Technische Universität Berlin

Notes

BlUB

Sascha Lange

Meta Blub

Let $\mathcal{S}$ denote state space and $\mathcal{A}$ denote action space. Let $\mathcal{P} = \{1, ..., P\}$ be the set of players and $\mathcal{N} = \{N_0, ..., N_P\}$ the set of (neural network) function approximators, where $N_i : \mathcal{S} \mapsto \mathcal{A}$, $N_i(s) = a$, corresponds to player i.

Definition 1 (Agreement, stability)**.** *Let $i \in \mathcal{P}, s, s' \in \mathcal{S}$.*

i) Agreement between two states s, s' is defined as the ratio $\frac{1}{P} \cdot |\{i \in \mathcal{P} : N_i(s) = N_i(s')\}|$

ii) Stability of a state s is defined as the ratio $\frac{2}{P(P-1)} \cdot |\{(N_i, N_j) \in \mathcal{N} \times \mathcal{N} : N_i(s) = N_j(s), i \neq j\}|$

Then we can define

Definition 2 (Distance)**.** *Let $i \in \mathcal{P}, s, s' \in \mathcal{S}$.*

The distance d between two states is given by: $d(s, s') = 1 - Agreement(s, s')$

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

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

The goal is, to combine the two notions in Definition 1, to obtain a *similarity* of two states, relative to our teammate (for reasons I explained on slack). What I initially suggested was to do it manually (i.e. no NN training), however, I think this can be done using meta learning in the following way:

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

- 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

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

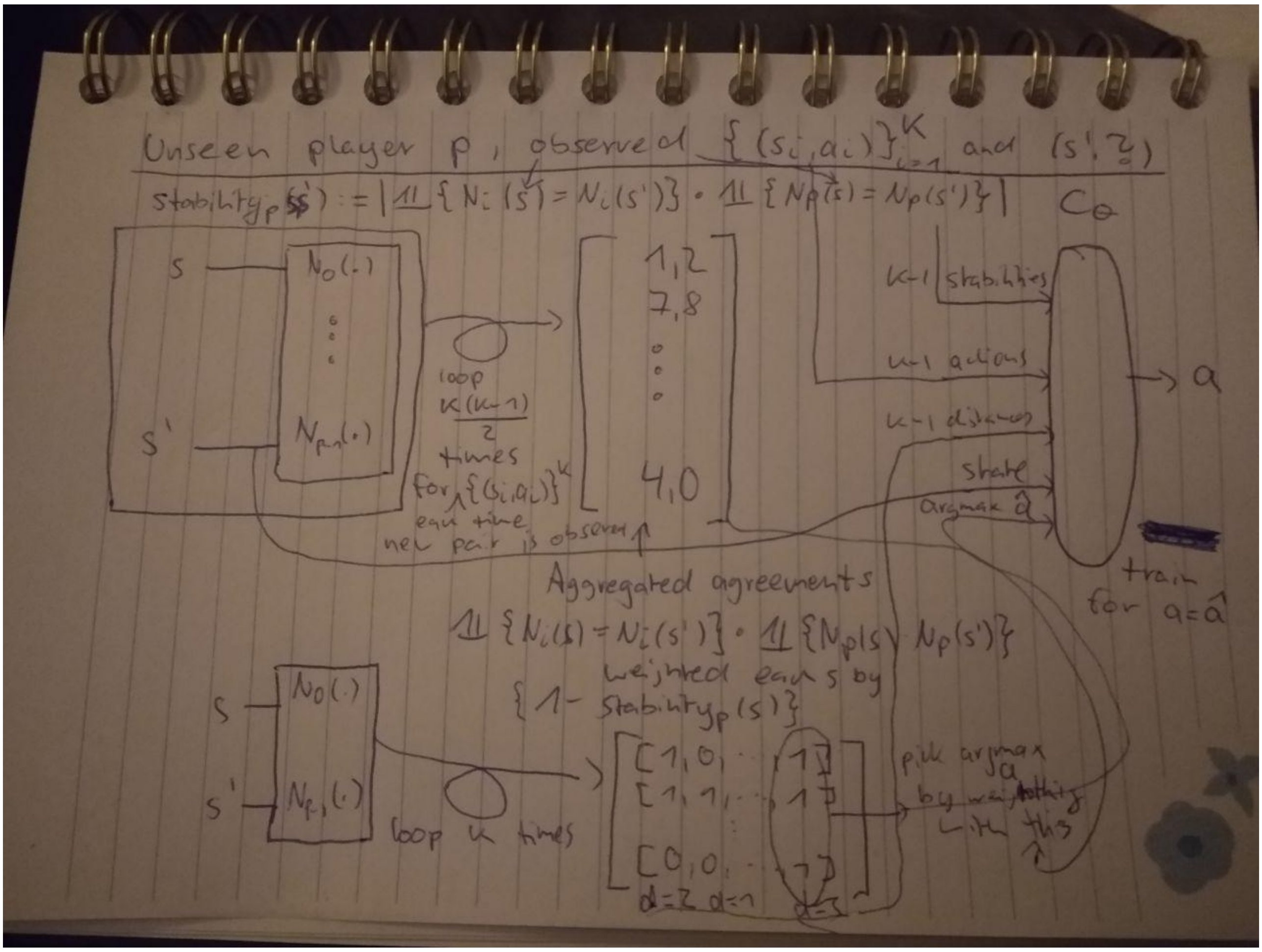
for a **single** Task A, and thus the meta-objective becomes

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

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:


Unseen player p, observed {(s_i, a_i)}^K_{i=1} and (s', ?)
stability_p(s) := | 1{N_i(s) = N_i(s')} · 1{N_p(s) = N_p(s')} |
C_θ
s
N_0(·)
s'
N_{p-1}(·)
loop K(K-1)/2 times
for {(s_i,a_i)}^K
each time new pair is observed
1,2
7,8
4,0
K-1 stabilities
K-1 actions
K-1 distances
share
argmax â
a
train for a=â
Aggregated agreements
1{N_i(s) = N_i(s')} · 1{N_p(s) N_p(s')}
weighted each s by
{1 - stability_p(s)}
N_0(·)
N_{p-1}(·)
loop K times
[1, 0, ... , 1]
[1, 1, ... , 1]
[0, 0, ... , 1]
d=2 d=1
pick argmax a
by weighting with this