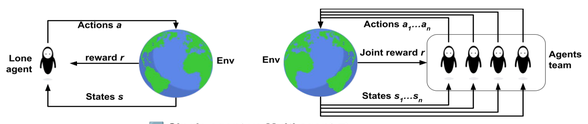
MACTAS: Self-Attention-Based Module for Inter-Agent Communication in Multi-Agent Reinforcement Learning

Maciej Wojtala, Bogusz Stefańczyk, Dominik Bogucki, Łukasz Lepak , Jakub Strykowski, Paweł Wawrzyński

Cooperative Multi-Agent Reinforcement Learning (CMARL)

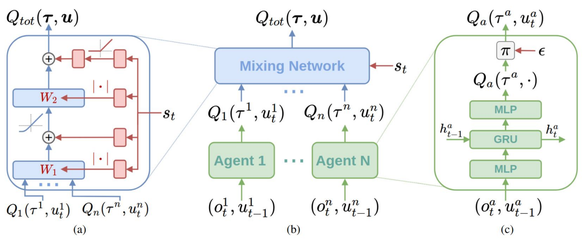
- **Each** agent sees their state and picks an action
- Goal to maximize **team total** reward
- The space of decisions grows combinatorially with the number of agents; the need for decoupling
- Exploration impacts the team of agents
- Teaches cooperation and team strategies
- Communication between agents is possible

A mixing network is a neural network that **combines individual agents' Q-values into a global Q-value** in a way that preserves maximum, enabling centralized training with decentralized execution paradigm (**CTDE**). In our experiments, we use **VDN**, **QMIX**, and **QPLEX**. There are others, like **QTRAN**, **Qatten**, **NA2Q**, **SHAQ** and many more.



⬅ Single agent vs Multi agent ➡

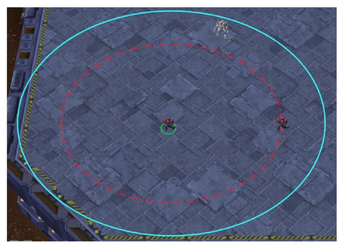
Diagram of environment<->agent(s) interaction in a single timestep



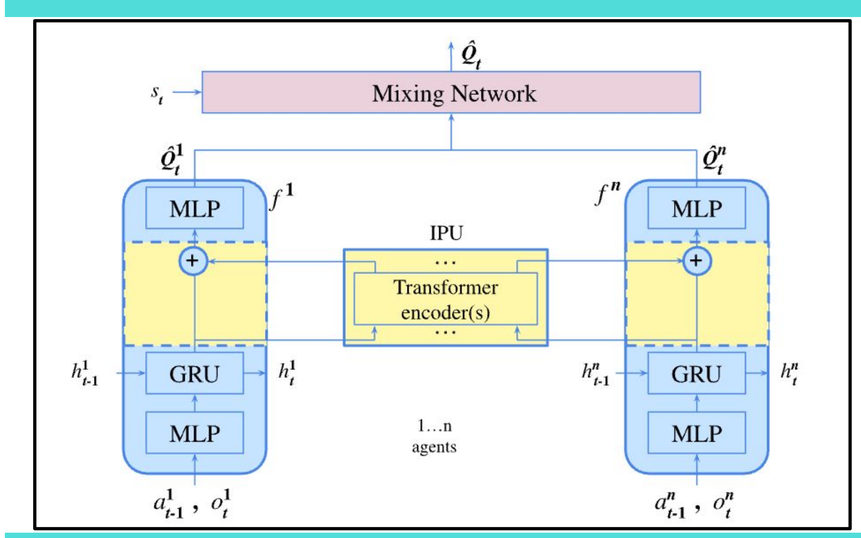
Sample mixing network architecture: Monotonic Value Function Factorisation (QMIX)

Starcraft Multi-Agent Challenge (SMAC and SMAC v2)

- **SMAC** is a widely used MARL algorithms benchmarking environment. It is based on the popular video game Starcraft 2.
- Algorithm-controlled units compete against the game's built-in AI.
- Various **battle scenarios** - no economy and no buildings.
- **Observation space** is an embedding vector of the units' area in sight, which includes the relative position of:
  - hostile units with their type, health, and shield,
  - friendly units with their type, health, and shield,
  - difficult terrain or obstacles.
- **Action space** includes discrete actions:
  - moving in 4 directions (up/down/left/right)
  - waiting or not operating,
  - IDs of the enemies to attack or friendly units to heal.
- **SMAC v2** is an improved version of the benchmark, it includes the randomization of units' initial positions, scenarios with randomized unit types, and customization options for units' observations.



MACTAS: Multi-Agent Communication via Transformer on Agents' States

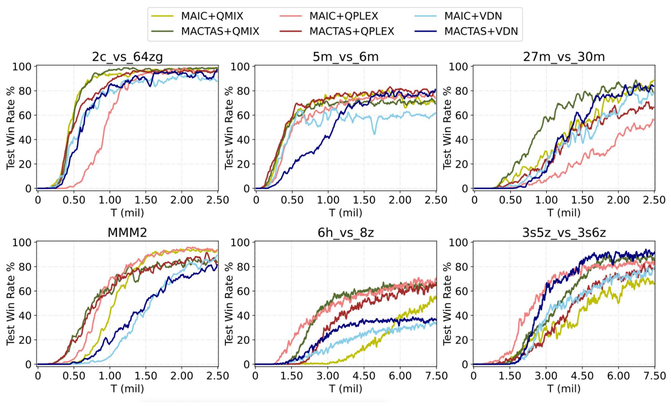


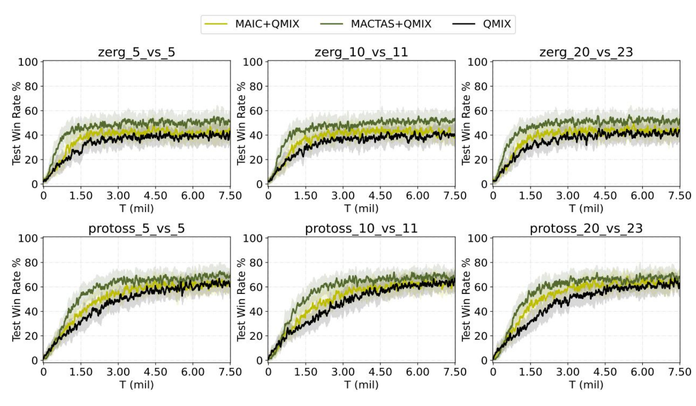
- We work with the agents' architecture,which consists of the input MLP, the recurrent layer, and the output MLP.
- After the recurrent layer, we add a central communication layer called the **Intermediate Processing Unit** (**IPU**).
- The IPU consists of **stacked Transformer encoder layers**.
- The IPU obtains as inputs the encoded history of agents' observations and actions.
- The output of the IPU is **messages directed individually to each agent**.

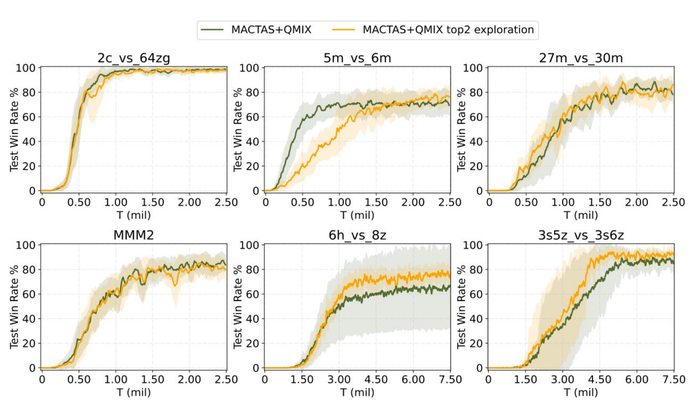
Top2 exploration

- Independent ε-greedy exploration may get stuck in suboptimal Nash equilibria.
- **We propose a combination of ε-greedy and Boltzmann exploration**.
- Each agent with probability ε chooses independently a random action and, with probability 1 - ε, it chooses from the top k of its actions (according to its local action-value function) using the Boltzmann strategy.
- Using k = 1 is equivalent to the pure ε-greedy policy. Selecting a small k leads to exploring a region that is considered quite good by the agent. We observe that k = 2 works well in practice.
- The **temperature of the Boltzmann exploration controls the amount of extra exploration** that we want to add.

Results







- We compare **MACTAS** with the state-of-the-art algorithm **MAIC** (Yuan et al. 2022), and bare mixing networks on **6 SMAC maps** in combinations with 3 mixer algorithms: **QMIX**, **QPLEX**, **VDN**,
- In **MAIC**, the state-of-the-art algorithm we use for comparison, agents explicitly model their teammates and use it to bias their value function.
- Various ultra-hard SMAC maps show MACTAS achieves better results in **9/18** cases - MAIC **3/18, 6 draws** - when measured by the area under the learning curve; and in **6/18** - MAIC **3/18, 9 draws** - measured by final performance.
- We test the behaviour of MACTAS, MAIC, and the bare mixer with QMIX as the mixer on **6 SMACv2 maps**. MACTAS performs better than MAIC, and bare QMIX on all maps. Improvement of MACTAS over MAIC is similar in to the differences between MAIC and QMIX.
- The new exploration scheme improves the results on the two most challenging SMAC maps, slows down the training on one small map, and does not affect the three other small maps.

TL;DR

- **Multi-Agent Communication via Transformer on Agents' States** (MACTAS) introduces a self-attention-based communication channel module for MARL algorithms.
- MACTAS **works with any action-value function decomposition method** (e.g., QMIX, QPLEX, VDN).
- Our solution maintains a **fixed number of trainable parameters regardless of the number of agents**.
- Extensive experiments on the SMAC show that MACTAS achieves state-of-the-art performance on several hard and super-hard scenarios, outperforming the previous best method.
- Additionally, we propose **a hybrid ε-greedy + Boltzmann exploration** scheme to mitigate suboptimal equilibria.

Sample strategies learned

MACTAS learns adaptive, human-like tactics - using positioning, focus fire, healing, and deception to overcome the enemies:

- **2c_vs_64zg** - Colossi use terrain to attack one Zergling group, then cross hills to avoid being surrounded.
- **27m_vs_30m** & **5m_vs_6m** - Marines form semicircles, focus fire on weak enemies, and retreat damaged units.
- **MMM2** - The Medivac switches healing targets, draws enemy fire when useful, and retreats when low on health.
- **6h_vs_8z** - Hydralisks kite Zealots by attacking from range while retreating and splitting into groups.
- **3s5z_vs_3s6z**: One Stalker acts as a decoy to lure enemies, allowing others to ambush and eliminate isolated targets.

2c_vs_64zg

27m_vs_30m

MMM2

6h_vs_8z

3s5z_vs_3s6z

Acknowledgements: This work was funded by IDEAS Research Institute and IDEAS NCBR. We gratefully acknowledge the Polish high-performance computing infrastructure PLGrid (HPC Center: ACK Cyfronet AGH) for providing computer facilities and support within the computational grant no. PLG/2025/018560 and no. PLG/2025/017992. We thank the University of Warsaw for providing access to the computing infrastructure.