuska, R., and De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172.
- [Buşoniu et al., 2010] Buşoniu, L., Babuška, R., and De Schutter, B. (2010). Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*, pages 183–221. Springer.
- [Cao et al., 2018] Cao, K., Lazaridou, A., Lanctot, M., Leibo, J. Z., Tuyls, K., and Clark, S. (2018). Emergent communication through negotiation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.
- [Chan et al., 2009] Chan, K. T., King, I., and Yuen, M.-C. (2009). Mathematical modeling of social games. In *2009 International Conference on Computational Science and Engineering*, volume 4, pages 1205–1210. IEEE.
- [Colman, 2003] Colman, A. M. (2003). Cooperation, psychological game theory, and limitations of rationality in social interaction. *Behavioral and brain sciences*, 26(2):139–153.
- [Consalvo, 2011] Consalvo, M. (2011). Using your friends: Social mechanics in social games. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 188–195.
- [de Weerd et al., 2012] de Weerd, M. M., Zhang, Y., and Klos, T. (2012). Multiagent task allocation in social networks. *Autonomous Agents and Multi-Agent Systems*, 25(1):46–86.
- [Eger and Martens, 2018] Eger, M. and Martens, C. (2018). Keeping the story straight: A comparison of commitment strategies for a social deduction game. In *Four-*

- teenth Artificial Intelligence and Interactive Digital Entertainment Conference.*
- [Fagin et al., 1995] Fagin, R., Halpern, J. Y., Moses, Y., and Vardi, M. Y. (1995). *Reasoning About Knowledge*. MIT Press.
- [Finin et al., 1994] Finin, T., Fritzon, R., McKay, D., and McEntire, R. (1994). Kqml as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*, pages 456–463.
- [Foerster et al., 2016] Foerster, J., Assael, I. A., De Freitas, N., and Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. In *Advances in neural information processing systems*, pages 2137–2145.
- [Genesereth et al., 1992] Genesereth, M. R., Fikes, R. E., et al. (1992). Knowledge interchange format-version 3.0: reference manual.
- [Graesser et al., 2019] Graesser, L., Cho, K., and Kiela, D. (2019). Emergent linguistic phenomena in multi-agent communication games. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3698–3708. Association for Computational Linguistics.
- [Guan et al., 2020] Guan, Y., Ren, Y., Li, S. E., Sun, Q., Luo, L., and Li, K. (2020). Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization. *IEEE Transactions on Vehicular Technology*, 69(11):12597–12608.
- [Hirata et al., 2016] Hirata, Y., Inaba, M., Takahashi, K., Toriumi, F., Osawa, H., Katagami, D., and Shinoda, K. (2016). Werewolf game modeling using action probabilities based on play log analysis. In *International Conference on Computers and Games*, pages 103–114. Springer.
- [Hosu and Rebedea, 2016] Hosu, I. and Rebedea, T. (2016). Playing atari games with deep reinforcement learning and human checkpoint replay. *CoRR*, abs/1607.05077.
- [Jang et al., 2017] Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- [Kajiwara et al., 2014] Kajiwara, K., Toriumi, F., Ohashi, H., Osawa, H., Katagami, D., Inaba, M., Shinoda, K., Nishino, J., et al. (2014). Extraction of optimal strategies in human wolf using reinforcement learning. *Proceedings of the 76th National Convention*, 2014(1):597–598.
- [Katagami et al., 2014] Katagami, D., Takaku, S., Inaba, M., Osawa, H., Shinoda, K., Nishino, J., and Toriumi, F. (2014). Investigation of the effects of nonverbal information on werewolf. In *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 982–987. IEEE.
- [Kottur et al., 2017] Kottur, S., Moura, J. M. F., Lee, S., and Batra, D. (2017). Natural language does not emerge ‘naturally’ in multi-agent dialog. In Palmer, M., Hwa, R., and Riedel, S., editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2962–2967. Association for Computational Linguistics.
- [Lazaridou and Baroni, 2020] Lazaridou, A. and Baroni, M. (2020). Emergent multi-agent communication in the deep learning era. *CoRR*, abs/2006.02419.
- [Leibo et al., 2019] Leibo, J. Z., Hughes, E., Lanctot, M., and Graepel, T. (2019). Autocurricula and the emergence of innovation from social interaction: A manifesto for multi-agent intelligence research. *CoRR*, abs/1903.00742.
- [Li et al., 2020] Li, Y., Ponti, E. M., Vulic, I., and Korhonen, A. (2020). Emergent communication pretraining for few-shot machine translation. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 4716–4731. International Committee on Computational Linguistics.
- [Liang et al., 2020] Liang, P. P., Chen, J., Salakhutdinov, R., Morency, L., and Kottur, S. (2020). On emergent communication in competitive multi-agent teams. In Seghrouchni, A. E. F., Sukthankar, G., An, B., and Yorke-Smith, N., editors, *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS ’20, Auckland, New Zealand, May 9-13, 2020*, pages 735–743. International Foundation for Autonomous Agents and Multiagent Systems.
- [Lowe et al., 2019] Lowe, R., Foerster, J. N., Boureau, Y., Pineau, J., and Dauphin, Y. N. (2019). On the pitfalls of measuring emergent communication. In Elkind, E., Veloso, M., Agmon, N., and Taylor, M. E., editors, *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS ’19, Montreal, QC, Canada, May 13-17, 2019*, pages 693–701. International Foundation for Autonomous Agents and Multiagent Systems.
- [Lowe et al., 2017] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*.
- [Luo et al., 2018] Luo, J., Green, S., Feghali, P., Legrady, G., and Koc, Ç. K. (2018). Visual diagnostics for deep reinforcement learning policy development. *CoRR*, abs/1809.06781.
- [Maddison et al., 2017] Maddison, C. J., Mnih, A., and Teh, Y. W. (2017). The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.
- [Modi et al., 2005] Modi, P. J., Shen, W.-M., Tambe, M., and Yokoo, M. (2005). Adopt: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161(1-2):149–180.
- [Mordatch and Abbeel, 2018] Mordatch, I. and Abbeel, P. (2018). Emergence of grounded compositional language in multi-agent populations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- [Nakamura et al., 2016] Nakamura, N., Inaba, M., Taka-

- hashi, K., Toriumi, F., Osawa, H., Katagami, D., and Shinoda, K. (2016). Constructing a human-like agent for the werewolf game using a psychological model based multiple perspectives. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE.
- [O’Brien and Nicol, 1998] O’Brien, P. D. and Nicol, R. C. (1998). Fipa—towards a standard for software agents. *BT Technology Journal*, 16(3):51–59.
- [Schmidhuber, 2015] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.
- [Schulman et al., 2015] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897.
- [Schulman et al., 2017] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.
- [Shoham et al., 2003] Shoham, Y., Powers, R., and Grenager, T. (2003). Multi-agent reinforcement learning: a critical survey. *Web manuscript*, 2.
- [Silver et al., 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.
- [Silver et al., 2014] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms.
- [Sutton et al., 1999] Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12:1057–1063.
- [Tampuu et al., 2017] Tampuu, A., Matiisen, T., Kodelja, D., Kuzovkin, I., Korjus, K., Aru, J., Aru, J., and Vicente, R. (2017). Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395.
- [Tan, 1993] Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337.
- [Thrun and Liu, 2005] Thrun, S. and Liu, Y. (2005). Multi-robot slam with sparse extended information filters. In *Robotics Research. The Eleventh International Symposium*, pages 254–266. Springer.
- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.
- [Vinyals et al., 2019] Vinyals, O., Babuschkin, I., Czarnnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.
- [Vytelingum et al., 2010] Vytelingum, P., Voice, T. D., Ramchurn, S. D., Rogers, A., and Jennings, N. R. (2010). Agent-based micro-storage management for the smart grid.
- [Wagner et al., 2003] Wagner, K., Reggia, J. A., Uriagereka, J., and Wilkinson, G. S. (2003). Progress in the simulation of emergent communication and language. *Adaptive Behavior*, 11(1):37–69.
- [Wang and Kaneko, 2018] Wang, T. and Kaneko, T. (2018). Application of deep reinforcement learning in werewolf game agents. In *2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, pages 28–33.
- [Wei et al., 2019] Wei, H., Liu, X., Mashayekhy, L., and Decker, K. (2019). Mixed-autonomy traffic control with proximal policy optimization. In *2019 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE.
- [Wiseman and Lewis, 2019] Wiseman, S. and Lewis, K. (2019). What do players rely on in social deduction games? In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts*, pages 781–787.
- [Wu, 2001] Wu, D.-J. (2001). Software agents for knowledge management: coordination in multi-agent supply chains and auctions. *Expert Systems with Applications*, 20(1):51–64.
- [Zhou and Roumeliotis, 2006] Zhou, X. S. and Roumeliotis, S. I. (2006). Multi-robot slam with unknown initial correspondence: The robot rendezvous case. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 1785–1792. IEEE.

## Appendix

### A. Appendix

Following, some specific aspects for the *RLupus* environment are discussed. We decided to provide them in the Appendix to avoid interrupting the normal flow of the paper. Indeed what comes in the next paragraphs are implementation details which are not required to understand the contribution of this article.

#### A.1. Environment

##### A.1.1. Observation Space

The observation space characterizes what the agents perceive in the environment. Multiple elements make up this space in the environment; each of them is later packed into a common dictionary and fed to the players:

- *Phase*:  $p \in \{1, 2, 3, 4\}$ , to represent the four different phases of the game.
- *Day*:  $d \in \mathcal{I}^+$ , to represent the day. An upper bound for the possible maximum day is set to 10 which reduces the one-hot encoder representation length.
- *Status Map*:  $d \in \mathcal{B}$ , this array maps a boolean value to an agent index, effectively informing the players if agent  $i$  is alive = 1 or dead = 0.
- *Own id*:  $oi \in [0, N]$ , at the start of each game the agents' ids are shuffled. To keep an agent informed about its new position this integer is necessary.
- *Targets*:  $t \in [-1, N]$ , to group all the targets.
- *Signal*:  $s \in [-1, SR - 1]$ , to group all the signals when there are any.

##### A.1.2. Rewards

In the environment there are five conditions that determine if an agent is punished or rewarded:

- *Day*: At the end of each day every agent is penalized by a small factor ( $-1$ ). The purpose is to train the agents to quickly win the game, trying to develop new policies to win faster.
- *Death*: Each time a player dies, it takes a  $-5$  penalty. While dying can be a strategy in some matches, this behavior needs to be controlled to avoid a high number of suicides.
- *Target Accord*: To keep the voting system become another form of communication, the agents are penalized when they voted for someone that later on are not killed. Since execution depends on the coordination and communication of the required group of agents, this rewards is aimed to encourage a cooperative behavior
- *Win/Lost*: Until now the rewards were assigned per agent. At the end of a match each group, either werewolves or villagers, is rewarded or penalized by a factor  $\pm 25$ .

##### A.1.3. Trainable Policy

Since the neural network architecture is positioned in between the policy and the value function, the loss function must take into account the contribution given by the latter, thus an additional error term must be included:

$$L^{CLIP+VF} = E_t [L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta)]$$

Where  $L_t^{VF}(\theta) = (V_\theta(s_t) - V_t^{targ})^2$  is the squared-error loss between the value function  $V_\theta$  at state  $s_t$  and the target function,  $c_1$  is its coefficient. This addition is of fundamental importance to estimate the critic loss, that is the how well the model is able to predict the value of each state.



An additional term is introduced to regularize the total loss:

$$L^{CLIP+VF+S} = E_t [L_t^{CLIP+VS}(\theta) + c_2 S[\pi_\theta](s_t)]$$

The entropy coefficient is maximized when all the policies have equal probability to be chosen, that is when the agent is acting at random. Adding such value to the loss function incentivizes the training algorithm to minimize the entropy value, thus avoiding the premature convergence of one action probability dominating the policy and preventing exploration<sup>15</sup>. The term  $c_2$  scales the value of the entropy.

## A.2. Baselines

Understanding how AIs learn is a hard problem in the field of machine learning. In computer vision and specifically with convolutional neural networks (CNN), visualizing the activation layers together with the filters is a common practice to better understand the structure of the network itself. On the other hand, for RL environments, there has been some research in the area of visual diagnostic for RL systems using CNNs to elaborate visual stimulus [Luo et al., 2018]; for emergent language, the preferred way to quantitatively evaluate the collaboration between agents has been accomplished by measuring the performances of the agents themselves.

It is trivial to see that, whichever method is preferred, there must be some kind of evaluation method for the communication system. In the following, such evaluation is estimated firstly by drawing some baselines with complete random agents, and then by comparing the latter with the results yield by the training.

A game with nine players is relatively short. These agents determine a simple set of actions that can be easily mapped in a table; for this reason, the following section will focus only on nine players' settings (9P). Before showing any results the study of the complete behavior of a 9P setting is performed by inspecting its tree of possibilities.

An expanded tree, or tree of possibilities, shows every outcome of a particular process. At each possible intersection, a new branch is created associated with a probability  $p$ ; in the context of the werewolf game, each branch represents an outcome where, during the execution phase, the player has killed either a villager or a werewolf.

It is worth noticing that in a random 9P environment, the villagers are most likely to lose. As can be seen in the expanded tree for the game, 2, the villagers either execute a werewolf during the first day, with probability  $p = 3/8$ , or they lose. Moreover the only chance to win is achievable with probability:

$$p = \frac{3}{8} \cdot \frac{1}{3} \cdot \frac{1}{4} = 0.03125$$

Hence they have a probability of 3.12% to win with a complete random policy.

---

<sup>15</sup>For a complete understanding of the influence entropy has on policy optimizations, the reader is referred to [Ahmed et al., 2019].

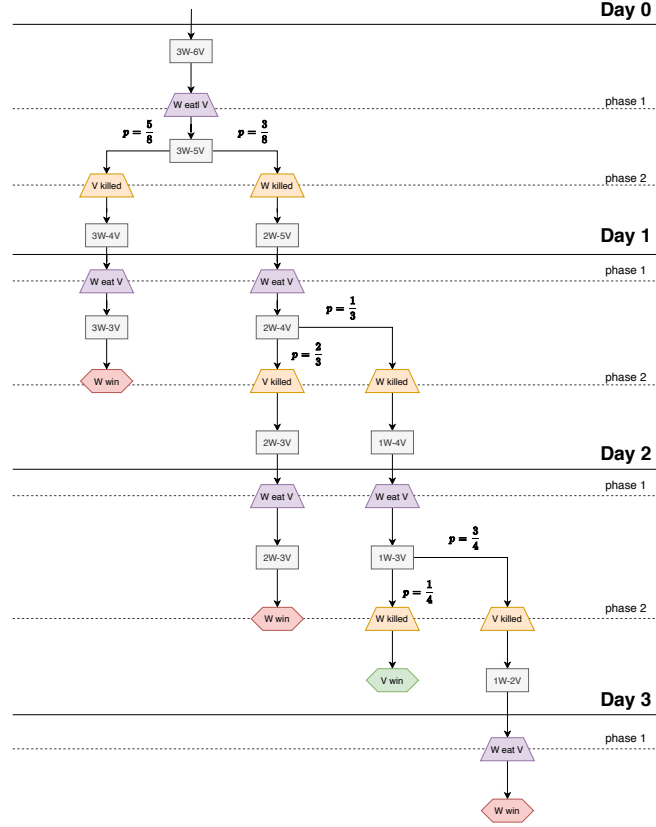


Fig. 2. Expanded tree for a 9 players game with 3 wolves and 6 villagers

### A.3. *AiWolf*

This section aims to provide some further details concerning the different implementation between the *RLupus* framework and the *AiWolf* one [Kajiwara et al., 2014].

#### A.3.1. Syntax protocol

On their [website](#) they define a specific protocol for the structure of sentences which is the combination of 6 words, i.e. unit of meaning:

1. *Subject*: an agent identifier which has a clear mapping with the *Own Id* parameter specified in the previous Section A.1.1
2. *Target*: an agent identifier, again the parallelism is the same with the *RLupus* setting.
3. *Role*: one of the 6 implemented roles. This kind of specification is not needed in our setting since only two roles are available.
4. *Species*: either human or werewolf.
5. *Verb*: a set of 15 valid verbs.
6. *Talk number*: a unique id for each sentence.

These atomic information units can define 13 different type of sentences which can be broadly divided into 5 categories:

1. Sentences expressing knowledge or intent.
2. Sentences about ongoing actions.
3. Sentences about past actions and their results.

4. Sentences that express dis/agreement.
5. Sentences referring to the flow of the conversation.

Finally, there are 8 types of operators which are used to frame sentences and express their relationship:

1. Request operators which are directed towards the acquisition of new knowledge.
2. Reasoning operators: e.g., because.
3. Time indication operators.
4. Logic operators such as: and, not, or, xor.

#### A.3.2. Comparison

An initial parallelism can be traced between the number of words and the signal parameters <sup>16</sup>. Still, the main difference is that *AiWolf* forces a prior meaning on the available words while *RLupus* leaves them free to be used by the agents. On one hand, defining a grammar and a syntax beforehand allows for a highly interpretable language, but on the other, this constrains the agents to adapt to a syntax that could not be optimal.

Moreover, since a sentence must be grammatically correct, some constraint enforcing this aspect must be applied to the agent through policy-shaping or on their output in a later phase. This further increases the complexity of the *AiWolf* environment and define a solid discrepancy with the freedom expressed in the *RLupus* one.

Finally, in the *RLupus* framework, no direct relationship between sentences can be express by the agents since no unique id is defined for a sentence in the observation space.

---

<sup>16</sup>SL, SR in Section A.1.1