Technische Universität Berlin

Notes

BlUB

Sascha Lange

Meta Blub

Let $\mathcal{S}$ denote state space and $\mathcal{A}$ denote action space. Let $\mathcal{I} = \{1, ..., P\}$ be the set of players and $\mathcal{P} = \{N_1, ..., N_P\}$ the set of corresponding (neural network) function approximators, where $N_i : \mathcal{S} \mapsto \mathcal{A}$, $N_i(s) = a$, corresponds to player i.

Playing with a new teammate p, the goal is to predict the action a that p will take, facing a new state s. Based on the pool $\mathcal{P}$ of pretrained agents, we will define the notion of *consistency* of two states, that will help us building a simple classifier to do so.

Definition 1 (consistency)**.** *For $s, s' \in \mathcal{S}$, let*

$$consistency_{\mathcal{P}}(s, s') = \frac{1}{P} \cdot |\{i \in \mathcal{P} : N_i(s) = N_i(s')\}|.$$

An agent $i \in \mathcal{I}$ is said to be consistent in (s, s'), if $N_i(s) = N_i(s')$.

We now define a simple distance function, which we can minimize later, to predict an action that p will take:

Definition 2 (Distance)**.** *For $s, s' \in \mathcal{S}$, let the distance be defined as*

$$d(s, s') = 1 - consisteny_{\mathcal{P}}(s, s').$$

After having observed data $D = \{(s_1, a_1), ..., (s_n, a_n)\}$ from a new teammate p, we can give the action classifier as

$$C_p(s) = \underset{a}{argmin}\{d(\tilde{s}, s) : (\tilde{s}, a) \in D\}.$$

So we guess as the action p takes in s, the action p took in the state $s_{i \leq n}$, with maximum $consistency_{\mathcal{P}}(s_{i \leq n}, s)$. However the suggested distance, does not account for the fact, that some players in $\mathcal{I}$ are more likely similar[1] to p, than others. When the consistency voting happens, we would like to give those agents more weight than others. Therefore, we introduce the following notion on the observed data D:

Definition 3 (agreement)**.** *For $s, s' \in \mathcal{S}, i, p \in \mathcal{P}$, let*

$$agreement_D(i, p) = \frac{2}{n(n-1)} \cdot |\{(s, s') \in D \times D : N_i(s) = N_i(s') \text{ and } N_p(s) = N_p(s')\}|.$$

The agreement of two agents i, p is higher, the more pairs $(s, s') \in D$ exist, that i and p are both consistent on. Under the assumption, that this implies the agents are more likely to perform the same actions in new states, we now define p-similarity of two states, as a weighted version of consistency:

Definition 4 (p-similarity)**.** *For $s, s' \in \mathcal{S}, i, p \in \mathcal{P}$, let*

$$similarity_p(s, s') = \frac{1}{P} \sum \{agreement_D(i, p) \in \mathbb{R} : i \text{ is consistent in } (s, s')\}. \quad (1)$$

Let C_θ denote our meta classifier. We want to learn $\theta = \theta_0$, such that

[1] similar w.r.t to which states they consider consistent

- after small number L of gradient steps on data D from agent A, to obtain θ_L, the network C_{θ_L} performs well on predicting actions of A

So we obtain updated network params after $i \leq L$ steps on D from A by

$$\theta_i^A = \theta_{i-1}^A - \alpha \Delta_\theta \mathcal{L}_A(C_{\theta_{i-1}^A})$$

for a **single** Task A, and thus the meta-objective becomes

$$\sum_{A \in POOL} \mathcal{L}_P(C_{\theta_L}^A) =: \mathcal{L}_{Meta},$$

where $\mathcal{L}_P$ denotes the loss on the hold out set corresponding to A. Both A and P are agents, but A denotes agents at training time and P denotes agents at test time, indicating that players can be humans. Note however, that **the different notation simply denotes disjoint data, but from the same agent P=A**.
Finally we have the outer loop update given by

$$\theta_0 = \theta_0 - \beta \Delta \mathcal{L}_{Meta}.$$

Using the idea of incorporating implicit soft cluster assignment (see slack) into the learning process we may obtain for C_θ the following architecture: