Language-Goal Imagination to Foster Creative Exploration in Deep RL

Tristan Karch* [1] **Nicolas Lair*** [2 3] **Cédric Colas*** [1] **Jean-Michel Dussoux** [3] **Clément Moulin-Frier** [1] **Peter Ford Dominey** [2] **Pierre-Yves Oudeyer** [1]

Abstract

Developmental machine learning studies how artificial agents can model the way children learn open-ended repertoires of skills. Children are known to use language and its compositionality as a tool to imagine descriptions of outcomes they never experienced before and target them as goals during play. We introduce IMAGINE, an intrinsically motivated deep RL architecture that models this ability. Such imaginative agents, like children, benefit from the guidance of a social peer who provides language descriptions. To take advantage of goal imagination, agents must be able to leverage these descriptions to interpret their imagined goals. This generalization is made possible by modularity: a decomposition between learned goal-achievement reward function and policy relying on deep sets, gated attention and object-centered representations. We introduce the Playground environment and study how this form of goal imagination improves generalization and exploration over agents lacking this capacity.

1. Introduction

Building autonomous machines that can discover and learn a variety of skills is a long-standing goal in Artificial Intelligence. In this quest, we can draw inspiration from the way children learn and explore open-ended environments (Asada et al., 2009). In such exploration, *intrinsic motivations* are key to trigger spontaneous exploration, for the mere purpose of experiencing novelty, surprise or learning progress (Kaplan & Oudeyer, 2007; Kidd & Hayden, 2015).

Children also leverage the properties of language to assimilate years of experience embedded in their culture, in only a few years (Tomasello, 1999; Bruner, 1991). As they discover language, their goal-driven exploration changes. Piaget (1926) identified a form of egocentric speech where children narrate their ongoing activities. Later, Vygotsky (1978) showed that they can generate novel plans and goals by using the expressive generative properties of language.

This paper presents **I**ntrinsic **M**otivations **A**nd **G**oal **IN**vention for **E**xploration (IMAGINE): a learning architecture which leverages natural language (NL) interactions with a descriptive social partner (SP) to explore procedurally-generated scenes and interact with objects. IMAGINE discovers meaningful environment interactions through its own exploration (Figure 1a) and episode-level NL descriptions provided by SP (1b). These descriptions are turned into targetable goals by the agent (1c). It learns to represent them by jointly training a language encoder mapping NL to goal embeddings and a goal-achievement reward function (1d). The latter provides training signals for policy learning (ticks in Figure 1d-e). More importantly, IMAGINE can invent new goals by composing known ones, and trains on them autonomously via internal signals (1f).

Related work. The idea that language understanding is grounded in one's experience of the world and should not be secluded from the perceptual and motor systems has a long history in Cognitive Science (Glenberg & Kaschak, 2002; Zwaan & Madden, 2005). This vision was transposed to intelligent systems (Steels, 2006; McClelland et al., 2019), applied to human-machine interaction (Dominey, 2005; Madden et al., 2010) and recently to deep RL via frameworks such as *BabyAI* (Chevalier-Boisvert et al., 2019).

In their review of *RL algorithms informed by NL*, Luketina et al. (2019) distinguish between *language-conditional* problems where language is required to solve the task and *language-assisted* problems where language is a supplementary help. In the first category, most works propose instruction-following agents (R. K. Branavan et al., 2010; Chen & Mooney, 2011; Bahdanau et al., 2019; Co-Reyes et al., 2018; Jiang et al., 2019; Goyal et al., 2019; Cideron et al., 2019). Although our system is *language-conditioned*, it is not *language-instructed*: it is never given any instruction or reward but sets its own goals and learns its own internal reward function. Bahdanau et al. (2019) and Fu et al. (2019) also learn a reward function but require extensive expert knowledge (expert dataset and known environment

*Equal contribution [1]Flowers Team, Inria Bordeaux, FR. [2]INSERM U1093, Dijon, FR. [3]Cloud Temple, Paris, FR.. Correspondence to: Nicolas Lair <nicolas.lair@inserm.fr>.

Accepted to the 1st Workshop on Language in Reinforcement Learning, ICML 2020, Copyright 2020 by the author(s).

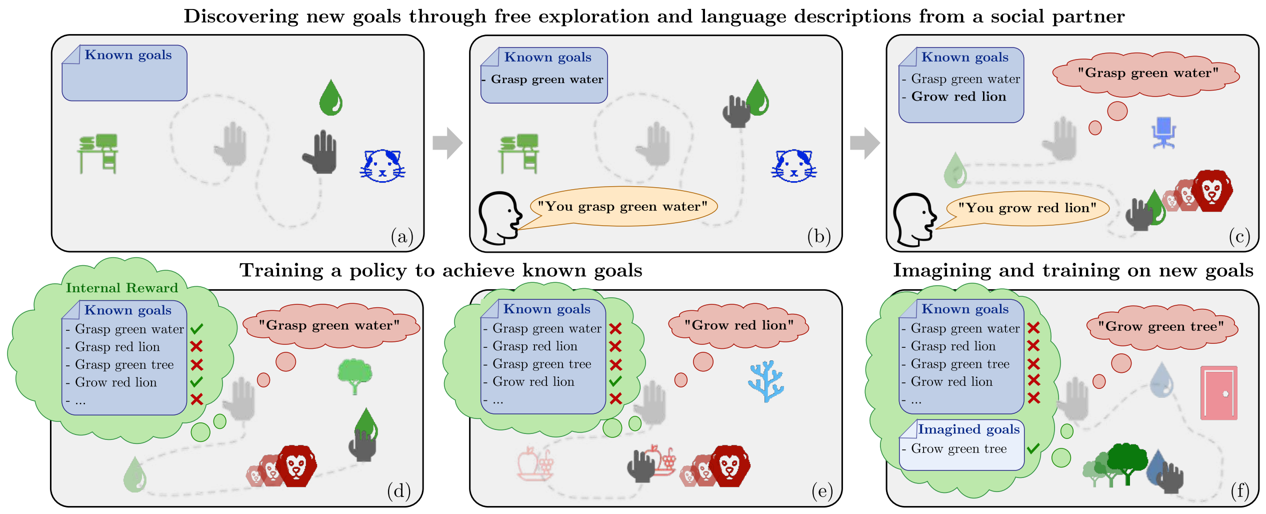


Figure 1: **IMAGINE overview**. In the *Playground* environment, the agent (hand) can move, grasp objects and grow some of them. Scenes are generated procedurally with objects of different types, colors and sizes. A social partner provides descriptive feedback (orange), that the agent converts into targetable goals (red bubbles).

dynamics respectively), whereas our agent uses experience generated by its own exploration.

Imagining goals by composing known ones only works in association with *systematic generalization* (Bahdanau et al., 2018; Hill et al., 2019): generalizations of the type *grow any animal* + *grasp any plant* → *grow any plant*. These were found to emerge in instruction-following agents, including generalizations to new combinations of motor predicates, object colors and shapes (Hermann et al., 2017; Hill et al., 2019; Bahdanau et al., 2019). Because systematic generalization can only occur when objects share common attributes (e.g. type, color), we represent objects as *single-slot object files* (Green & Quilty-Dunn, 2017): separate entities characterized by shared attributes. We introduce a new object-centered inductive bias: object-based modular architectures based on Deep Sets (Zaheer et al., 2017).

Contributions. We introduce goal imagination using language compositionality to drive exploration of intrisically motivated agents and a modular architecture to enable it.

2. Problem Definition

Open-ended learning environment. We consider a setup where agents evolve in an environment filled with objects and without prior on the set of possible interactions. An agent decides what to learn by setting its own goals, and has no access to external rewards. However, to allow the agent to learn relevant skills, a social partner (SP) watches the scene to guide the agent. Following a developmental approach (Asada et al., 2009), we propose a hard-coded surrogate SP modelling important aspects of human development:

- At the beginning of each episode, the agent chooses a goal by formulating a sentence. SP then provides agents with optimal learning opportunities by organizing the scene with: 1) the required objects to reach the goal (relevant scene) 2) procedurally-generated distracting objects (new scene and providing further discovery opportunities). This is close to a developmental scaffolding modelling the Zone of Proximal Development (ZPD) introduced by Vygotsky to describe infant-parent learning dynamics (Vygotsky, 1978).
- At the end of each episode, SP utters a set of sentences describing achieved and meaningful outcomes (except sentences from a test set). Linguistic guidance given through descriptions are a key component of how parents "teach" language to infants. It contrasts with instruction following (providing a linguistic command and then a reward), rarely seen in real parent-child interactions (Tomasello, 2009; Bornstein et al., 1992).

Learning objective. This paper investigates how goal imagination can lead agents to creatively explore their environment and discover interesting interactions with objects around. In this quest, SP guides agents towards a set of interesting outcomes by uttering NL descriptions. Through compositional recombinations of these sentences, goal imagination aims to drive creative exploration, to push agents to discover outcomes beyond the set of outcomes known by SP. We evaluate this desired behavior by two metrics: 1) the generalization of the policy to new language goals unknown to SP, (test set defined in Supp. Section 7) ; 2) goal-oriented exploration metrics. We measure generalization for each goal as the success rate SR over 30 episodes and report $\overline{\text{SR}}$ the average over goals. We evaluate exploration with the *interesting interaction count* (I2C). Given a set of interaction, I2C measures the penchant of agents to explore interactions of this set, (details in Supp. Section 8). We report means and *std* over 10 seeds and statistical significance using a two-tail Welch's t-test at level $\alpha = 0.05$ (Colas et al., 2019b).

3. Methods

3.1. The *Playground* environment

We introduce *Playground*, a simple environment designed to study the impact of goal imagination on exploration and generalization by disentangling it from the problems of perception and fully-blown NL understanding. *Playground* is a continuous 2D world, with procedurally-generated scenes containing $N = 3$ objects, from 32 different object types (*e.g. dog, cactus, sofa, water, etc.*), organized into 5 categories (*animals, furniture, plants, etc*), see Figure 1.

Agent perception and embodiment. Agents have access to state vectors describing the scene: the agent's body and the objects. Each object is represented by a set of features describing its type, position, color, size and whether it is grasped. The agent can perform bounded translations in the 2D plane, grasp and release objects with its gripper. It can make animals and plants grow by bringing them the right supply (food or water for animals, water for plants).

Goal and description A simple grammar generates the descriptions of the 256 achievable goals, such as *go bottom left, grasp red cat* or *grow green animal*. Supp. Section 7 completely details the environment and the grammar

3.2. The IMAGINE Architecture

IMAGINE agents build a repertoire of goals and train two internal models: 1) a goal-achievement reward function $\mathcal{R}$ to predict whether a given description matches a behavioral trajectory; 2) a policy Π to achieve behavioral trajectories matching descriptions. The architecture is presented in Figure 2 and follows this logic:

1. The *Goal Generator* samples a target goal g_{target} from known and imagined goals ($\mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{im}}$).
2. The agent (*RL Agent*) interacts with the environment using its policy Π conditioned on g_{target}.
3. State-action trajectories are stored in *mem*(Π).
4. SP's descriptions of the last state are considered as potential goals $\mathcal{G}_{\text{SP}}(\mathbf{s}_T) = \mathcal{D}_{\text{SP}}(\mathbf{s}_T)$.
5. *mem*($\mathcal{R}$) stores positive pairs ($\mathbf{s}_T$, $\mathcal{G}_{\text{SP}}(\mathbf{s}_T)$) and infers negative pairs ($\mathbf{s}_T$, $\mathcal{G}_{\text{known}} \setminus \mathcal{G}_{\text{SP}}(\mathbf{s}_T)$).
6. The agent then updates:
   - *Goal Gen.*: $\mathcal{G}_{\text{known}} \leftarrow \mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{SP}}(\mathbf{s}_T)$ and $\mathcal{G}_{im} \leftarrow \text{Imagination}(\mathcal{G}_{\text{known}})$.
   - *Language Encoder* (L_e) and *Reward Function* ($\mathcal{R}$) are updated using data from *mem*($\mathcal{R}$).
   - *RL agent*: We sample a batch of state-action transitions ($\mathbf{s}$, $\mathbf{a}$, $\mathbf{s}'$) from *mem*(Π). Then, we use *Hindsight Replay* and $\mathcal{R}$ to bias the selection of substitute goals to train on (g_{s}) and compute the associated rewards ($\mathbf{s}$, $\mathbf{a}$, $\mathbf{s}'$, g_{s}, r). Finally, the policy and critic are trained via RL.

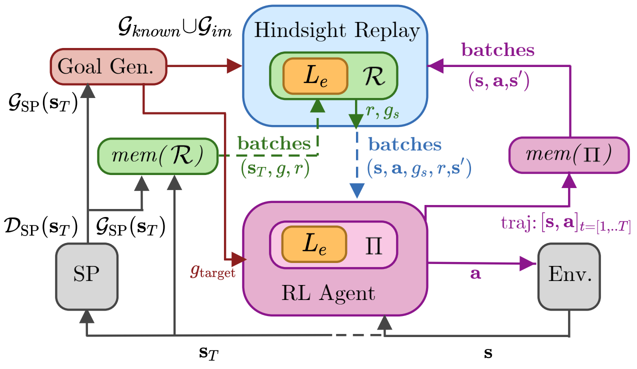
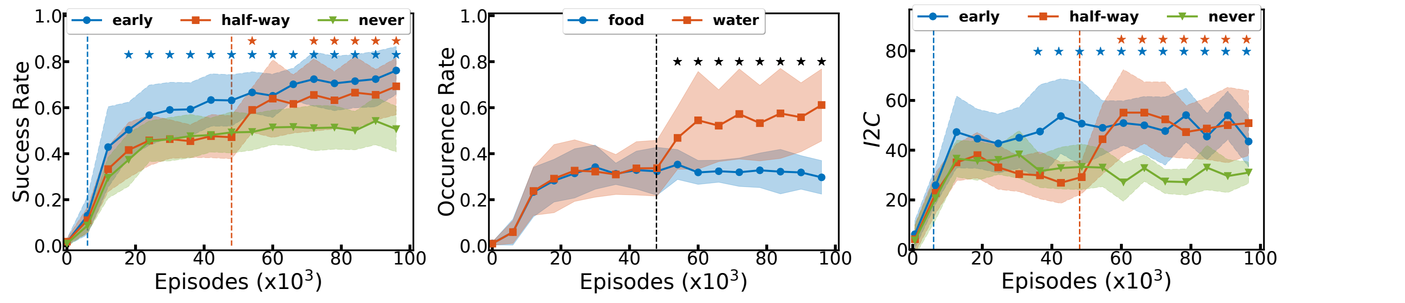

$\mathcal{G}_{known} \cup \mathcal{G}_{im}$
Hindsight Replay
L_e
$\mathcal{R}$
batches
$(\mathbf{s}, \mathbf{a}, \mathbf{s}')$
Goal Gen.
$\mathcal{G}_{\text{SP}}(\mathbf{s}_T)$
r, g_s
mem($\mathcal{R}$)
batches
$(\mathbf{s}_T, g, r)$
batches
$(\mathbf{s}, \mathbf{a}, g_s, r, \mathbf{s}')$
mem(Π)
$\mathcal{D}_{\text{SP}}(\mathbf{s}_T)$
$\mathcal{G}_{\text{SP}}(\mathbf{s}_T)$
traj: $[\mathbf{s}, \mathbf{a}]_{t=[1,..T]}$
L_e
Π
SP
g_{target}
RL Agent
$\mathbf{a}$
Env.
$\mathbf{s}_T$
$\mathbf{s}$

Figure 2: **IMAGINE architecture.** Colored boxes show the different modules of IMAGINE. Lines represent update signals (dashed) and function outputs (plain). The language encoder L_e is shared. See Supp. Section 11.1

Goal generator. It is a generative model of NL goals. It generates target goals g_{target} for data collection and substitute goals g_{s} for hindsight replay. When goal imagination is disabled, the goal generator samples uniformly from the set of known goals $\mathcal{G}_{\text{known}}$, sampling random vectors if empty. When enabled, it samples with equal probability from $\mathcal{G}_{\text{known}}$ and $\mathcal{G}_{\text{im}}$ (set of imagined goals). $\mathcal{G}_{\text{im}}$ is generated using a mechanism grounded in construction grammar that leverages the compositionality of language to imagine new goals from $\mathcal{G}_{\text{known}}$. The heuristic consists in computing sets of *equivalent words*: words that appear in two sentences that only differ by one word. For example, from *grasp red lion* and *grow red lion*, *grasp* and *grow* can be considered *equivalent* and from *grasp green tree* one can imagine a new goal *grow green tree* (see Figure 1f). Among them, some are meaningless, some are syntactically correct but infeasible (e.g. *grow red lamp*) and some belong to $\mathcal{G}^{\text{test}}$, or even to $\mathcal{G}^{\text{train}}$ before they are encountered by the agent and described by SP. The pseudo-code and all imaginable goals are provided in Supp. Section 9.

Object-centered modular architectures. The goal-achievement reward function, policy and critic leverage novel *modular-attention* (MA) architectures based on Deep Sets (Zaheer et al., 2017), gated attention mechanisms (Chaplot et al., 2017) and object-centered representations. The idea is to ensure efficient skill transfer between objects, no matter their position in the state vector. This is done through the combined use of a shared neural network that encodes object-specific features and a permutation-invariant function to aggregate the resulting latent encodings. The shared network independently encodes, for each object, an affordance between this object (object observations), the agent (body observations) and its current goal. The goal embedding, generated by the language encoder L_e, is first cast



(a) Testing set performance (b) Behavioral adaptation (c) Exploration metrics on testing set

Figure 3: **Goal imagination drives exploration and generalization.** Vertical dashed lines mark the onset of goal imagination. (a) $\overline{\text{SR}}$ on testing set. (b) Behavioral adaptation, empirical probabilities that the agent brings supplies to a plant when trying to grow it. (c) I2C computed on the testing set. Stars indicate significance (a and c are tested against *never*).

into an attention vector in $[0, 1]$, then fused with the concatenation of object and body features via an Hadamard product (gated-attention (Chaplot et al., 2017)). The resulting object-specific encodings are aggregated by a permutation-invariant function and mapped to the desired output via a final network (e.g. into actions or action-values). Supp. Section 11.1 provides visual representations and details about the reward function, policy and critic based on this architecture.

4. Experiments and Results

As IMAGINE achieve near perfect generalizations on training goals: $\overline{\text{SR}} = 0.95 \pm 0.05$, we focus on the testing set performance. Supp. Sections 8 to 10 provide more results.

Global generalization performance. Figure 3a shows $\overline{\text{SR}}$ on the set of testing goals, when the agent starts imagining new goals early (after $6 \cdot 10^3$ episodes), half-way (after $48 \cdot 10^3$ episodes) or when not allowed to do so. Imagining goals leads to significant improvements in generalization.

A particular generalization: growing plants. Agents learn to grow animals from SP's descriptions, but are never told they could grow plants. When evaluated offline on the *growing-plants* goals before goal imagination, agents' policies perform a sensible zero-shot generalization and bring them water or food with equal probability, as they would do for animals (Figure 3b, left). As they start imagining these goals, their behavior adapts (Figure 3b, right). A reward function with good zero-shot abilities only provides positive rewards when the agent brings water. The policy slowly adapts to this internal reward signal and pushes agents to bring more water. We name this: *behavioral adaptation*.

Exploration. Figure 3c presents the I2C metric computed on the set of interactions related to $\mathcal{G}^{\text{test}}$ and demonstrates the exploration boost triggered by goal imagination. Supp. Section 8 additional results on exploration.

Modularity is Essential to Goal Imagination. Flat architectures (FA) consider the whole scene at once. We compared policies based on MA and in Table 4. Both used MA reward function as the FA reward function showed poor performance on $\mathcal{G}^{\text{train}}$. MA shows stronger generalization and is the only architecture allowing an additional boost with goal imagination. Only MA policy can leverage the novel reward signals coming from imagined goals and show *behavioral adaptation*.

Figure 4: Policy architectures performance. $\overline{\text{SR}}_{\text{test}}$ at convergence.

	MA *	FA
Im.	0.76 ± 0.1	0.15 ± 0.05
No Im.	0.51 ± 0.1	0.17 ± 0.04
p-val	4.8e-5	0.66

5. Discussion and Conclusion

IMAGINE is an autonomous learning architecture that leverages NL interactions with a social partner. IMAGINE sets its own goals and builds behavioral repertoires without external rewards. As such, it is distinct from traditional instruction-following RL agents. This is done through the joint training of a language encoder for goal representation and a goal-achievement reward function to generate internal rewards. Our proposed modular architectures with gated-attention enable efficient out-of-distribution generalization of the reward function and policy. The ability to imagine new goals by composing known ones leads to further improvements over initial generalization abilities and fosters exploration beyond the set of interactions relevant to SP. Our agent even tries to grow pieces of furniture with supplies, a behavior that can echo the way a child may try to feed his doll.

IMAGINE does not need externally-provided rewards but learns which behaviors are *interesting* from language-based interactions with SP. In contrast with hand-crafted reward functions, NL descriptions provide an easy way to guide machines towards relevant interactions. In addition, Section 10 shows that agents can learn to achieve goals from a relatively small number of descriptions, paving the way towards human-provided descriptions.

6. Ackowledgement

Cédric Colas and Tristan Karch are partly funded by the French Ministère des Armées - Direction Générale de l'Armement. Nicolas Lair is supported by ANRT/CIFRE contract No. 151575A20 from Cloud Temple.

References

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, O. P., and Zaremba, W. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, pp. 5048–5058, 2017.

Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Ogino, M., and Yoshida, C. Cognitive developmental robotics: A survey. *IEEE transactions on autonomous mental development*, 1(1):12–34, 2009.

Bahdanau, D., Murty, S., Noukhovitch, M., Nguyen, T. H., de Vries, H., and Courville, A. Systematic generalization: What is required and can it be learned?, 2018.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Kohli, P., and Grefenstette, E. Learning to Understand Goal Specifications by Modelling Reward. In *International Conference on Learning Representations*, jun 2019.

Bornstein, M. H., Tamis-LeMonda, C. S., Tal, J., Ludemann, P., Toda, S., Rahn, C. W., Pêcheux, M.-G., Azuma, H., and Vardi, D. Maternal responsiveness to infants in three societies: The united states, france, and japan. *Child development*, 63(4):808–821, 1992.

Bruner, J. The Narrative Construction of Reality. *Critical Inquiry*, 18(1):1–21, oct 1991. ISSN 0093-1896. doi: 10.1086/448619.

Chan, H., Wu, Y., Kiros, J., Fidler, S., and Ba, J. Actrce: Augmenting experience via teacher's advice for multi-goal reinforcement learning, 2019.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R. Gated-attention architectures for task-oriented language grounding, 2017.

Chen, D. L. and Mooney, R. J. Learning to Interpret Natural Language Navigation Instructions from Observations. In *AAAI Conference on Artificial Intelligence (AAAI), 2011*, 2011.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. Baby{AI}: First Steps Towards Grounded Language Learning With a Human In the Loop. In *International Conference on Learning Representations*, 2019.

Cideron, G., Seurin, M., Strub, F., and Pietquin, O. Self-educated language agent with hindsight experience replay for instruction following. *arXiv preprint arXiv:1910.09451*, 2019.

Co-Reyes, J. D., Gupta, A., Sanjeev, S., Altieri, N., Andreas, J., DeNero, J., Abbeel, P., and Levine, S. Guiding policies with language via meta-learning, 2018.

Colas, C., Oudeyer, P., Sigaud, O., Fournier, P., and Chetouani, M. CURIOUS: intrinsically motivated modular multi-goal reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pp. 1331–1340, 2019a.

Colas, C., Sigaud, O., and Oudeyer, P.-Y. A hitchhiker's guide to statistical comparisons of reinforcement learning algorithms. *arXiv preprint arXiv:1904.06979*, 2019b.

Dominey, P. F. Emergence of grammatical constructions: evidence from simulation and grounded agent experiments. *Connection Science*, 17(3-4):289–306, sep 2005. ISSN 0954-0091. doi: 10.1080/09540090500270714.

Forestier, S. and Oudeyer, P.-Y. Modular active curiosity-driven discovery of tool use. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 3965–3972. IEEE, 2016.

Fu, J., Korattikara, A., Levine, S., and Guadarrama, S. From Language to Goals: Inverse Reinforcement Learning for Vision-Based Instruction Following. In *International Conference on Learning Representations*, 2019.

Glenberg, A. M. and Kaschak, M. P. Grounding language in action. *Psychonomic Bulletin & Review*, 9(3):558–565, sep 2002. ISSN 1069-9384. doi: 10.3758/BF03196313.

Goldberg, A. E. Constructions: A new theoretical approach to language. *Trends in cognitive sciences*, 7(5):219–224, 2003.

Goyal, P., Niekum, S., and Mooney, R. J. Using Natural Language for Reward Shaping in Reinforcement Learning. In *IJCAI 2019*, mar 2019. URL `http://arxiv.org/abs/1903.02020`.

Green, E. J. and Quilty-Dunn, J. What is an object file? *The British Journal for the Philosophy of Science*, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., Wainwright, M., Apps, C., Hassabis, D., and Blunsom, P. Grounded Language Learning in a Simulated 3D World. jun 2017. URL http://arxiv.org/abs/1706.06551.

Hill, F., Lampinen, A., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A. Emergent systematic generalization in a situated agent, 2019.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Jiang, Y., Gu, S., Murphy, K., and Finn, C. Language as an Abstraction for Hierarchical Deep Reinforcement Learning. In *Workshop on "Structure & Priors in Reinforcement Learning"at ICLR 2019*, jun 2019. URL http://arxiv.org/abs/1906.07343.

Kaplan, F. and Oudeyer, P.-Y. In search of the neural circuits of intrinsic motivation. *Frontiers in neuroscience*, 1:17, 2007.

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., and Bousquet, O. Measuring compositional generalization: A comprehensive method on realistic data, 2019.

Kidd, C. and Hayden, B. Y. The psychology and neuroscience of curiosity. *Neuron*, 88(3):449–460, 2015.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., Whiteson, S., and Rocktäschel, T. A Survey of Reinforcement Learning Informed by Natural Language. *IJCAI'19*, jun 2019. URL http://arxiv.org/abs/1906.03926.

Madden, C., Hoen, M., and Dominey, P. F. A cognitive neuroscience perspective on embodied language for human–robot cooperation. *Brain and Language*, 112(3):180–188, mar 2010. ISSN 0093-934X. doi: 10.1016/J.BANDL.2009.07.001.

Mankowitz, D. J., Zídek, A., Barreto, A., Horgan, D., Hessel, M., Quan, J., Oh, J., van Hasselt, H., Silver, D., and Schaul, T. Unicorn: Continual learning with a universal, off-policy agent. *CoRR*, abs/1802.08294, 2018. URL http://arxiv.org/abs/1802.08294.

McClelland, J. L., Hill, F., Rudolph, M., Baldridge, J., and Schütze, H. Extending machine language models toward human-level language understanding. *arXiv preprint arXiv:1912.05877*, 2019.

Piaget, J. *The language and thought of the child*. Routledge, 1926. ISBN 0415267501.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

R. K. Branavan, S., S. Zettlemoyer, L., and Barzilay, R. Reading Between the Lines: Learning to Map High-level Instructions to Commands. In *ACL 2010 - 48th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pp. 1268–1277, 2010.

Schaul, T., Horgan, D., Gregor, K., and Silver, D. Universal value function approximators. In *International Conference on Machine Learning*, pp. 1312–1320, 2015.

Steels, L. Semiotic dynamics for embodied agents. *IEEE Intelligent Systems*, 21(3):32–38, 2006.

Tomasello, M. *The cultural origins of human cognition*. Harvard University Press, 1999. ISBN 9780674005822.

Tomasello, M. *Constructing a language*. Harvard university press, 2009.

Vygotsky, L. S. Tool and Symbol in Child Development. In *Mind in Society*, chapter Tool and Symbol in Child Development, pp. 19–30. Harvard University Press, 1978. ISBN 0674576292. doi: 10.2307/j.ctvjf9vz4.6.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In *Advances in neural information processing systems*, pp. 3391–3401, 2017.

Zaremba, W., Sutskever, I., and Vinyals, O. Recurrent neural network regularization, 2014.

Zwaan, R. and Madden, C. Embodied sentence comprehension. *Grounding Cognition: The Role of Perception and Action in Memory, Language, and Thinking*, pp. 224–245, 2005. doi: 10.1017/CBO9780511499968.010.

Supplementary Material

This supplementary material provides additional methods, results and discussion, as well as implementation details.

- Section 7 gives a complete description of our setup and of the *Playground* environment.
- Section 8 presents a focus on exploration and how it is influenced by goal imagination.
- Section 9 presents a focus on the goal imagination mechanism we use for IMAGINE and identifies key properties of imagination.
- In Section 10, we investigate the case of more realistic feedback from SP.
- Section 11 gives all necessary implementation details.

7. Complete Description of the Playground Environment and Its Language

Environment description. The environment is a 2D square: $[-1.2, 1.2]^2$. The agent is a disc of diameter 0.05 with an initial position $(0, 0)$. Objects have sizes uniformly sampled from $[0.2, 0.3]$ and their initial positions are randomized so that they are not in contact with each other. The agent has an action space of size 3 bounded in $[-1, 1]$. The first two actions control the agent's continuous 2D translation (bounded to 0.15 in any direction). The agent can grasp objects by getting in contact with them and closing its gripper (positive third action), unless it already has an object in hand. Objects include 10 animals, 10 plants, 10 pieces of furniture and 2 supplies. Admissible categories are *animal, plant, furniture, supply* and *living_thing* (animal or plant), see Figure 5. Objects are assigned a color attribute (red, blue or green). Their precise color is a continuous RGB code uniformly sampled from RGB subspaces associated with their attribute color. Each scene contains 3 of these procedurally-generated objects (see paragraph about the Social Partner below).

Agent perception. At time step t, we can define an observation $\mathbf{o}_t$ as the concatenation of body observations (2D-position, gripper state) and objects' features. These two types of features form affordances between the agent and the objects around. These affordances are necessary to understand the meaning of object interactions like *grasp*. The state $\mathbf{s}_t$ used as input of the models is the concatenation of $\mathbf{o}_t$ and $\mathbf{\Delta o}_t = \mathbf{o}_t - \mathbf{o}_0$ to provide a sense of time. This is required to acquire the understanding and behavior related to the *grow* predicate, as the agent needs to observe and produce a change in the object's size.

Social Partner. SP has two roles:

- *Scene organization*: SP organize the scene according to the goal selected by the agent. When the agent selects a goal, it communicates it to SP. If the goal starts by the word *grow*, SP adds a procedurally-generated supply (water or food for animals, water for plants) of any size and color to the scene. If the goal contains an object (e.g. *red cat*), SP adds a corresponding object to the scene (with a procedurally generated size and RGB color). Remaining objects are generated procedurally. As a result, the objects required to fulfill a goal are always present and the scene contains between 1 (*grow* goals) and 3 (*go* goals) random objects. Note that all objects are procedurally generated (random initial position, RGB color and size).

- *Scene description*: SP provides NL descriptions of interesting outcomes experienced by the agent at the end of episodes. By default, SP respects the 3 following properties: *precision*: descriptions are accurate, *exhaustiveness*: it provides all valid descriptions for each episode and *full-presence*: it is always available. Section 10 investigates relaxations of the last two assumptions. It takes the final state of an episode ($\mathbf{s}_T$) as input and returns matching NL descriptions: $\mathcal{D}_{\text{SP}}(\mathbf{s}_T) \subset \mathcal{D}^{\text{SP}}$. When SP provides *descriptions*, the agent considers them as targetable *goals*. This mapping $\mathcal{D}^{\text{SP}} \to \mathcal{G}^{\text{train}}$ simply consists in removing the first *you* token (e.g. turning *you grasp red door* into the goal *grasp red door*). Given the set of previously discovered goals ($\mathcal{G}_{\text{known}}$) and new descriptions $\mathcal{D}_{\text{SP}}(\mathbf{s}_T)$, the agent infers the set of goals that were not achieved: $\mathcal{G}_{\text{na}}(\mathbf{s}_T) = \mathcal{G}_{\text{known}} \setminus \mathcal{D}_{\text{SP}}(\mathbf{s}_T)$, where $\setminus$ indicates the complement.

Grammar. We now present the grammar that generates descriptions for the set of goals achievable in the Playground environment ($\mathcal{G}^A$). **Bold** and { } refer to sets of words while *italics* refers to particular words:

1. Go: *(e.g. go bottom left)*
   - *go* + ***zone***
2. Grasp: *(e.g. grasp any animal)*
   - *grasp* + ***color*** ∪ {*any*} + ***object type*** ∪ ***object category***
   - *grasp* + *any* + ***color*** + *thing*
3. Grow: *(e.g. grow blue lion)*
   - *grow* + ***color*** ∪ {*any*} + ***living thing*** ∪ {*living_thing, animal, plant*}
   - *grow* + *any* + ***color*** + *thing*

Word sets are defined by:

- ***zone*** = {*center, top, bottom, right, left, top left, top right, bottom left, bottom right*}
- ***color*** = {*red, blue, green*}
- ***object type*** = ***living thing*** ∪ ***furniture*** ∪ ***supply***
- ***object category*** = {*living_thing*, *animal*, *plant*, *furniture*, *supply*}
- ***living thing*** = ***animal*** ∪ ***plant***

- ***animal*** = {*dog, cat, chameleon, human, fly, parrot, mouse, lion, pig, cow*}
- ***plant*** = {*cactus, carnivorous, flower, tree, bush, grass, algae, tea, rose, bonsai*}
- ***furniture*** = {*door, chair, desk, lamp, table, cupboard, sink, window, sofa, carpet*}
- ***supply*** = {*water, food*}
- ***predicate*** = {*go, grasp, grow*}

Training and testing goals sets The set of achievable goals is partitioned into *training* ($\mathcal{G}^{\text{train}}$) and *testing* ($\mathcal{G}^{\text{test}}$). Goals from $\mathcal{G}^{\text{test}}$ are intended to evaluate the ability of our agent to explore the set of achievable outcomes beyond the set of outcomes described by SP. $\mathcal{G}^{\text{test}}$ maximizes the compound divergence with a null atom divergence with respect to $\mathcal{G}^{\text{train}}$: testing sentences (compounds) are out of the distribution of $\mathcal{G}^{\text{train}}$ sentences, but their words (atoms) belong to the distribution of words in $\mathcal{G}^{\text{train}}$ (Keysers et al., 2019). SP only provides descriptions from $\mathcal{G}^{\text{train}}$. Note that some goals might be syntactically valid but not achievable. This includes all goals of the form *grow* + ***color*** $\cup$ {*any*} + ***furniture*** $\cup$ {*furniture*} (e.g. *grow red lamp*).

Besides all goals related to the object *flower* are unknown to the SP and cannot be uttered. The agent can encounter *flower* but is never able to identify the object by its name. This will prevent the agent to imagine new goals about *flower* object.

IMAGINE Pseudo-Code. Algorithm 1 outlines the pseudo-code of our learning architecture. See Main Section 3.2 for high-level descriptions of each module and function.

Algorithm 1 IMAGINE

```
1: Input: env, SP
2: Initialize: $L_e$, $\mathcal{R}$, $\Pi$, $mem(\mathcal{R})$, $mem(\Pi)$, $\mathcal{G}_{\text{known}}$, $\mathcal{G}_{\text{im}}$
           # Random initializations for networks
           # empty sets for memories and goal sets
3: for $e = 1 : N_{episodes}$ do
4:   if $\mathcal{G}_{\text{known}} \neq \emptyset$ then
5:     sample $g_{\text{NL}}$ from $\mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{im}}$
6:     $g \leftarrow L_e(g_{\text{NL}})$
7:   else
8:     sample $g$ from $\mathcal{N}(0, \mathbf{I})$
9:   $s_0 \leftarrow$ env.reset()
10:  for $t = 1 : T$ do
11:    $a_t \leftarrow \pi(s_{t-1}, g)$
12:    $s_t \leftarrow$ env.step($a_t$)
13:    $mem_\Pi$.add($s_{t-1}, a_t, s_t$)
14:  $\mathcal{G}_{\text{SP}} \leftarrow$ SP.get_descriptions($s_T$)
15:  $\mathcal{G}_{\text{known}} \leftarrow \mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{SP}}$
16:  $mem(\mathcal{R})$.add($s_T$, $g_{\text{NL}}$) for $g_{\text{NL}}$ in $\mathcal{G}_{\text{SP}}$
17:  if goal imagination allowed then
18:    $\mathcal{G}_{\text{im}} \leftarrow$ Imagination($\mathcal{G}_{\text{known}}$) # see Algorithm 2
19:  Batch$_\Pi \leftarrow$ ModularBatchGenerator($mem(\Pi)$)
     # Batch$_\Pi$={$(s, a, s')$}
20:  Batch$_\Pi \leftarrow$ Hindsight(Batch$_\Pi$, $\mathcal{R}$, $\mathcal{G}_{\text{known}}$, $\mathcal{G}_{\text{im}}$) #
     Batch$_\Pi$={$(s, a, r, g, s')$} where $r = \mathcal{R}(s, g)$
21:  $\Pi \leftarrow$ RL_Update(Batch$_\Pi$)
22:  if $e$ % reward_update_freq == 0 then
23:    Batch$_\mathcal{R} \leftarrow$ ModularBatchGenerator($mem(\mathcal{R})$)
24:    $L_e$, $\mathcal{R} \leftarrow$ LE&RewardFunctionUpdate(Batch$_\mathcal{R}$)
```

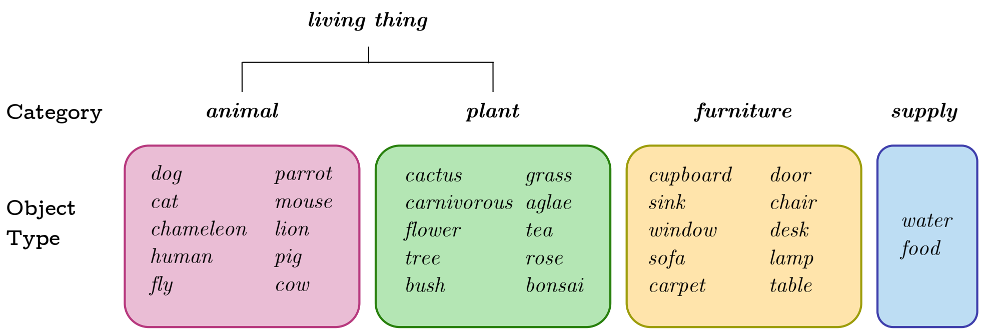


Figure 5: Representation of possible objects types and categories.

Table 1: Testing goals in $\mathcal{G}^{\text{test}}$.

Non-imaginable goals	*Grasp any flower*, *Grasp blue flower*, *Grasp green flower*, *Grasp red flower*, *Grow any flower*, *Grow blue flower*, *Grow green flower*, *Grow red flower*
Imaginable goals	*Grasp blue door*, *Grasp green dog*,*Grasp red tree*, *Grow green dog* *Grasp any animal*, *Grasp blue animal*, *Grasp green animal*, *Grasp red animal* *Grasp any fly*, *Grasp blue fly*, *Grasp green fly*, *Grasp red fly* *Grow any algae*, *Grow any bonsai*, *Grow any bush*, *Grow any cactus* *Grow any carnivorous*, *Grow any grass*, *Grow any living_thing*, *Grow any plant* *Grow any rose*, *Grow any tea*, *Grow any tree*, *Grow blue algae* *Grow blue bonsai*, *Grow blue bush*,*Grow blue cactus*, *Grow blue carnivorous* *Grow blue grass*, *Grow blue living_thing*, *Grow blue plant*, *Grow blue rose* *Grow blue tea*, *Grow blue tree*,*Grow green algae*, *Grow green bonsai* *Grow green bush*, *Grow green cactus*, *Grow green carnivorous*, *Grow green grass* *Grow green living_thing*, *Grow green plant*, *Grow green rose*, *Grow green tea* *Grow green tree*, *Grow red algae*, *Grow red bonsai*, *Grow red bush* *Grow red cactus*, *Grow red carnivorous*, *Grow red grass*, *Grow red living_thing* *Grow red plant*, *Grow red rose*, *Grow red tea*, *Grow red tree*

8. Focus on exploration

Interesting Interactions. *Interesting interactions* are trajectories of the agent that humans could infer as goal-directed. If an agent brings water to a plant and grows it, it makes sense for a human. If it then tries to do this for a lamp, it also feels goal-directed, even though it does not work. This type of behavior characterizes the penchant of agents to interact with objects around them, to try new things and, as a result, is a good measure of exploration.

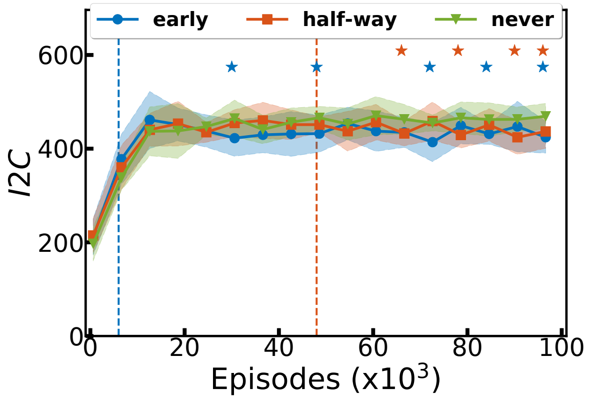
Sets of interesting interactions. We consider three sets of interactions: 1) interactions related to training goals; 2) to testing goals; 3) the extra set. This *extra set* contains interactions where the agent brings water or food to a piece of furniture or to another supply. Although such behaviors do not achieve any of the goals, we consider them as interesting exploratory behaviors. Indeed, they testify that agents try to achieve imagined goals that are meaningful from the point of view of an agent that does not already know that doors cannot be grown, i.e. corresponding to a meaningful form of generalization after discovering that animals or plants can be grown (e.g. *grow any door*).

The Interesting Interaction Count metric. We count the number of interesting interactions computed over all final transitions from the last 600 episodes (1 epoch). Agents do not need to target these interactions, we just report the number of times they are experienced. Indeed, the agent does not have to target a particular interaction for the trajectory to be interesting from an exploratory point of view. The HER mechanism ensures that these trajectories can be replayed to learn about any goal, imagined or not. Computed on the extra set, the *Interesting Interaction Count* (I2C) is the number of times the agent was found to bring supplies to a furniture or to other supplies over the last epoch:

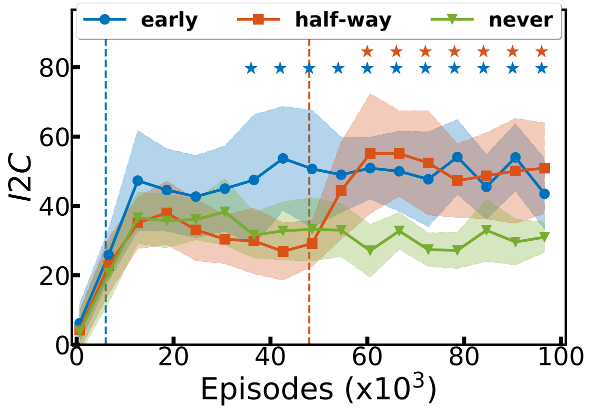
$$\text{I2C}_{\text{train}} = \sum_{i \in \mathcal{I} = \mathcal{G}_{\text{extra}}} \sum_{t=1}^{600} \delta_{i,t},$$

where $\delta_{i,t} = 1$ if interaction i was achieved in episode t, 0 otherwise and $\mathcal{I}$ is the set of interesting interactions (here from the extra set) performed during an epoch.

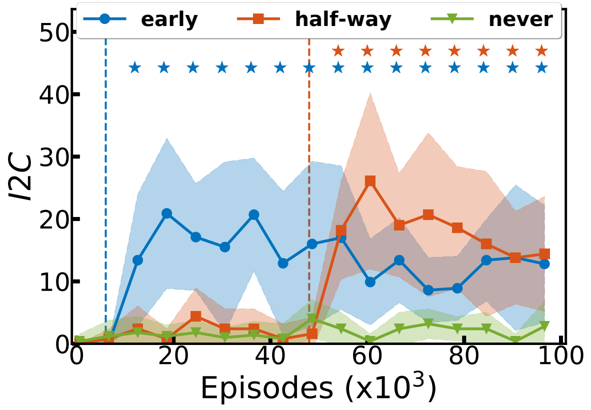
Agents that are allowed to imagine goals achieve higher scores in the testing and extra sets of interactions, while maintaining similar exploration scores on the training set, see Figures 6a to 6c.



(a)



(b)



(c)

Figure 6: **Exploration metrics** (a) Interesting interaction count (I2C) on training set, (b) I2C on testing set, (c) I2C on extra set. Goal imagination starts early (vertical blue line), half-way (vertical orange line) or does not start (*no imagination* baseline in green).

9. Focus on Goal Imagination

9.1. Goal Imagination Algorithm

Algorithm 2 presents the algorithm underlying our goal imagination mechanism. This mechanism is inspired from the *Construction Grammar* (CG) literature and generates new sentences by composing known ones (Goldberg, 2003). It computes sets of equivalent words by searching for sentences with an edit distance of 1: sentences where only one word differs. These words are then labelled equivalent, and can be substituted in known sentences. Note that the goal imagination process filters goals that are already known. Although all sentences from $\mathcal{G}^{\text{train}}$ can be imagined, there are filtered out of the imagined goals as they are discovered. Imagining goals from $\mathcal{G}^{\text{train}}$ before they are discovered drives the exploration of IMAGINE agents. In our setup, however, this effect remains marginal as all the goals from $\mathcal{G}^{\text{train}}$ are discovered in the first epochs.

Imagined goals. We run our goal imagination mechanism based on the Construction Grammar Heuristic (CGH) from $\mathcal{G}^{\text{train}}$. After filtering goals from $\mathcal{G}^{\text{train}}$, this produces 136 new imagined sentences. Table 2 presents the list of these goals while Figure 7 presents a Venn diagram of the various goal sets. Among these 136 goals, 56 belong to the testing set $\mathcal{G}^{\text{test}}$. This results in a coverage of 87.5% of $\mathcal{G}^{\text{test}}$, and a precision of 45%. In goals that do not belong to $\mathcal{G}^{\text{test}}$, goals of the form *Grow* + $\{$*any*$\}$ $\cup$ **color** + **furniture** $\cup$ **supplies** (e.g. *Grow any lamp*) are *meaningful* to humans, but are not achievable in the environment (*impossible*).

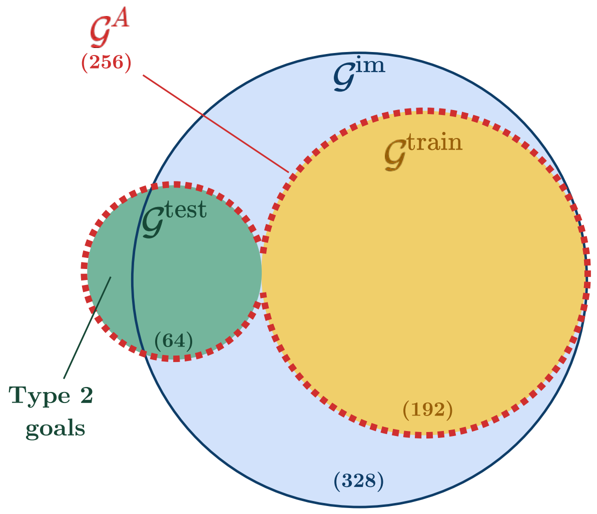

$\mathcal{G}^{A}$
(256)
$\mathcal{G}^{\text{im}}$
$\mathcal{G}^{\text{train}}$
$\mathcal{G}^{\text{test}}$
(64)
(192)
Type 2 goals
(328)

Figure 7: Venn diagram of goal spaces.

Goal Imagination Pseudo-Code. The pseudo-code of the goal imagination mechanism is presented in Algorithm 2. The edit distance between two sentences refers to the number of words to modify to transform one sentence into the other.

Algorithm 2 Goal Imagination.

```
1:  Input: $\mathcal{G}_{\text{known}}$ (discovered goals)
2:  Initialize: word_eq (list of sets of equivalent words, empty)
3:  Initialize: goal_template (list of template sentences used for imagining goals, empty)
4:  Initialize: $\mathcal{G}_{\text{im}}$ (empty)
5:  for $g_{\text{NL}}$ in $\mathcal{G}_{\text{known}}$ do {Computing word equivalences}
6:      new_goal_template = True
7:      for $g_m$ in goal_template do
8:          if edit_distance$(g_{\text{NL}}, g_m) < 2$ then
9:              new_goal_template = False
10:             if edit_distance$(g_{\text{NL}}, g_m) == 1$ then
11:                 $w_1, w_2 \leftarrow$ get_non_matching_words$(g_{NL}, g_m)$
12:                 if $w_1$ and $w_2$ not in any of word_eq sets then
13:                     word_eq.add$(\{w_1, w_2\})$
14:                 else
15:                     for eq_set in word_eq do
16:                         if $w_1 \in$ eq_set or $w_2 \in$ eq_set then
17:                             eq_set = eq_set $\cup \{w_1, w_2\}$
18:     if new_goal_template then
19:         goal_template.add$(g_{\text{NL}})$
20: for $g$ in goal_template do {Generating new sentences}
21:     for $w$ in $g$ do
22:         for eq_set in word_eq do
23:             if $w \in$ eq_set then
24:                 for $w'$ in eq_set do
25:                     $g_{im} \leftarrow$ replace$(g, w, w')$
26:                     if $g_{im} \notin \mathcal{G}_{\text{known}}$ then
27:                         $\mathcal{G}_{\text{im}} = \mathcal{G}_{\text{im}} \cup \{g_{im}\}$
28: $\mathcal{G}_{\text{im}} = \mathcal{G}_{\text{im}} \setminus \mathcal{G}_{\text{known}}$          {filtering known goals.}
```

Table 2: All imaginable goals $\mathcal{G}^{\text{im}}$ generated by the Construction Grammar Heuristic.

Goals from $\mathcal{G}^{\text{train}}$	$\mathcal{G}^{\text{train}}$. (Note that known goals are filtered from the set of imagined goals. However, any goal from $\mathcal{G}^{\text{train}}$ can be imagined before it is encountered in the interaction with SP.)
Goals from $\mathcal{G}^{\text{test}}$	All goals from $\mathcal{G}^{\text{test}}$ that are imaginable, see Table 1
Syntactically incorrect goals	*Go bottom top*, *Go left right*, *Grasp red blue thing*, *Grow blue red thing*, *Go right left*, *Go top bottom*, *Grasp green blue thing*, *Grow green red thing*, *Grasp green red thing* *Grasp blue green thing*, *Grasp blue red thing*, *Grasp red green thing*.
Syntactically correct but unachievable goals	*Go center bottom*, *Go center top*, *Go right center*, *Go right bottom*, *Go right top*, *Go left center*, *Go left bottom*, *Go left top*, *Grow green cupboard*, *Grow green sink*, *Grow blue lamp*, *Go center right*, *Grow green window*, *Grow blue carpet*, *Grow red supply*, *Grow any sofa*, *Grow red sink*, *Grow any chair*, *Go top center*, *Grow blue table*, *Grow any door*, *Grow any lamp*, *Grow blue sink*, *Go bottom center*, *Grow blue door*, *Grow blue supply*, *Grow green carpet*, *Grow blue furniture*, *Grow green supply*, *Grow any window*, *Grow any carpet*, *Grow green furniture*, *Grow green chair*, *Grow green food*, *Grow any cupboard*, *Grow red food*, *Grow any table*, *Grow red lamp* , *Grow red door*, *Grow any food*, *Grow blue window*, *Grow green sofa*, *Grow blue sofa*, *Grow blue desk*, *Grow any sink*, *Grow red cupboard*, *Grow green door*, *Grow red furniture*, *Grow blue food*, *Grow red desk* , *Grow red table*, *Grow blue chair*, *Grow red sofa*, *Grow any furniture*, *Grow red window*, *Grow any desk*, *Grow blue cupboard*, *Grow red chair*, *Grow green desk*, *Grow green table*, *Grow red carpet*, *Go center left*, *Grow any supply*, *Grow green lamp*, *Grow blue water*, *Grow red water*, *Grow any water*, Grow green water, *Grow any water*, Grow green water.

9.2. How Does Goal Imagination Properties Impact Performance?

Properties of imagined goals. We propose to characterize goal imagination mechanisms by two properties:

1. *Coverage*: the fraction of $\mathcal{G}^{\text{test}}$ found in $\mathcal{G}_{\text{im}}$
2. *Precision*: the fraction of the imagined goals that are achievable.

We compare our goal imagination mechanism based on the construction grammar heuristic (CGH) to variants characterized by:

1. *Lower coverage:* To reduce the coverage of CGH while maintaining the same precision, we simply filter half of the goals that would have been imagined by CGH. This filtering is probabilistic, resulting in different imagined sets for different runs. It happens online, meaning that the coverage is always half of the coverage that CGH would have had at the same time of training.
2. *Lower precision:* To reduce precision while maintaining the same coverage, we sample a random sentence (random words from the words of $\mathcal{G}^{\text{train}}$) for each goal imagined by CGH that does not belong to $\mathcal{G}^{\text{test}}$. Goals from $\mathcal{G}^{\text{test}}$ are still imagined via the CGH mechanism. This variants only doubles the imagination of sentences that do not belong to $\mathcal{G}^{\text{test}}$.
3. *Oracle:* Perfect precision and coverage is achieved by filtering the output of CGH, keeping only goals from $\mathcal{G}^{\text{test}}$. Once the 56 goals that CGH can imagine are imagined, the oracle variants adds the 8 remaining goals: those including the word *flower* that the agent cannot imagine.
4. *Random goals:* Each time CGH would have imagined a new goal, it is replaced by a randomly generated sentence, using words from the words of $\mathcal{G}^{\text{train}}$. It leads to near null coverage and precision and works as a baseline.

Note that all variants imagine goals at the same speed as the CGH algorithm. They simply filter or add noise to its output. For each variant, we measure the precision and coverage at the end of experiments, when all goals from $\mathcal{G}^{\text{train}}$ have been discovered and report them in Figure 3.

CGH vs Oracle Figure 9a shows that CGH achieves a generalization performance on par with the oracle. Reducing the coverage of the goal imagination mechanism still brings significant improvements in generalization.

Effect of lower coverage on generalization In Figure 9a, the Low Coverage condition seems to achieve good results, close to CGH. The *Low Coverage* variant of our goal imagination mechanism only covers 43.7% the test set with a 45% precision. To further study the impact of a lower coverage, we report in Figure 9b, success rates on testing goals of type *grow + plant* and compare with the *no imagination* baseline (green). We split in two: goals that were imagined (blue), and goals that were not (orange). The generalization performance on goals from $\mathcal{G}^{\text{test}}$ that the agent imagined (blue) are not significantly higher than the generalization performance on goals from $\mathcal{G}^{\text{test}}$ that were not imagined. As they are both significantly higher than the *no imagination* baseline, this implies that training on imagined goals boosts zero-shot generalization on similar goals that were not imagined.

Table 3: Coverage and Precision for different imagination mechanisms

	Coverage	**Precision**
CGH	0.87	0.45
Oracle	1	1
Low Coverage	0.44	0.45
Low Precision	0.87	0.30
Random Goal Imagination	≈ 0	≈ 0

Effect of lower precision on generalization Reducing the precision of imagined goals (gray curve) seems to impede generalization (no significant difference with the *no imagination* baseline), as shown in Figure 9a.

9.3. How Does Goal Imagination Properties Impact Exploration?

Figure 9 presents the I2C exploration scores on the training, testing and extra sets for the different goal imagination mechanisms above. Let us discuss each of these scores:

1. *Training interactions.* In Figure 9c, we see that decreasing the precision (Low Precision and Random Goal conditions) affects exploration on interactions from the training set, where it falls below the exploration of the *no imagination* baseline. This is due to the addition of meaningless goals forcing agent to allow less time to meaningful interactions relatively.
2. *Testing interactions.* Figure 9d shows that all goal imagination heuristics enable a significant exploration boost. The random goal baseline acts as a control condition. It demonstrates that the generalization boost is not due to a mere effect of network regularization introduced by adding random goals (no significant effect w.r.t. the *no imagination* baseline). In the same spirit, we also ran a control using random goal embedding, which did not produce any significant effects. We also see that the highest exploration scores on interactions from the test set comes from the oracle. Because it

shows high coverage and precision, its spends more time on the diversity of interactions from the testing set. What is more surprising is the exploration score of the low coverage condition, higher than the exploration score of CGH. With an equal precision, CGH should show better exploration, as it covers more test goals. However, the *Low Coverage* condition, by spending more time exploring each of its imagined goals (it imagined fewer), probably learned to master them better, increasing the robustness of its behavior towards those. This insight advocates for the use of goal selection methods based on learning progress (Forestier & Oudeyer, 2016; Colas et al., 2019a). Agents could estimate their learning progress on imagined goals using their internal reward function and its zero-shot generalization. Focusing on goals associated to high learning progress might help agents filter goals they can learn about from others.

3. *Extra interactions.* Figure 9e shows that only the goal imagination mechanisms that invent goals not covered by the testing set manage to boost exploration in this extra set. The oracle perfectly covers the testing set, but does not generate goals related to other objects (e.g. *grow any lamp*).

10. Focus on SP feedback

In this section, we study the relaxation of 2 of the 3 properties imposed on SP and wonder if we could Use More Realistic Feedbacks. We study the relaxation of the *full-presence* and *exhaustiveness* assumptions of SP. We first relax *full-presence* while keeping *exhaustiveness* (blue, yellow and purple curves). When SP has a 10% chance of being present (yellow), imaginative agents show generalization performance on par with the unimaginative agents trained in a full-presence setting (green), see Figure 8). However, when the same amount of feedback is concentrated in the first 10% episodes (purple), goal imagination enables significant improvements in generalization (w.r.t. green). This is reminiscent of children who require less and less attention as they grow into adulthood and is consistent with Chan et al. (2019). Relaxing *exhaustiveness*, SP only provides one positive and one negative description every episode (red) or in 50% of the episodes (gray). Then, generalization performance matches the one of unimaginative agents in the exhaustive setting (green).

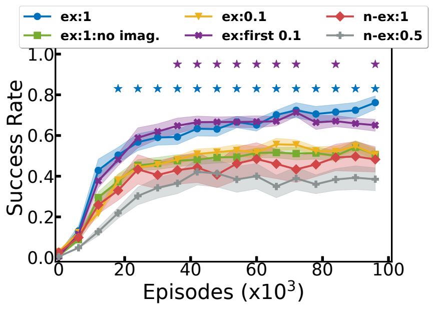
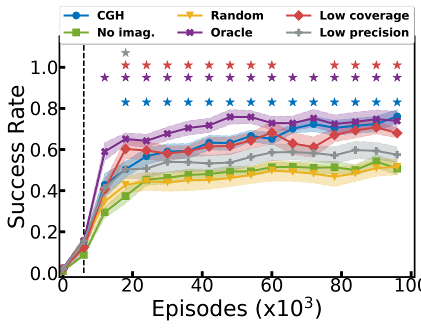
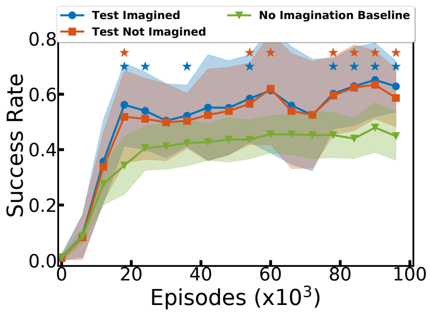


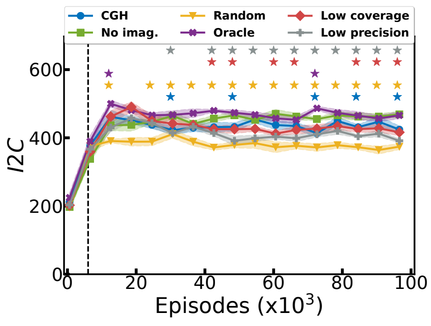
Figure 8: **Influence of social feedbacks.** $\overline{\text{SR}}$ on $\mathcal{G}^{\text{test}}$ for different social strategies. Stars indicate significant differences w.r.t. *ex:1 no imag.*. sem plotted, 5 seeds.



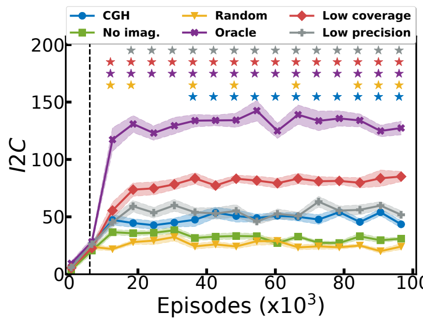
(a) $\overline{\textbf{SR}}$ **on** $\mathcal{G}^{\text{test}}$



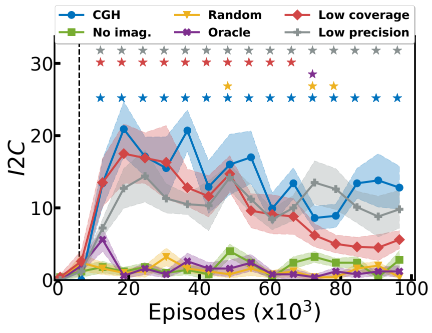
(b) $\overline{\textbf{SR}}$ **for Low Cov.**



(c) **I2C on** $\mathcal{G}^{\text{train}}$



(d) **I2C on** $\mathcal{G}^{\text{test}}$



(e) **I2C on** $\mathcal{G}^{\text{extra}}$

Figure 9: **Exploration metrics** For different goal imagination mechanisms: (a) Interesting interaction count (I2C) on training set, I2C on testing set, (c) I2C on extra set. Goal imagination starts early (vertical line), except for the *no imagination* baseline (green). We report *sem* (standard error of the mean) instead of *std* to improve readability. Stars indicate significant differences w.r.t the *no imagination* condition.(as usual, 10 seeds).

11. Architecture and implementation details

11.1. Architecture

Figure 2 shows the architecture of IMAGINE and its different modules. In this section, we give more details on each modules :

Language encoder. The language encoder (L_e) embeds NL goals ($L_e : \mathcal{G}^{\text{NL}} \rightarrow \mathbb{R}^{100}$) using an LSTM (Hochreiter & Schmidhuber, 1997). It is trained jointly with the reward function. L_e acts as a goal translator, turning the goal-achievement reward function, policy and critic into language-conditioned functions.

Reward function. Learning a goal-achievement reward function ($\mathcal{R}$) is framed as binary classification: $\mathcal{R}(\mathbf{s}, \mathbf{g}) : \mathcal{S} \times \mathbb{R}^{100} \rightarrow \{0, 1\}$. We use the MA architecture described in Section 3.2 with attention vectors $\boldsymbol{\alpha}^g$, a shared network $\text{NN}^{\mathcal{R}}$ with output size 1 and a logical OR aggregation. $\text{NN}^{\mathcal{R}}$ computes object-dependent rewards r_i in $[0, 1]$ from the object-specific inputs and the goal embedding. The final binary reward is computed by NN^{OR} which outputs 1 whenever $\exists j : r_j > 0.5$. We pre-trained a neural-network-based OR function to enable end-to-end training. Figure 10a shows the detailed architecture of the reward function. The overall reward function can be written as:

$$\mathcal{R}(\mathbf{s}, g) = \text{NN}^{\text{OR}}([\text{NN}^{\mathcal{R}}(\mathbf{s}_{obj(i)} \odot \boldsymbol{\alpha}^g)]_{i \in [1..N]})$$

Data. Interacting with the environment and SP, the agent builds a set of entries $[\mathbf{s}_T,\ g,\ r]$ with $g \in \mathcal{G}_{\text{known}}$ where $r \in \{0,\ 1\}$ rewards the achievement of g in state $\mathbf{s}_T$: $r = 1$ if $g \in \mathcal{G}_{\text{SP}}(\mathbf{s}_T)$ and 0 otherwise. L_e and $\mathcal{R}$ are periodically updated jointly by backpropagation on this dataset.

Multi-goal RL agent. Our agent is controlled by a goal-conditioned policy Π (Schaul et al., 2015) based on the MA architecture (see Figure 10b). It uses an attention vector $\boldsymbol{\beta}^g$, a shared network NN^{Π}, a sum aggregation and a mapper NN^{a} that outputs the actions. Similarly, the critic produces action-values via $\boldsymbol{\gamma}^g$, NN^{Q} and $\text{NN}^{\text{a-v}}$ respectively, see equations below. Both are trained using DDPG (Lillicrap et al., 2015), although any other off-policy algorithm can be used. Figure 10b shows the detailed architecture of the reward function. The overall function can be written as

$$\Pi(\mathbf{s}, g) = \text{NN}^{\text{a}}(\sum_{i \in [1..N]} \text{NN}^{\Pi}(\mathbf{s}_{obj(i)} \odot \boldsymbol{\beta}^g))$$

$$(\mathbf{s}, \mathbf{a}, g) = \text{NN}^{\text{a-v}}(\sum_{i \in [1..N]} \text{NN}^{Q}([\mathbf{s}_{obj(i)},\ \mathbf{a}] \odot \boldsymbol{\gamma}^g))$$

Hindsight Experience Replay (HER). Our agent uses a form of Hindsight Experience Replay (Andrychowicz et al., 2017). In multi-goal RL with discrete sets of goals, HER is traditionally used to modify transitions sampled from the replay buffer. It replaces originally targeted goals by others randomly selected from the set of goals (Andrychowicz et al., 2017; Mankowitz et al., 2018). This enables to transfer knowledge between goals, reinterpreting trajectories in the light of new goals. In that case, a reward function is required to compute the reward associated to that new transition (new goal). To improve on random goal replay, we favor goal substitution towards goals that actually match the state and have higher chance of leading to rewards. In IMAGINE, we scan a set of 40 goal candidates for each transition, and select substitute goals that match the scene when possible, with probability $p = 0.5$.

11.2. Implementation details

Reward function inputs and hyperparameters. The following provides extra details about the inputs.

- *The object-dependent sub-state* $\mathbf{s}_{obj(i)}$ contains information about both the agent's body and the corresponding object i:

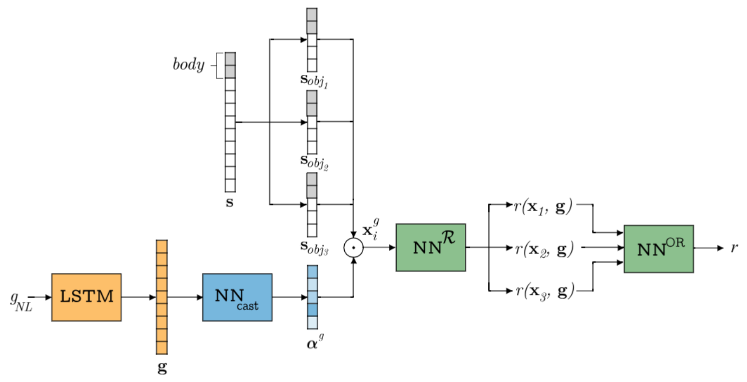
$$\mathbf{s}_{obj(i)} = [\mathbf{o}_{body}, \Delta\mathbf{o}_{body}, \mathbf{o}_{obj(i)}, \Delta\mathbf{o}_{obj(i)}]$$

  where $\mathbf{o}_{body}$ and $\mathbf{o}_{obj(i)}$ are body- and obj_i-dependent observations, and $\Delta\mathbf{o}^t_{body} = \mathbf{o}^t_{body} - \mathbf{o}^0_{body}$ and $\Delta\mathbf{o}^t_{obj(i)} = \mathbf{o}^t_{obj(i)} - \mathbf{o}^0_{obj(i)}$ measure the difference between the initial and current observations.

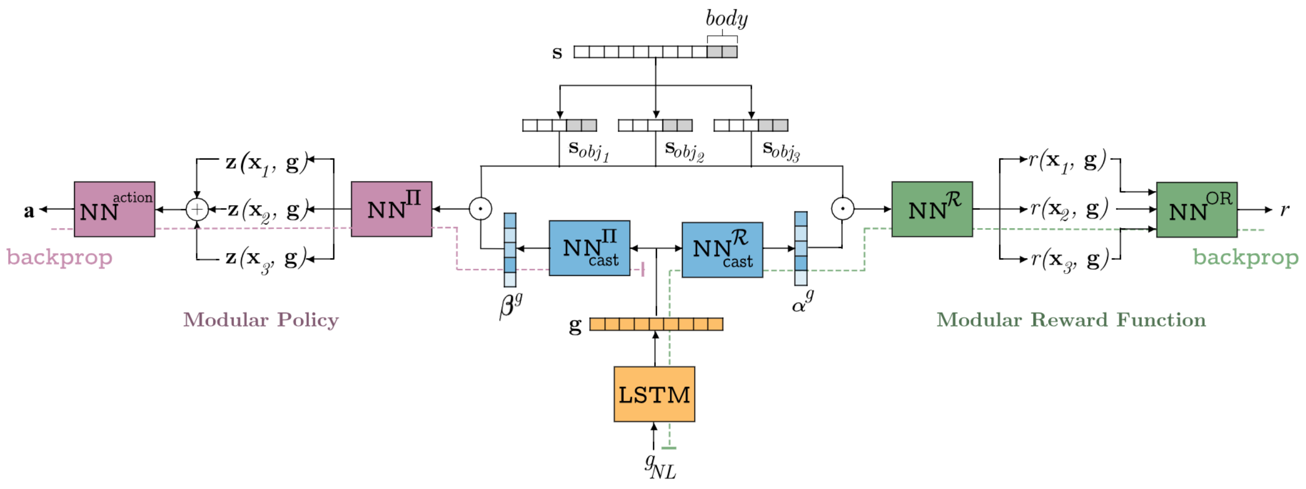
- *The attention vector* $\boldsymbol{\alpha}^g$ that is integrated with $\mathbf{s}_{obj(i)}$ through an Hadamard product to form the model input: $\mathbf{x}^g_i = \mathbf{s}_{obj(i)} \odot \boldsymbol{\alpha}^g$. This attention vector is a simple mapping from $\mathbf{g}$ to a vector of the size of $\mathbf{s}_{obj(i)}$ contained in $[0, 1]^{size(\mathbf{s}_{obj(i)})}$. This cast is implemented by a one-layer neural network with sigmoid activations NN^{cast} such that $\boldsymbol{\alpha}^g = \text{NN}^{\text{cast}}(\mathbf{g})$.

For the three architectures the number of hidden units of the LSTM and the sizes of the hidden layers of fully connected networks are fixed to 100. NN parameters are initialized using He initialization (He et al., 2015) and we use one-hot word encodings. The LSTM is implemented using `rnn.BasicLSTMCell` from tensorflow 1.15 based on Zaremba et al. (2014). The states are initially set to zero. The LSTM's weights are initialized uniformly from $[-0.1, 0.1]$ and the biases initially set to zero. The LSTM use a $tanh$ activation function whereas the NN are using ReLU activation functions in their hidden layers and sigmoids at there output.

Reward function training schedule. The architecture are trained via backpropagation using the Adam Optimizer (Kingma & Ba, 2014). The data is fed to the model in batches of 512 examples. Each batch is constructed so that



(a) $MA^{\mathcal{R}}$



(b)

Figure 10: **Modular attention architectures**: (a) *Modular-attention* reward function ($MA^{\mathcal{R}}$) (b) Complete *Modular-attention* architecture MA reward + MA policy. In both figures, the reward function is represented on the right in green, the policy on the left in pink, the language encoder in the bottom in yellow and the attention mechanisms at the center in blue.

it contains at least one instance of each goal description g_{NL} (goals discovered so far). We also use a modular buffer to impose a ratio of positive rewards of 0.2 for each description in each batch. When trained in parallel of the policy, the reward function is updated once every 1200 episodes. Each update corresponds to up to 100 training epochs (100 batches). We implement a stopping criterion based on the F_1-score computed from a held-out test set uniformly sampled from the last episodes (20% of the last 1200 episodes (2 epochs)). The update is stopped when the F_1-score on the held-out set does not improve for 10 consecutive training epochs.

RL implementation and hyperparameters. In the policy and critic architectures, we use hidden layers of size 256 and ReLU activations. Attention vectors are cast from goal embeddings using single-layer neural networks with sigmoid activations. We use the He initialization scheme for (He et al., 2015) and train them via backpropagation using the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) (Kingma & Ba, 2014).

Our learning algorithm is built on top of the OpenAI Baselines implementation of HER-DDPG.[1] We leverage a parallel implementation with 6 actors. Actors share the same policy and critic parameters but maintain their own memory and conduct their own updates independently. Updates are then summed to compute the next set of parameters broadcast to all actors. Each actor is updated for 50 epochs with batches of size 256 every 2 episodes of environment interactions. Using hindsight replay, we enforce a ratio $p = 0.5$ of transitions associated with positive rewards in each batch. We use the same hyperparameters as Plappert et al. (2018).

[1]The OpenAI Baselines implementation of HER-DDPG can be found at https://github.com/openai/baselines, a link to our Github repository will be added in the final version.

Computing resources. The RL experiments contain 8 conditions of 10 seeds each, and 4 conditions with 5 seeds (SP study). Each run leverages 6 cpus (6 actors) for about 36h for a total of 2.5 cpu years. Experiments presented in this paper requires machines with at least 6 cpu cores.