How to evaluate communication

Maxime Toquebiau[1,2], Nicolas Bredeche[2], Faïz Ben Amar[2], Jae Yun Jun Kim[1]

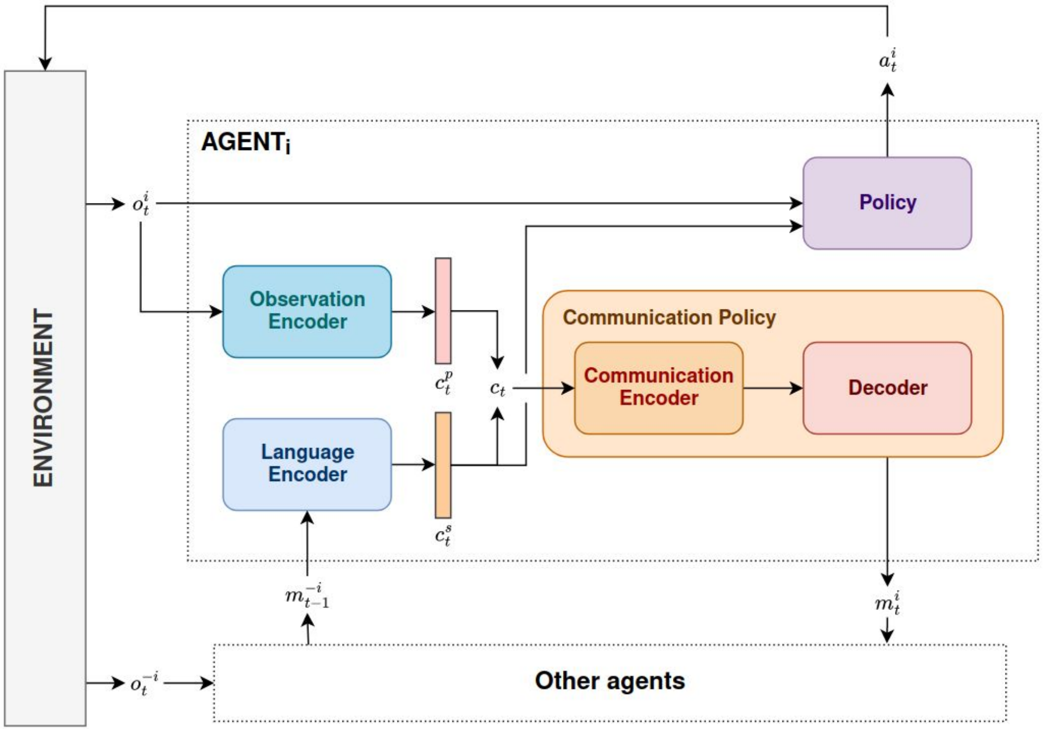
[1]ECE Paris
[2]ISIR, Sorbonne Universités

November 23th, 2023

Table of Content

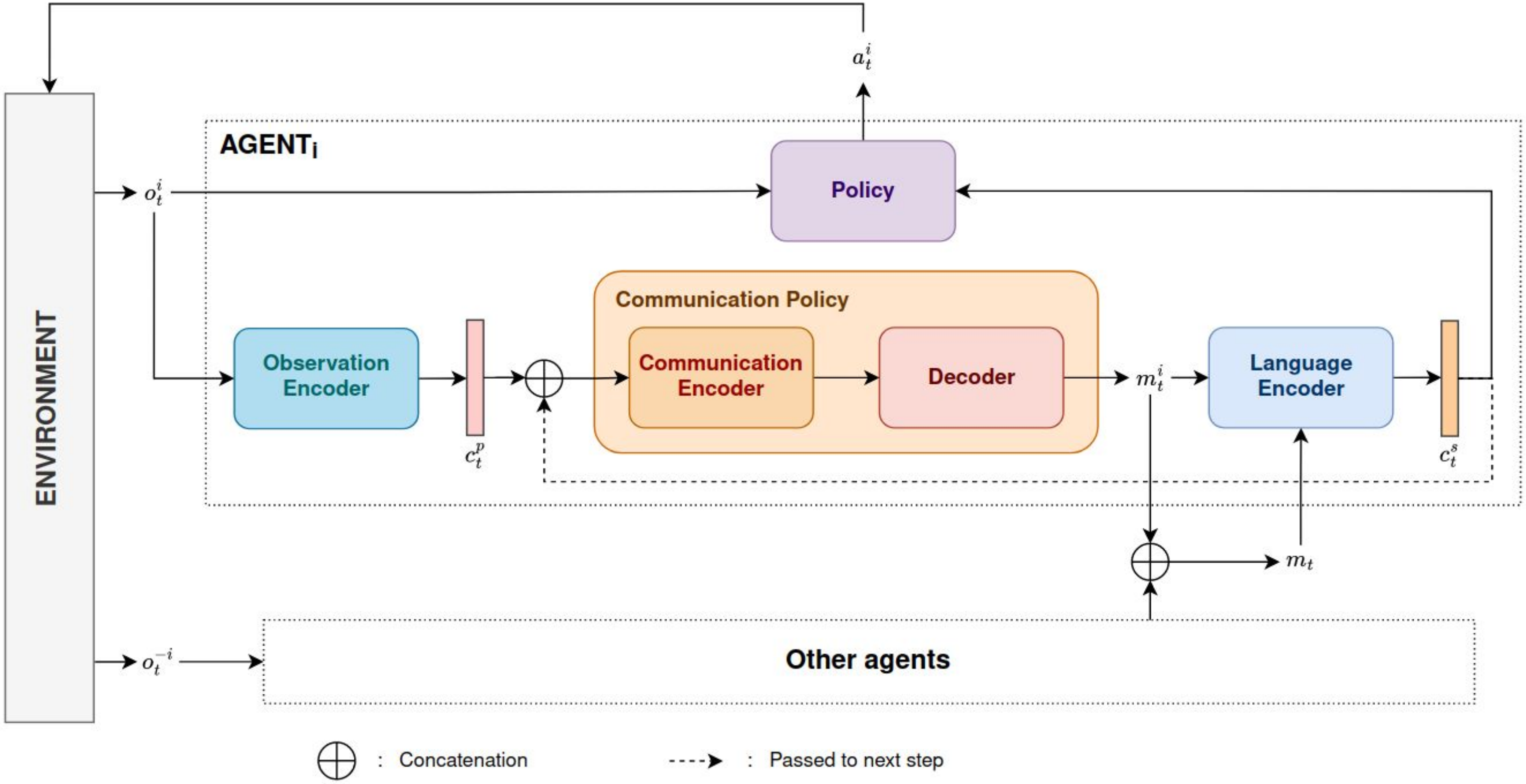
Architecture for communicating with language

Old version



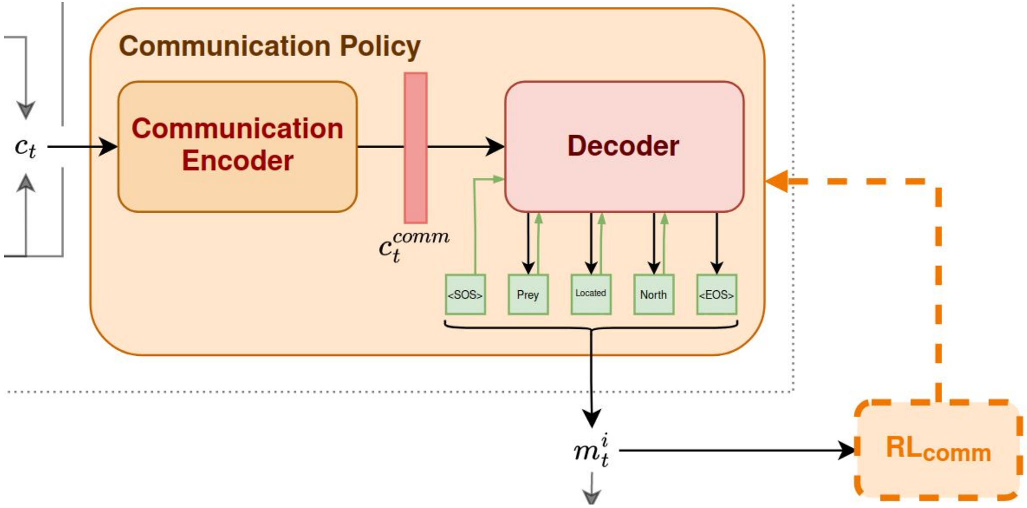
Architecture for communicating with language

Update



Communication Policy

Fine-tuning the Decoder



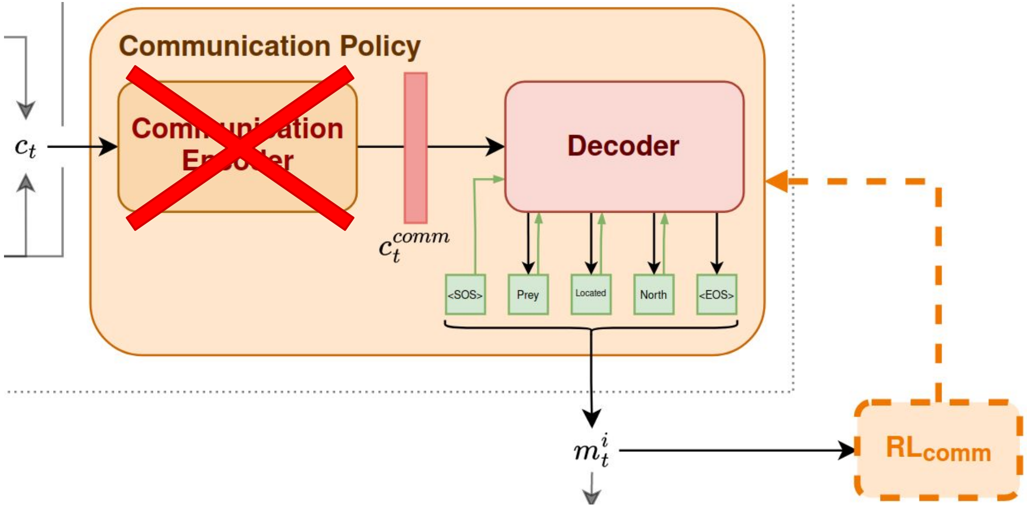
Fine-tuning the decoder with PPO:

- **Task:** Generating messages (sequences of tokens)
- **States:** previous token
- **Actions:** next token
- **Reward:**
  - env reward
  - + penalty for message length
  - + penalty for diverging from pre-trained decoder [1]

[1] Ouyang et al., *Training language models to follow instructions with human feedback*, 2022.

Communication Policy

Fine-tuning the Decoder



Simplifying the architecture:

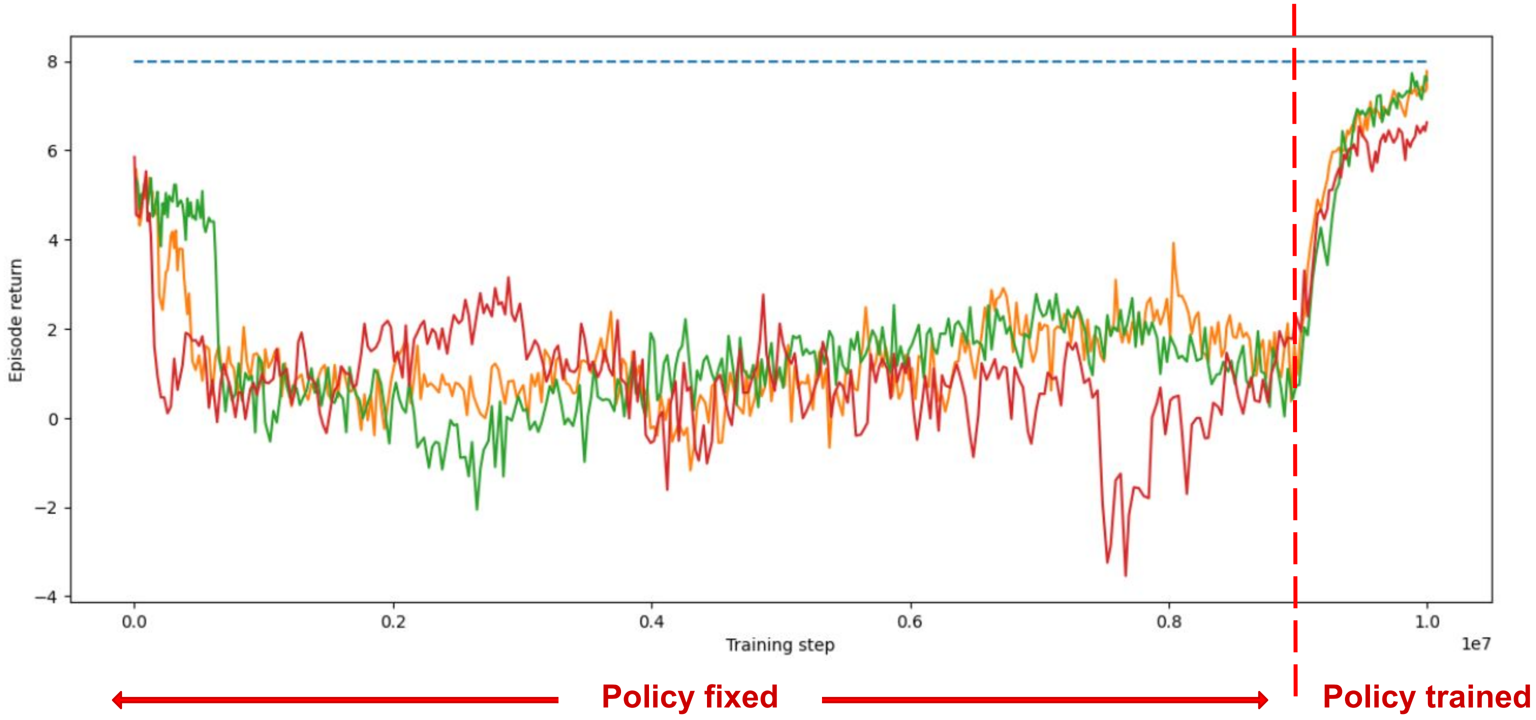
- No communication encoder
- Take observation encoding as input to the decoder

Reward:

- Got after generating a token:
  - penalty for generating a token (-0.01)
  - penalty for KL divergence from pre-trained decoder
- Got after generating the whole message:
  - reward from environment r_t

Communication Policy

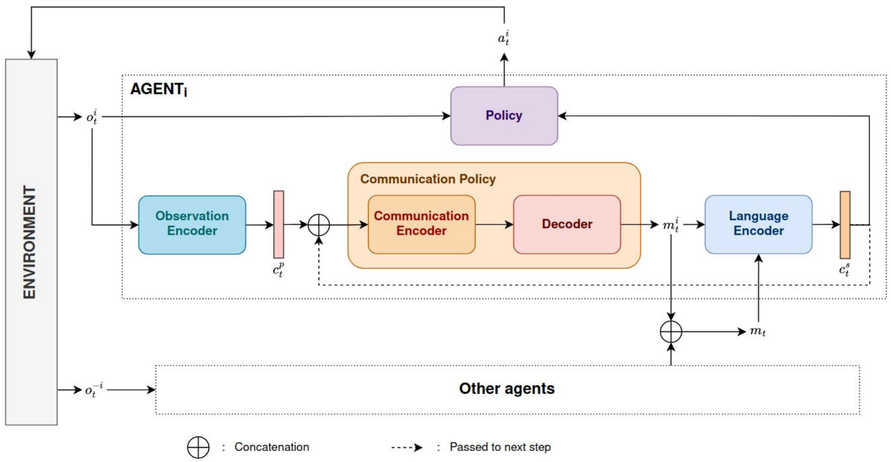
Fine-tuning the Decoder: First Results



Communication Policy

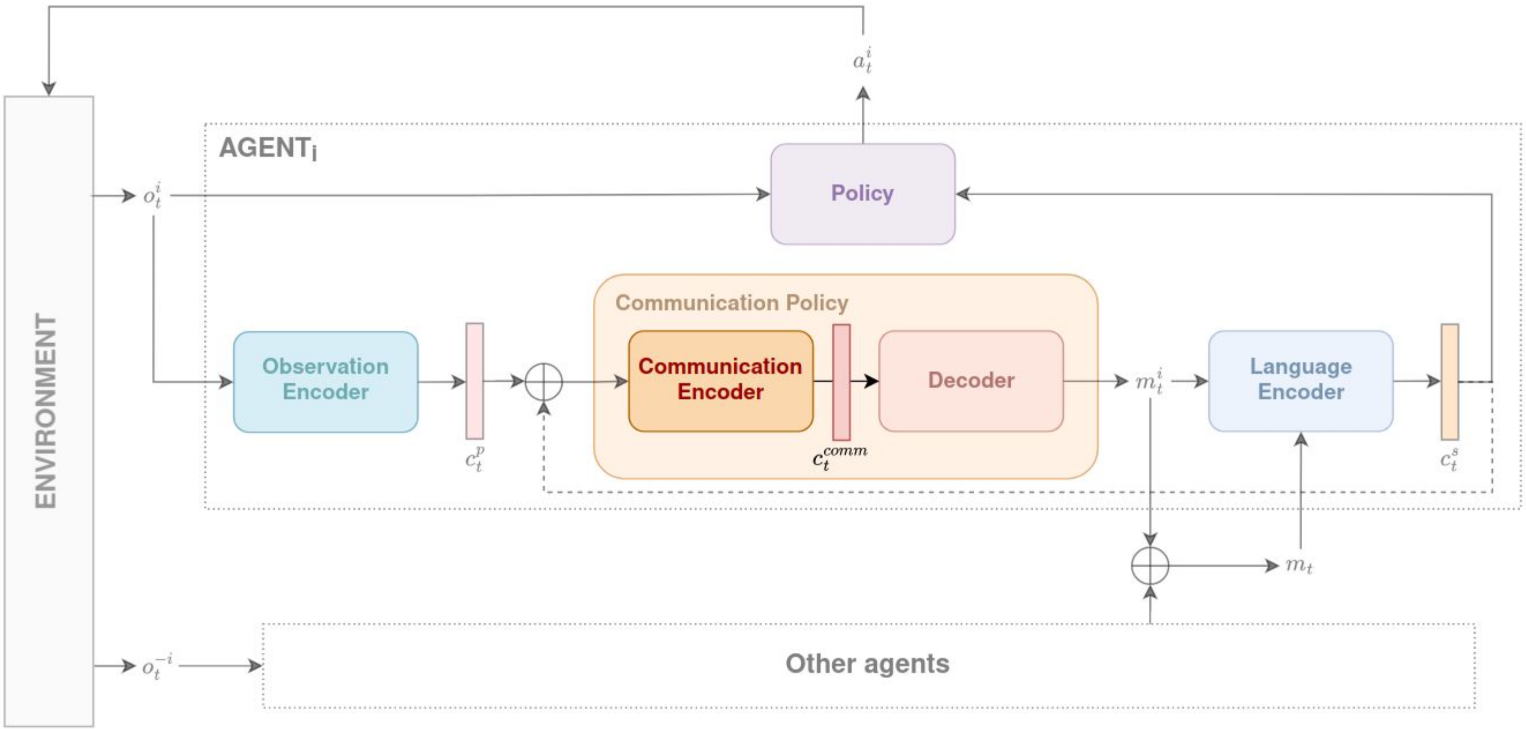
Fine-tuning the Decoder: Issues

- Message rewards from the environment
    - Reward r_t doesn't reflect the quality of the message
    - Need to compute the return got from this message => train at the end of each episode
- Task may be too hard
- Need a communication encoder to have information about previous messages



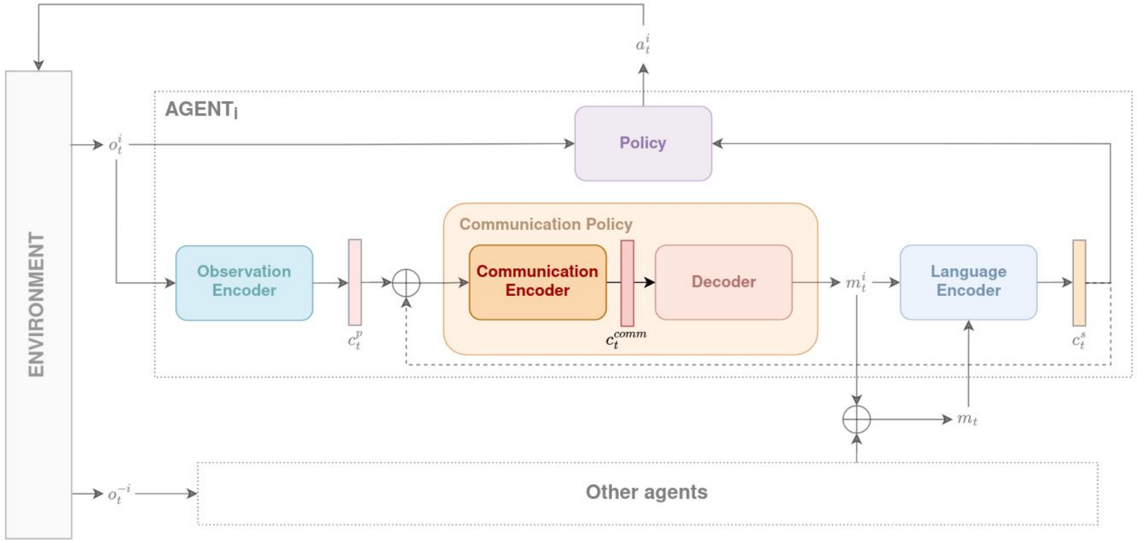
Communication Policy

Learning a Communication Encoder



Communication Policy

Learning a Communication Encoder

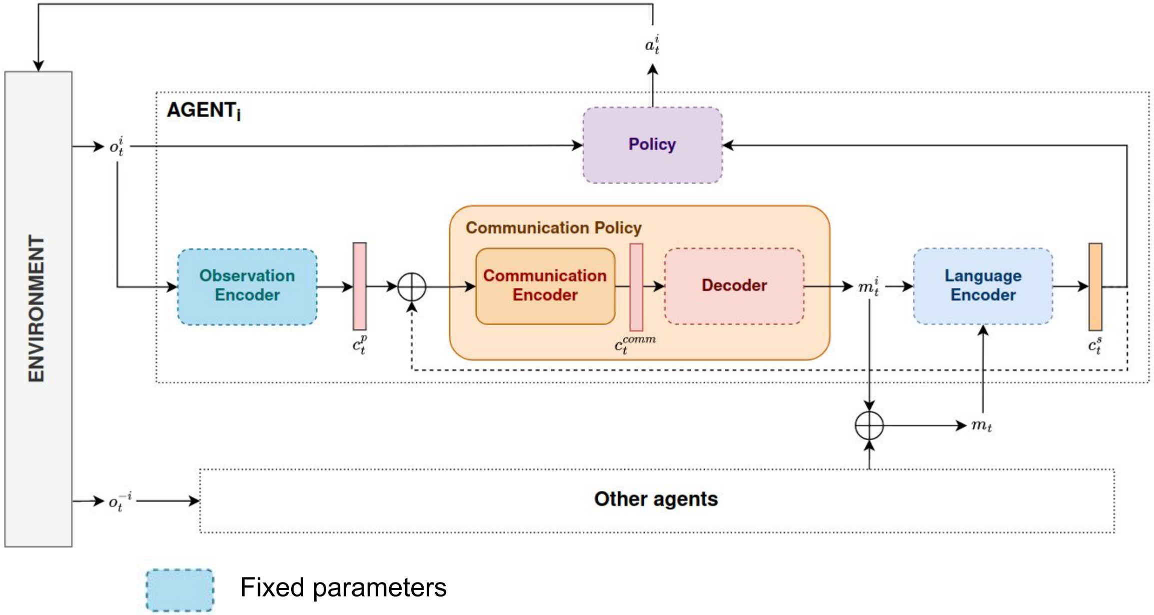


Communication Encoder as a Policy:

- **Task:** Choose what to communicate
- **Reward:** Quality of the message generated by the Decoder
- **Multi-agent setting:** Quality of the message depends on other agents messages and actions

Communication Policy

Communication Encoder learnt with MAPPO

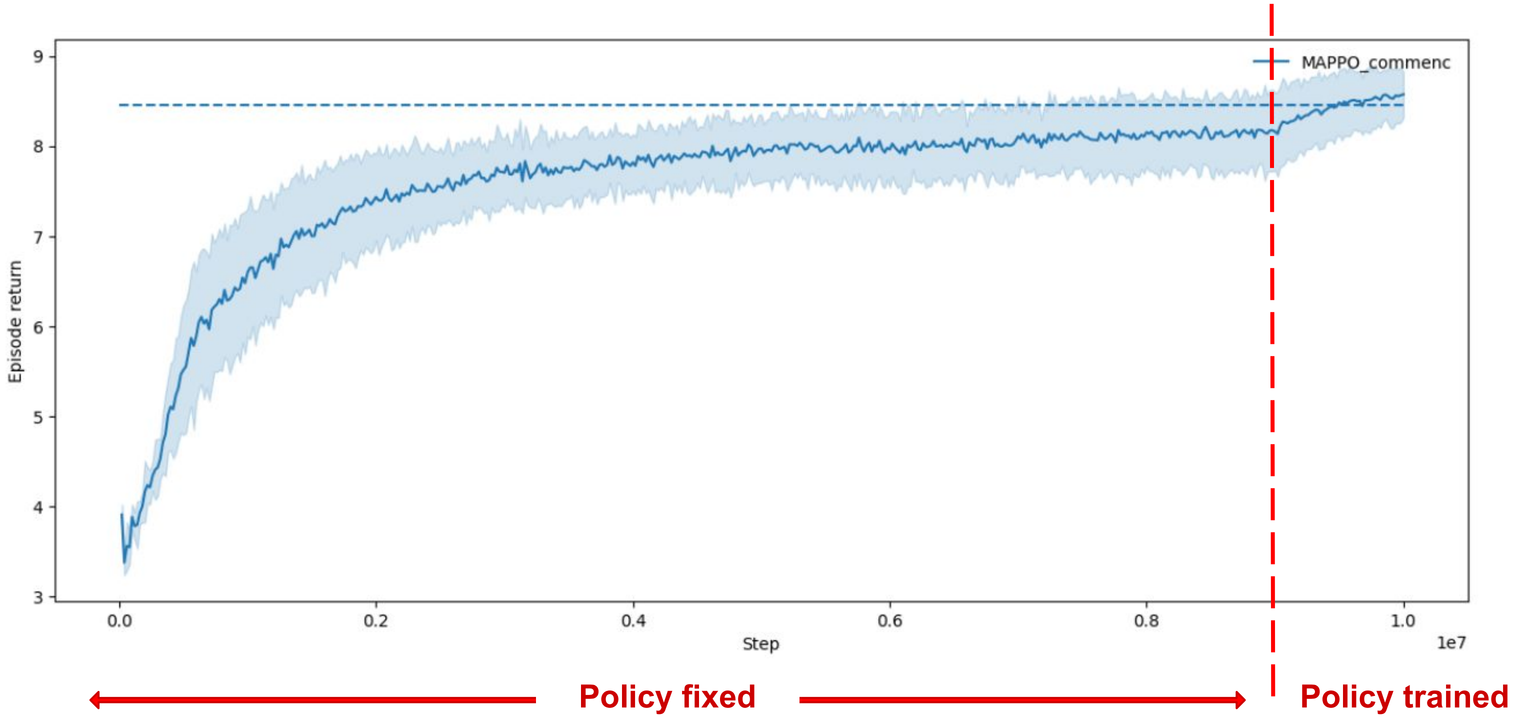


Communication Encoder with MAPPO:

- **State:** personal and social contexts
- **Action:** a communication context (vector of floats)
- **Reward:**
  - env reward
  - + penalty for message length
  - + penalty for distance from personal context.
- **Algo:** MAPPO (or any other MADRL) to learn a policy and value in a multi-agent setting

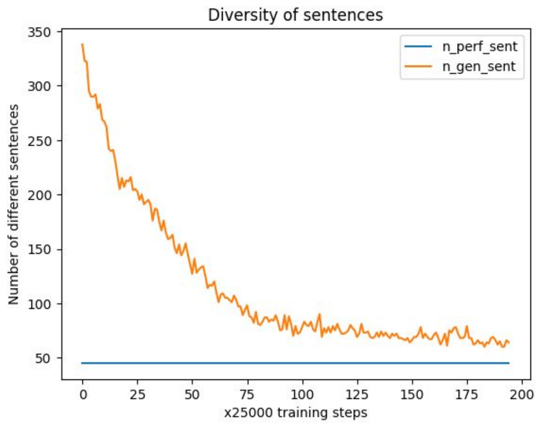
Communication Policy

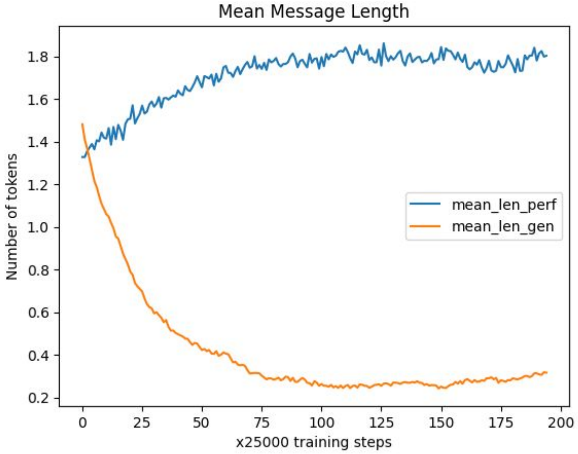
MAPPO Communication Encoder: First Results

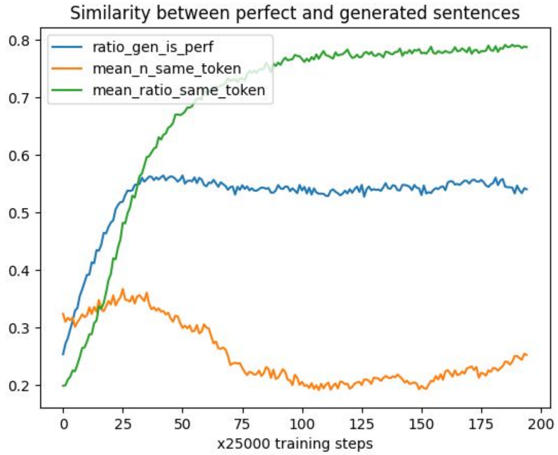


Communication Policy

MAPPO Communication Encoder: First Results



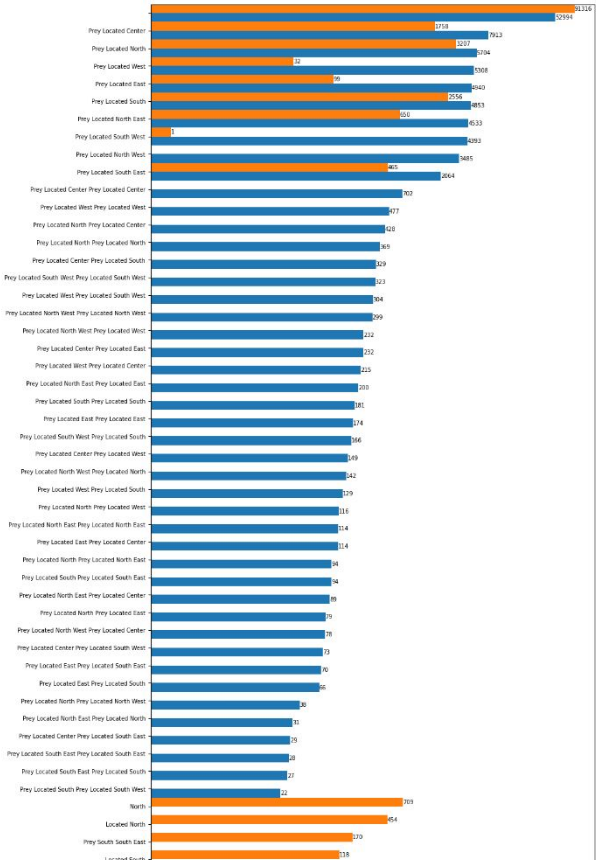


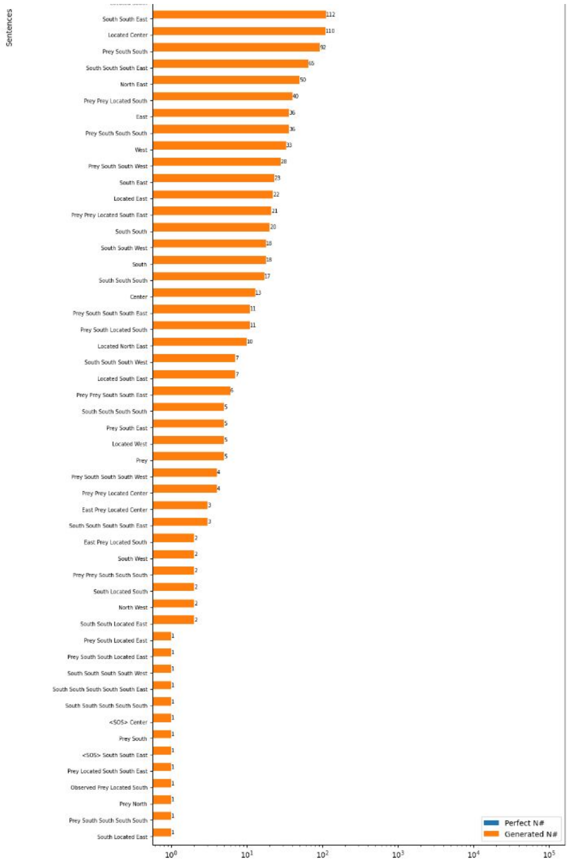


- `n_perf_sent` : Number of different perfect sentence
- `n_gen_sent` : Number of different generated sentence
- `mean_len_perf` : Average length of perfect sentences
- `mean_len_gen` : Average length of generated sentences
- `ratio_gen_is_perf` : Ratio of generated sentences that are identical to the corresponding perfect sentences
- `mean_n_same_token` : Average number of similar tokens between generated and perfect sentences
- `mean_ratio_same_token` : Average ratio of similar tokens between generated and perfect sentences

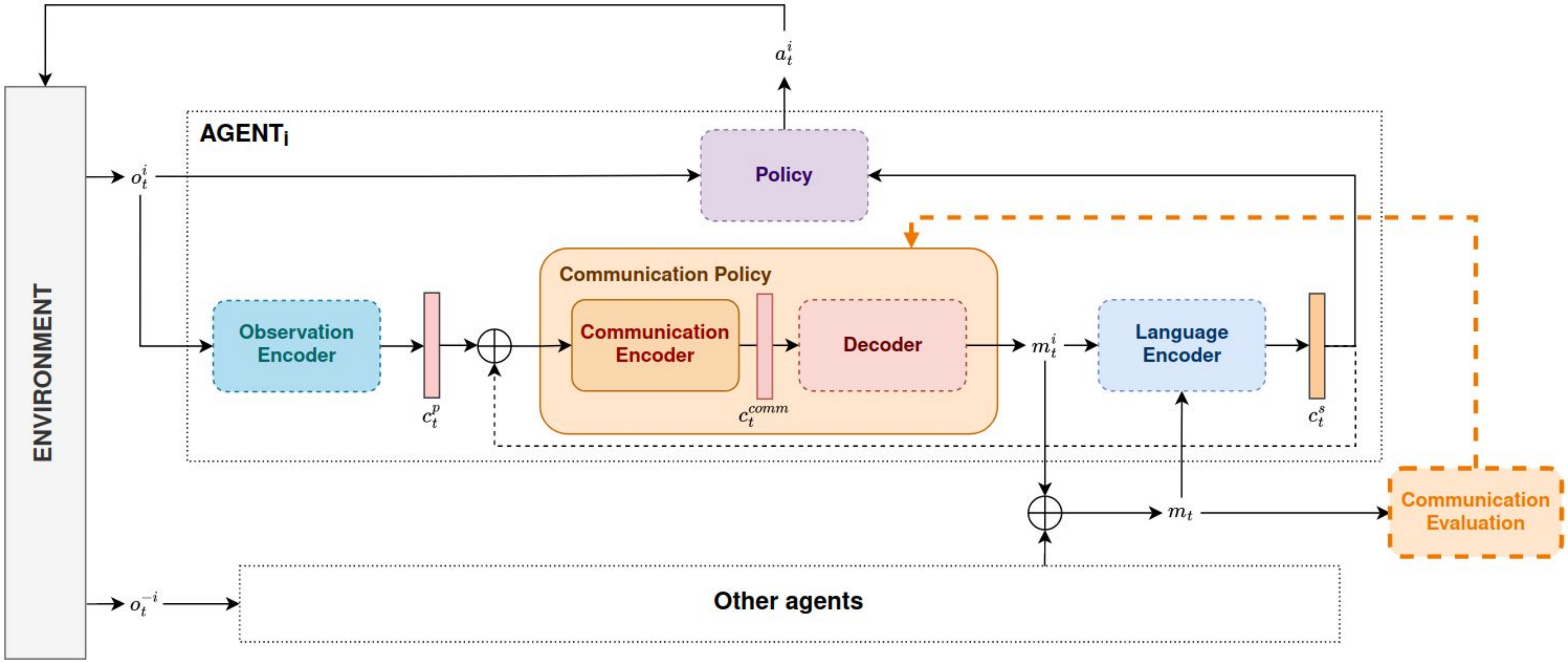
Communication Policy

MAPPO Communication Encoder: First Results



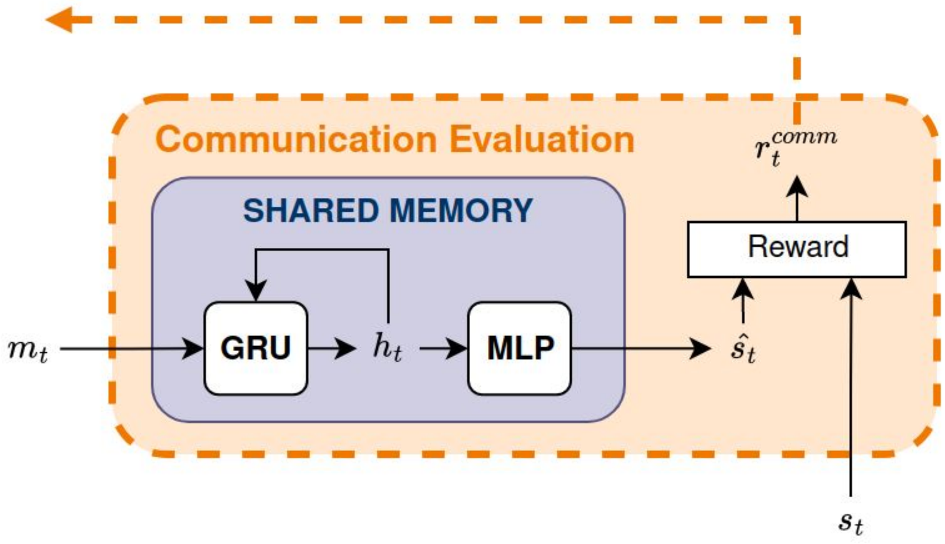


Evaluating the quality of messages



Evaluating the quality of messages

Shared Memory



Shared Memory for training the Communication Policy:

- **Learning task:** Predict the global state
- **Reward:** Reward messages that communicate valuable information a about the current state of the environment
  - MSE: $r_t^{comm} = \delta_t = ||\hat{s}_t - s_t||_2$
  - Shaping: $r_t^{comm} = \delta_{t-1} - \delta_t$
- Learnt during pre-training phase

Next steps

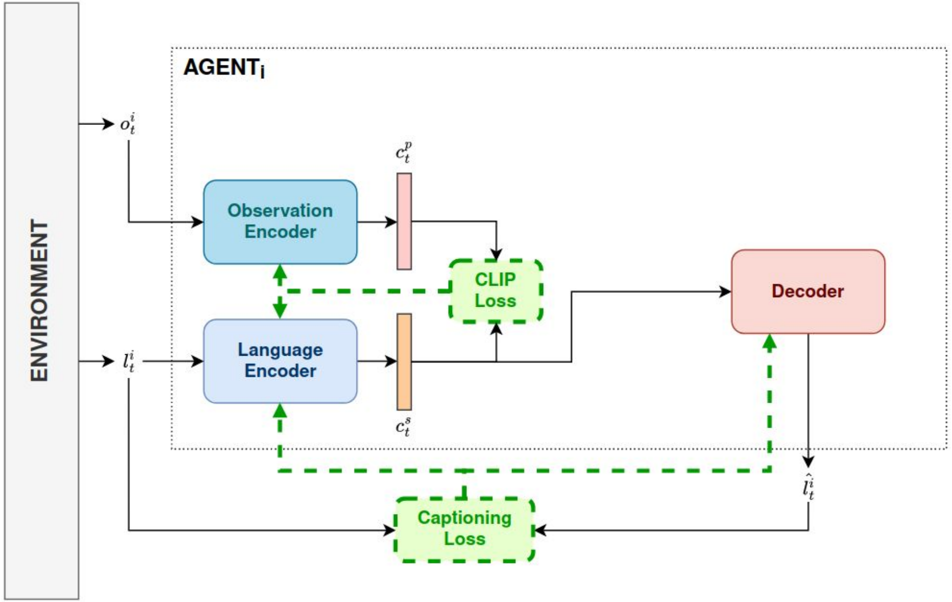
- Pre-training with Shared Memory
- Fine-tuning communication with Shared Memory
- Moving to BabyAI
- Submit to conf end of 2023/start of 2024
    - IJCAI (January 17th)
    - ICML (February 1st)
    - RSS (February 2nd)
    - Practical applications of Agents and Multi-Agent System (rank B) (deadline TBC)

Thank you for you attention !

Questions ?

Training Process

Phase 1: Learning to ground and generate language



CLIP Loss

With the **Observation Encoder** $\omega : \mathbb{R}^N \rightarrow \mathbb{R}^M$,
and the **Language Encoder** $\lambda : \mathbb{R}^{L \times V} \rightarrow \mathbb{R}^M$,

the grounding objective is:

$$J(\theta_\omega, \theta_\lambda) = max[cosim(\omega(o_k), \lambda(l_k))]$$

+

Captioning Loss

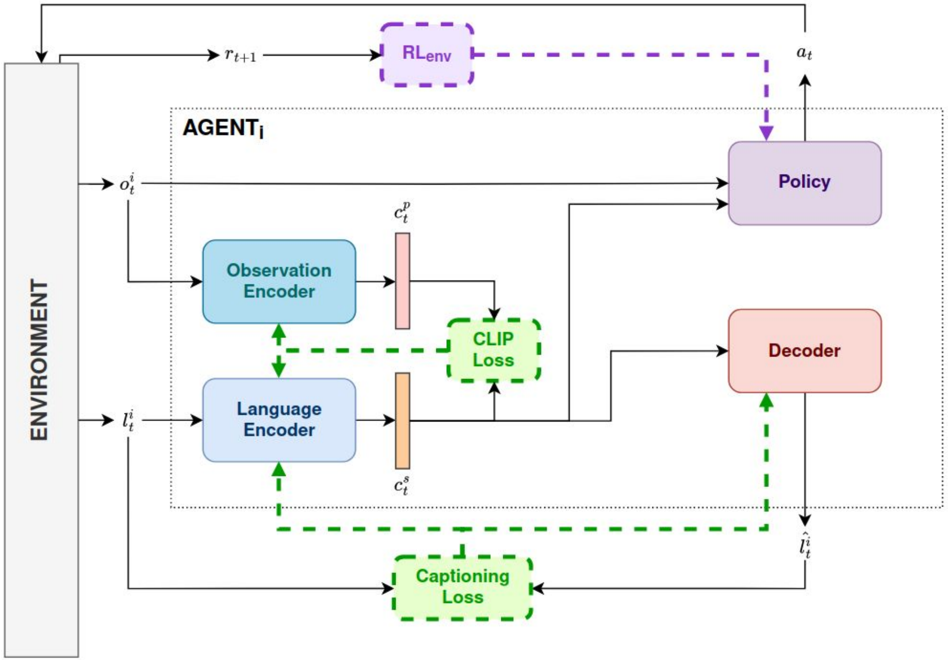
With the **Language Encoder** $\lambda : \mathbb{R}^{L \times V} \rightarrow \mathbb{R}^M$,
and the **Decoder** $\delta : \mathbb{R}^M \rightarrow \mathbb{R}^{L \times V}$,

the captioning objective is:

$$J(\theta_\lambda, \theta_\delta) = min\left[\frac{1}{N}\sum_{i=0}^{N}(\hat{l}_i - l_i)^2\right]$$

Training Process

Phase 2: Learning a working policy with “perfect messages”



CLIP Loss

+

Captioning Loss

+

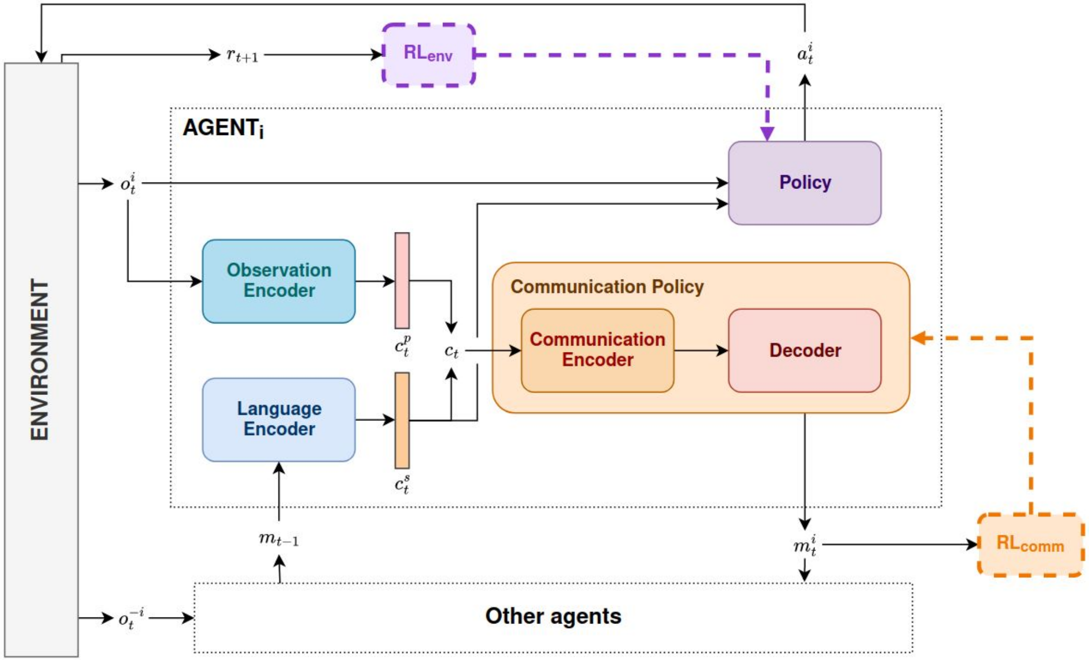
RL from Environment rewards

Training with any reinforcement learning algorithm using rewards from the environment.

Most likely **MAPPO**, so with, for each agent i, an **Actor** $\pi^i_{\phi_{act}}(o^i_t) = a^i_t$ and a **Critic** $V^i_{\phi_{crit}}(\mathbf{o}_t)$, learning to maximise rewards r_t from the environment by optimising the PPO objective

Training Process

Phase 3: Learning the communication policy



Fixed parameters

RL$_{\text{env}}$

+

RL from Communication quality

We train a **Communication Encoder** $\mathbb{C} : \mathbb{R}^{2M} \rightarrow \mathbb{R}^{M}$ to learn to choose which information to share, and we fine-tune the **Decoder** δ (pre-trained on captioning) to generate useful messages.

The reward for communication quality can be defined as,

$$r_t^{comm} = r_t - \beta \log\left(\frac{\delta_{FT}}{\delta_{PT}}\right) + \ldots$$

with δ_{FT} the current fine-tuned version of the decoder and δ_{PT} the pre-trained version of the decoder with fixed parameters. We use PPO for learning the communication policy.