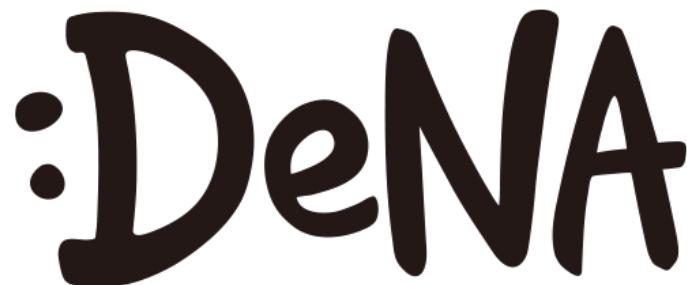


Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games



RL-Tokyo #68
Ikki Tanaka
@ikki407

本研究を選んだ動機

- Gaming Companyで働いており、Atariよりは高級なGameであるStarCraftに強化学習を適用している応用が興味深かったから
- DeepMindやMicrosoftがBlizzardと協力して、StarCraftをRLの題材にしていることは知っているが、具体的な研究や実験などを調査したことがなかったから
- PvPや複数人でプレイするような現代ゲームにRLを適用する際の問題設定や課題、アルゴリズムなどを知りたかったから

PvP: Player versus player

本研究の貢献

直訳注意!!

- スケーラブルかつ効果的なAgent間の通信プロトコルを維持するために、ベクトル拡張版Actor-Critic法を用いたBiCNet[’bIknet]を提案
 - 教師データ無しに、熟練プレイヤーが行うような様々な協調戦略をBiCNetは学習することができた
 - 提案したBiCNetは異種Agentが存在するタスクにも容易に適用可能なことを示した (e.g. 回復専門Agentと攻撃専門Agentによるパーティー)
 - 数値実験で様々なシナリオ・手法に対して評価を行い、SOTAな性能を示し、かつLarge-Scaleな現実世界への応用に対してのポテンシャルがあることも示した
1. ベクトル拡張版Actor-Critic法: 決定論の方策を用いた連続行動空間にも適用可能なAC法
 2. 異種Agent: Homogeneous agent

YouTube Demo

- <https://www.youtube.com/watch?v=kW2q15MNFug>



目次

- マルチエージェント強化学習とは
- Multiagent Bidirectionally-Coordinated Nets (BiCNet)
 - 大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化
 - 個別の報酬を用いたマルチエージェントActor-Critic
- 実験
 - 性能比較
 - 学習した協調戦略の分析&可視化
- まとめ

マルチエージェント強化学習とは

- 複数の自律的に行動するエージェントが存在する環境において、ある目標を達成するために、各エージェント間の協調的な行動戦略を学習すること
- 通常、各エージェントの行動は他のエージェントの行動選択方針にも相互に影響を与えるため、どのように行動するのが良いかという方策を求めるることは困難
 - RLにより獲得できると注目されている
- 2種類のエージェントが存在本研究では両方実験
 - Homogenousエージェント: 同じ行動空間を持つ
 - Heterogeneousエージェント: 異なる行動空間を持つ
- 応用例: スケジューリング、タスク割り当て、自律ロボット、RoboCup

マルチエージェント系としてのStarCraft

- リアルタイム戦略ゲーム
- 複数エージェント（ユニット）を制御して、敵軍隊を破壊するゲーム
 - 協調的な行動が鍵になる
 - E.g. Hit and Run, move w/o collision, focus fire w/o overkill
- 囲碁よりも可能状態数が多いためコンピュータにとって最も難しいゲームの一種として考えられている
 - エージェントが増えるほどパラメタ空間が指数的に増大
 - 既存の単純に学習器を結合する方法では動的な変化に対応できない
- 本研究ではゼロ和確率ゲームとして定式化し解いている

目次

- マルチエージェント強化学習とは
- **Multiagent Bidirectionally-Coordinated Nets (BiCNet)**
 - 大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化
 - 個別の報酬を用いたマルチエージェントActor-Critic
- 実験
 - 性能比較
 - 学習した協調戦略の分析&可視化
- まとめ

大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化

■ 競争的なゼロ和確率ゲーム

- 味方とは協力し
- 敵とは戦う

タプルで表現

$$(\mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \{\mathcal{B}_i\}_{i=1}^M, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^{N+M})$$

\mathcal{S} : 状態空間

\mathcal{A}_i : Agent i の行動空間

\mathcal{B}_j : Agent j の行動空間

$i \in [1, N]$: controlled agent i

$j \in [1, M]$: enemy agent j

$\mathcal{T} : \mathcal{S} \times \mathcal{A}^N \times \mathcal{B}^M \rightarrow \mathcal{S}$: 遷移確率

$\mathcal{R}_i : \mathcal{S} \times \mathcal{A}^N \times \mathcal{B}^M \rightarrow \mathbb{R}$: 報酬関数
 $i \in [1, N + M]$

■ 大域的報酬 (Global Reward) を定義

- 健康状態(Health level)の差分に基づく時変報酬

$$r(\mathbf{s}, \mathbf{a}, \mathbf{b}) \equiv \frac{1}{M} \sum_{j=N+1}^M \Delta \mathcal{R}_j^t(\mathbf{s}, \mathbf{a}, \mathbf{b}) - \frac{1}{N} \sum_{i=1}^N \Delta \mathcal{R}_i^t(\mathbf{s}, \mathbf{a}, \mathbf{b})$$

学習エージェント
から見た報酬

$\Delta \mathcal{R}_j^t(\cdot) \equiv \mathcal{R}_j^{t-1}(\mathbf{s}, \mathbf{a}, \mathbf{b}) - \mathcal{R}_j^t(\mathbf{s}, \mathbf{a}, \mathbf{b})$: 時間ステップ間の健康状態の差分

$\mathbf{a}_\theta : \mathcal{S} \rightarrow \mathcal{A}^N, \mathbf{b}_\phi : \mathcal{S} \rightarrow \mathcal{B}^M$: 決定論的方策

$\mathbf{a} \in \mathcal{A}^N, \mathbf{b} \in \mathcal{B}^M$: 時刻 t, 状態 s が与えられた時の行動

$\mathcal{A}_i = \mathcal{A}, \mathcal{B}_j = \mathcal{B}$: 一旦 Homogenous な行動空間を定義

大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化

■ 競争的なゼロ和確率ゲーム

- 味方とは協力し
- 敵とは戦う

タプルで表現

$$(\mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \{\mathcal{B}_i\}_{i=1}^M, \mathcal{T}, \{\mathcal{R}_i\}_{i=1}^{N+M})$$

■ 大域的報酬 (Global Reward) を定義

- 健康状態(Health level)の差分に基づく時変報酬

$$r(\mathbf{s}, \mathbf{a}, \mathbf{b}) \equiv \frac{1}{M} \sum_{j=N+1}^M \Delta \mathcal{R}_j^t(\mathbf{s}, \mathbf{a}, \mathbf{b}) - \frac{1}{N} \sum_{i=1}^N \Delta \mathcal{R}_i^t(\mathbf{s}, \mathbf{a}, \mathbf{b})$$

学習エージェント
から見た報酬

↑敵エージェントの
健康状態の減少量

↑味方エージェントの
健康状態の減少量

$$\Delta \mathcal{R}_j^t(\cdot) \equiv \mathcal{R}_j^{t-1}(\mathbf{s}, \mathbf{a}, \mathbf{b}) - \mathcal{R}_j^t(\mathbf{s}, \mathbf{a}, \mathbf{b})$$

$\mathbf{a}_\theta : \mathcal{S} \rightarrow \mathcal{A}^N$, $\mathbf{b}_\phi : \mathcal{S} \rightarrow \mathcal{B}^M$: 決定論的方策

$\mathbf{a} \in \mathcal{A}^N$, $\mathbf{b} \in \mathcal{B}^M$: 時刻t, 状態sが与えられた時の行動

$\mathcal{A}_i = \mathcal{A}$, $\mathcal{B}_j = \mathcal{B}$: 一旦Homogenousな行動空間を定義

\mathcal{S} : 状態空間

\mathcal{A}_i : Agent iの行動空間

\mathcal{B}_j : Agent jの行動空間

$i \in [1, N]$: controlled agent i

$j \in [1, M]$: enemy agent j

$\mathcal{T} : \mathcal{S} \times \mathcal{A}^N \times \mathcal{B}^M \rightarrow \mathcal{S}$: 遷移確率

$\mathcal{R}_i : \mathcal{S} \times \mathcal{A}^N \times \mathcal{B}^M \rightarrow \mathbb{R}$: 報酬関数
 $i \in [1, N + M]$

最適行動価値関数

- 学習エージェントの目的
 - 割引報酬和の期待値の**最大化** $\mathbb{E}[\sum_{k=0}^{+\infty} \lambda^k r_{t+k}]$, where $0 \leq \lambda < 1$
- 敵エージェントの目的
 - 上式の**最小化**
- 結果的にMinimaxゲームとなる
$$Q_{\text{SG}}^*(\mathbf{s}, \mathbf{a}, \mathbf{b}) = r(\mathbf{s}, \mathbf{a}, \mathbf{b}) + \lambda \max_{\theta} \min_{\phi} Q_{\text{SG}}^*(\mathbf{s}', \mathbf{a}_{\theta}(\mathbf{s}'), \mathbf{b}_{\phi}(\mathbf{s}'))$$
- $Q_{\text{SG}}^*(\mathbf{s}, \mathbf{a}, \mathbf{b})$ は最適行動価値関数
 - ベルマン最適方程式に従う

$\mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{b})$ により決定



一般的にMinimax Q学習はマルチエージェントRLでよく使われる

最適行動価値関数

■ 学習エージェントの目的

- 割引報酬和の期待値の**最大化** $\mathbb{E}[\sum_{k=0}^{+\infty} \lambda^k r_{t+k}]$, where $0 \leq \lambda < 1$

割引率

■ 敵エージェントの目的

- 上式の**最小化**

■ 結果的にMinimaxゲームとなる

$$Q_{\text{SG}}^*(\mathbf{s}, \mathbf{a}, \mathbf{b}) = r(\mathbf{s}, \mathbf{a}, \mathbf{b}) + \lambda \max_{\theta} \min_{\phi} Q_{\text{SG}}^*(\mathbf{s}', \mathbf{a}_\theta(\mathbf{s}'), \mathbf{b}_\phi(\mathbf{s}'))$$

■ $Q_{\text{SG}}^*(\mathbf{s}, \mathbf{a}, \mathbf{b})$ は最適行動価値関数

- ベルマン最適方程式に従う

$\mathcal{T}(\mathbf{s}, \mathbf{a}, \mathbf{b})$ により決定



一般的にMinimax Q学習はマルチエージェントRLでよく使われる

1. StarCraftではMinimax Q学習は計算コストが高い
2. Adversarial Q学習(上式)では敵エージェントのモデリング必要

仮想プレイとDNNによる敵の方策学習

- StarCraftでは計算コストが高い → **仮想プレイ (Fictitious Play)**
 - 相手がAIエージェントの時、自分/敵は相手がそれまでに取った行動の経験分布を元に最適戦略を取るプレイ方法
 - 反復的にQ関数を最適化する
- 敵エージェントのモデリング必要 → **DNNによる敵の方策学習**
 - SLの枠組みでDNNを用いて決定論の方策 \mathbf{b}_ϕ を学習
 - 敵の経験サンプル(s, a)を持ちいて方策ネットワークを訓練

$$\Delta \phi \propto \frac{\partial \mathbf{b}_\phi(s)}{\partial \phi}$$

- 学習した敵の方策を固定することで確率ゲームはシンプルなMDPに

$$Q^\theta(s, a) = r(s, a) + \lambda Q^\theta(s', a_\theta(s'))$$

ここまででは一般的な大域的報酬(GLOBAL REWARD)を用いたゼロ和確率ゲーム(MINIMAX Q学習)のお話

ここからは個別的報酬(INDIVIDUAL REWARD)のお話

個別的報酬を用いたマルチエージェントActor-Critic

■ 大域的報酬の欠点

- 集団的協調は通常ある種の内部構造を持つという事実を無視（局所的協調や局所的報酬関数）
- 各エージェントは協調を促進する固有の目的を持つ傾向

■ 個別的報酬と相互作用しているエージェントを用いて↑をモデル化

$$r_i(\mathbf{s}, \mathbf{a}, \mathbf{b}) \equiv \frac{1}{|j|} \sum_{j=N+1 \cap top-K(i)}^M \Delta \mathcal{R}_j(\mathbf{s}, \mathbf{a}, \mathbf{b}) - \frac{1}{|i'|} \sum_{i'=1 \cap top-K(i)}^N \Delta \mathcal{R}_{i'}(\mathbf{s}, \mathbf{a}, \mathbf{b})$$

$top-K(i)$: エージェント*i*と相互作用している他エージェントTopKのリスト

■ 結果として、N本のベルマン方程式を得る

方策関数とQ関数は同じ
パラメタ θ をもつ

$$Q_i^\theta(\mathbf{s}, \mathbf{a}) = r_i(\mathbf{s}, \mathbf{a}) + \lambda Q_i^\theta(\mathbf{s}', \mathbf{a}_\theta(\mathbf{s}'))$$

$$i \in \{1, \dots, N\}$$

Off-Policy Deterministic Actor-Critic Algorithm

■ 目的関数（学習エージェントにとっての）

$$J(\theta) = \mathbb{E}_{s \sim p_1} \left[\sum_{i=1}^N r_i(s, \mathbf{a}_\theta(s)) \right] = \int_{\mathcal{S}} p_1(s) \sum_{i=1}^N Q_i^\theta(s, \mathbf{a})|_{\mathbf{a}=\mathbf{a}_\theta(s)} ds.$$

■ Actor (決定論の方策勾配)

- 連続行動空間ではモデルフリーな方策反復法は困難(ステップごとにmax演算行うから)

$$\nabla_\theta J(\theta) = \frac{\partial}{\partial \theta} \int_{\mathcal{S}} p_1(s) \sum_{i=1}^N Q_i^{\mathbf{a}_\theta}(s, \mathbf{a}) ds = \mathbb{E}_{s \sim \rho_{\mathbf{a}_\theta}^{\mathcal{T}}(s)} \left[\sum_j \left(\sum_{i=1}^N \frac{\partial Q_i^{\mathbf{a}_\theta}(s, \mathbf{a})|_{\mathbf{a}=\mathbf{a}_\theta(s)}}{\partial \mathbf{a}_j} \right) \frac{\partial \mathbf{a}_{j,\theta}(s)}{\partial \theta} \right]$$

■ Critic ($Q^\xi(s, a)$, Off-policy探索)

$$\nabla_\xi L(\xi) = \mathbb{E}_{s \sim \rho_{\mathbf{a}_\theta}^{\mathcal{T}}(s)} \left[\sum_{i=1}^N (r_i(s, \mathbf{a}_\theta(s)) + \lambda Q^\xi(s', \mathbf{a}_\theta(s')) - Q^\xi(s, \mathbf{a}_\theta(s))) \frac{\partial Q_i^\xi(s, \mathbf{a}_\theta(s))}{\partial \xi} \right]$$

ActorとCriticのネットワークに対してSGD適用できる

くどくど言ってないでアルゴリズム見た方が早い

Algorithm 1 BiCNet algorithm

Initialise actor network $\mathbf{a}_\theta(\mathbf{s})$ and critic network $Q^\xi(\mathbf{s}, \mathbf{a})$ with parameters ξ and θ
Initialise target network $\mathbf{a}_{\theta'}(\mathbf{s})$ and critic network $Q^{\xi'}(\mathbf{s}, \mathbf{a})$ with $\xi' \leftarrow \xi$ and $\theta' \leftarrow \theta$
Initialise replay buffer R
for episodes=1, M **do**
 initialise a random process \mathcal{N} for action exploration
 receive initial observation state s_1
 for t=1, T **do**
 select and execute action $a_i^t = \mathbf{a}_\theta(s_t) + \mathcal{N}_t$ for each agent i
 receive reward r_i^t and observe new state s_i^{t+1}
 store transition $[s_i^t, a_i^t, r_i^t, s_i^{t+1}]_{i=1}^N$ in R
 sample a random minibatch of M transitions from R
 compute target value for each agent in each transition $(\hat{Q}_1^t, \hat{Q}_2^t, \dots, \hat{Q}_N^t)$ using the Bi-RNN:

$$\hat{Q}_i^t = r_i^t + \gamma Q^{\xi'}(s_i^{t+1}, \mathbf{a}_{\theta'}(s_i^{t+1}))$$

compute critic update according to Eq.(9) via BPTT:

$$\Delta\xi = \frac{1}{M} \sum_{m=1}^M \nabla_\xi L(\xi)$$

compute actor update according to Eq.(8) via BPTT:

$$\Delta\theta = \frac{1}{M} \sum_{m=1}^M \nabla_\theta J(\theta)$$

update the target network:

$$\xi' \leftarrow \gamma\xi + (1 - \gamma)\xi', \quad \theta' \leftarrow \gamma\theta + (1 - \gamma)\theta'$$

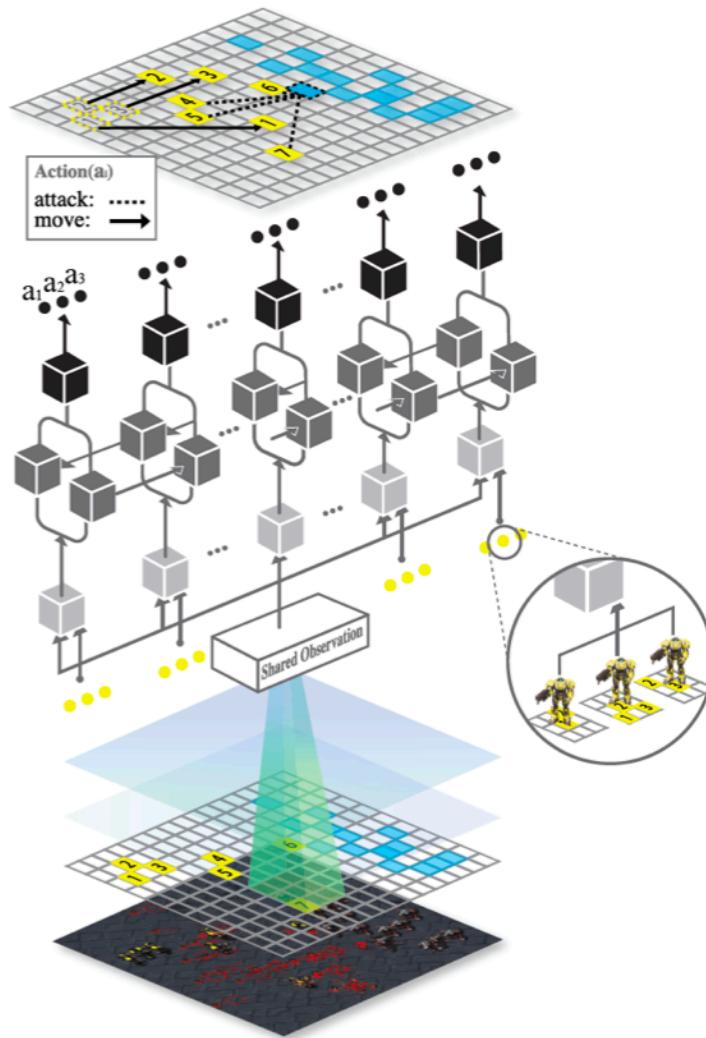
end for
end for

BiCNet

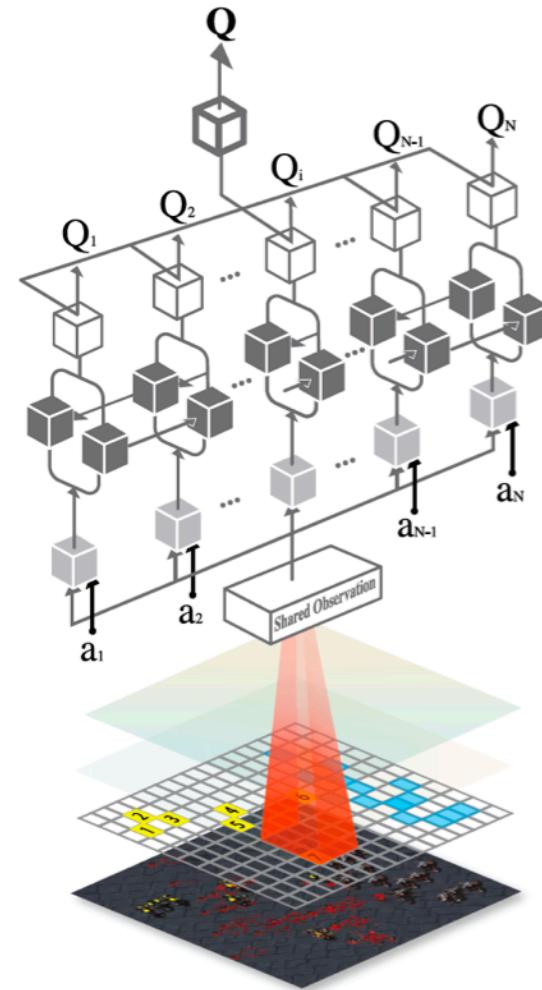
Bi-directional RNNは↓を参考

M. Schuster, et al., Bidirectional recurrent neural networks, 1997

Policy
Net
(Actor)



Q
Net
(Critic)



(a) Multiagent policy networks with grouping

(b) Multiagent Q networks with reward shaping

Figure 1: Bidirectionally-Coordinated Net (BiCNet).



Attention Neron



Bi-directional RNN



Policy Action



Value Function



Reward Shaping



Agent

BiCNetに対する補足

■ 既存研究

- エージェント間の依存性は各エージェントのネットワークの出力を他のエージェントのネットワークに入力することで考慮してきた
- →エージェントが増加すると崩壊する

■ 提案手法では双方向RNN（Actor-Critic）を用いることで、エージェントの依存性は隠れ層の中でモデル化される

- 全行動からの勾配がネットワーク内を伝播するので、エージェント間の依存性をこのネットワークだけで考慮できる

■ 役割を維持するためにRNNへの入力順は固定

- 複数の最適行動間に存在する任意のありうる組合せに対処するため

■ モデルをScaleさせるため・集団活動を学ばせるためにエージェントをGroupingした

目次

- マルチエージェント強化学習とは
- Multiagent Bidirectionally-Coordinated Nets (BiCNet)
 - 大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化
 - 個別の報酬を用いたマルチエージェントActor-Critic
- 実験
 - 性能比較
 - 学習した協調戦略の分析&可視化
- まとめ

入出力・行動

■ 入力

- $72 \times 72 \times 16$
- 72×72 : ゲーム画像入力サイズ
- 16: ピクセルのHP, DMG, Map情報など
- Criticだけ: Actorが出力した行動

■ 出力

- Actor: 行動 for each agent
- Critic: Q-value for each agent

■ 行動

- 3次元ベクトル
- [ATK or MOVE[-1,1], Degree[rad?], Distance[px?]]

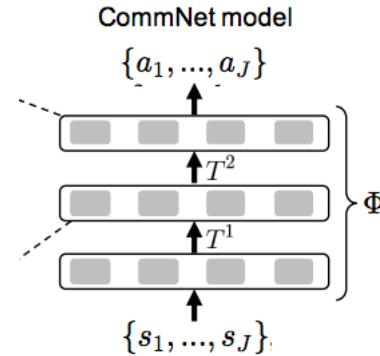


0以上なら攻撃

比較手法

■ Rule-Based AI

1. Built-in AI
2. Weakest AI: 最も弱い敵を攻撃
3. Closest AI: 最も近い敵を攻撃



CommNet: 構造が対称的で
HeterogeneousなAgentに対応不可
スケーラブルか不透明

■ RL-Based AI

1. Independent controller(IND): エージェント毎にNN作成
2. Fully-connected(FC): 全エージェント情報を入力に対してNN作成. 戦闘が変化すると異なるモデル構築と再訓練必要
3. CommNet: 段階的に全エージェントの平均情報を伝播するNN
4. GreedyMDP with Episodic Zero-Order Optimisation(GMEZO): エピソード毎に方策・Q関数を更新. パラメタ空間にNoise乗せ探索.

* [CommNet] S Sukhbaatar, et al., Learning Multiagent Communication with Backpropagation, 2016

* [GMEZO] N Usunier, Episodic Exploration for Deep Deterministic Policies: An Application to StarCraft Micromanagement Tasks, 2016

性能比較（勝率）

Table 1: Performance comparison. M: *Marine*, Z: *Zergling*, W: *Wraith*.

Combat	Rule Based			RL Based				
	Built-in	Weakest	Closest	IND	FC	GMEZO	CommNet	BiCNet
20 M vs. 30 Z	1.00	.000	.870	.940	.001	.882	1.00	1.00
5 M vs. 5 M	.720	.900	.700	.311	.076	.909	.950	.920
15 M vs. 16 M	.610	.000	.670	.590	.438	.627	.682	.712
10 M vs. 13 Z	.550	.230	.410	.522	.430	.570	.440	.640
15 W vs. 17 W	.440	.000	.300	.310	.460	.420	.470	.531

- 4/5でSOTA
- 5M vs, 5Mでは“focus fire”（集中攻撃）が鍵になる
 - BiCNet2番目の理由
 - BiCNetはより多くのエージェントの場合に効果を発揮する
 - しかしスキル獲得速度は速い(85%までComm50, BiC10バトル)

学習結果の可視化

■ 最終隠れ層をt-SNEで可視化

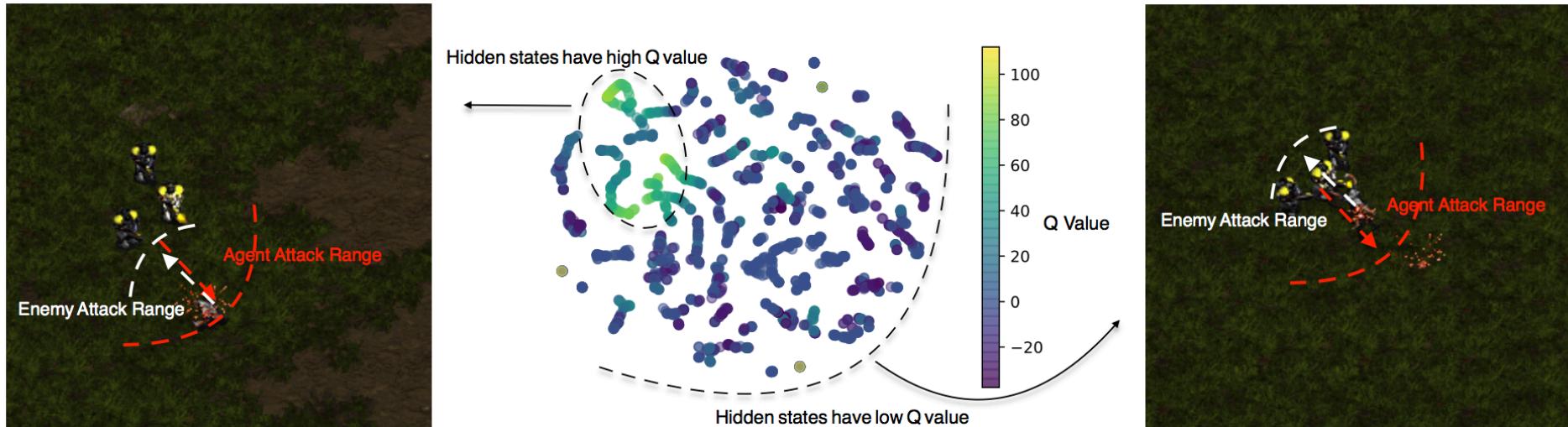


Figure 5: Visualisation for 3 Marines vs. 1 Super Zergling combat. **Left:** a case with high Q -value; **Middle:** Visualisation of hidden layer outputs using t-SNE, coloured in terms of their Q -values; **Right:** a case with low Q value.

- High Q value → 有利な状況
- Low Q value → 不利な状況

BiCNetにより学習された協調戦略

■ Coordinated moves without collision

- 始めはぶつかったり、お互い邪魔してた
- 学習が進むにつれて、協調的に行動



(a) Early stage of training (b) Early stage of training

(c) Well-trained

(d) Well-trained

Figure 6: Coordinated moves without collision in combat 3 *Marines* (*ours*) vs. 1 *Super Zergling* (*enemy*). The first two (a) and (b) illustrate that the collision happens when the agents are close by during the early stage of the training; the last two (c) and (d) illustrate coordinated moves over the well-trained agents.

BiCNetにより学習された協調戦略

■ Hit and Run tactics

- Loop: ATK → ATKされる → 逃げる → 安全 → 再び攻撃

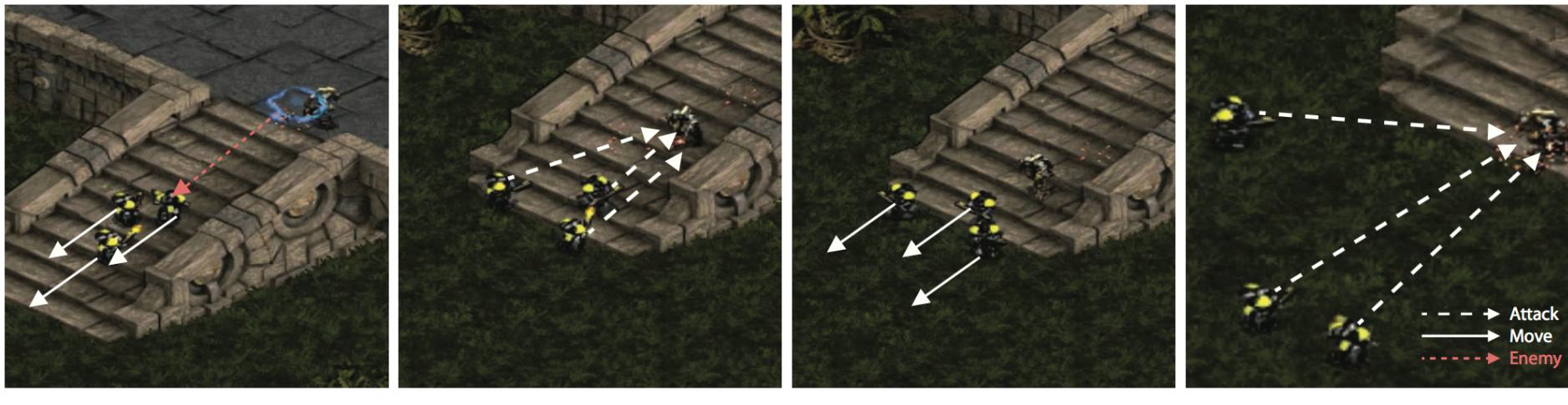


Figure 7: *Hit and Run* tactics in combat 3 *Marines* (ours) vs. 1 *Zealot* (enemy).

BiCNetにより学習された協調戦略

■ Coordinated cover attack

- 蒙きつけ役エージェントが敵をおびき寄せる
- おびき寄せた敵を他のエージェントが攻撃
- Hit and Runの上位互換（実際のバトルでも使われる）

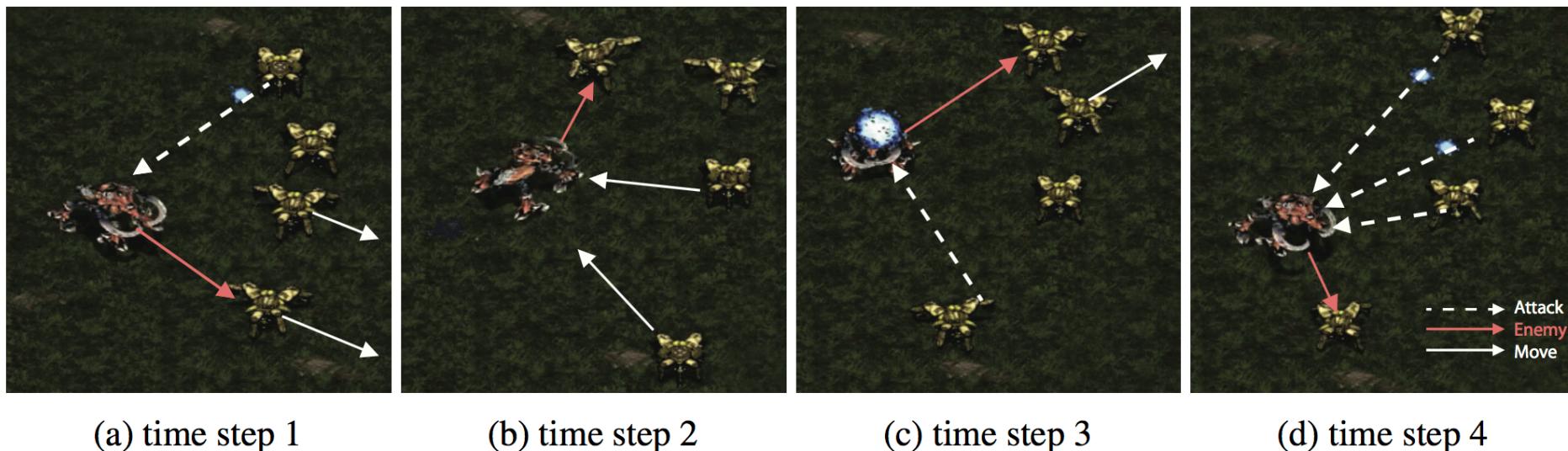


Figure 8: Coordinated cover attack in combat 4 Dragoons (ours) vs. 2 Ultralisks (enemy).

敵を強くした時だけ獲得した

BiCNetにより学習された協調戦略

■ Focus fire without overkill

- エージェントを分散させたり、1体の敵だけ攻撃する（Overkill）のは戦略的に良くはない
- 提案手法ではグループ間・グループ内の振る舞いを学習
- グループ毎に異なる敵を効率的に攻撃するように

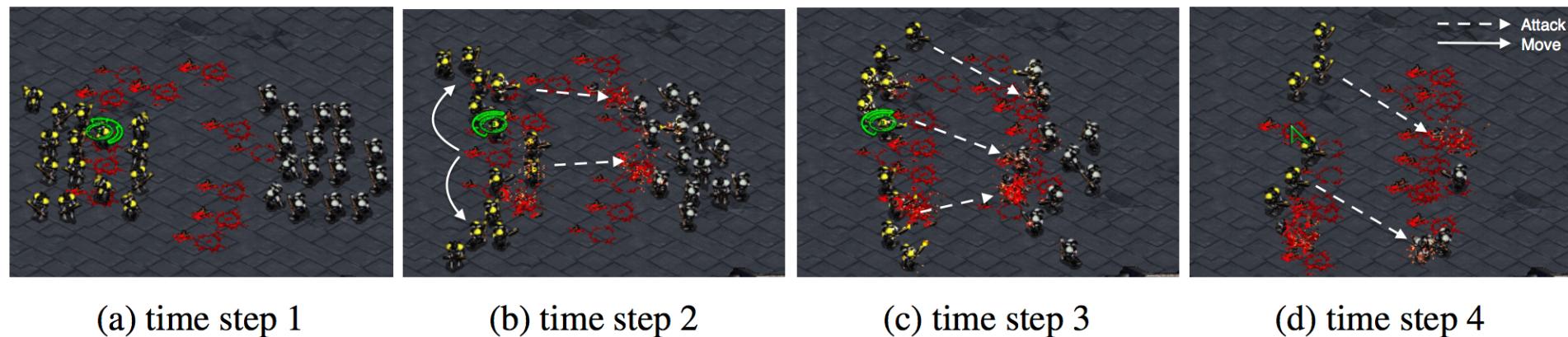
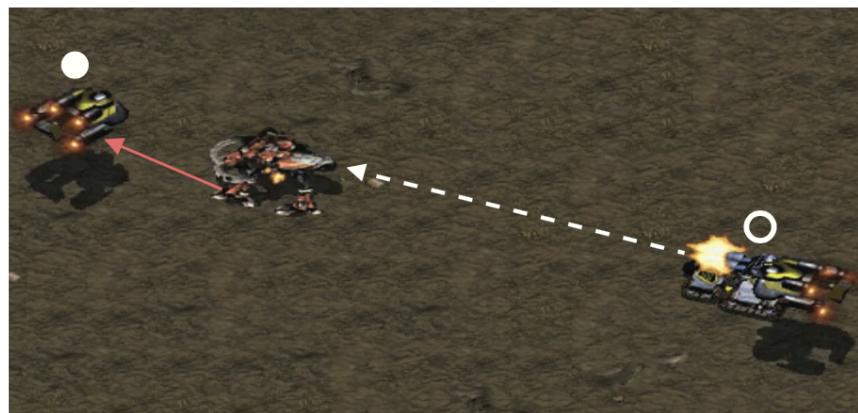


Figure 10: "focus fire" in combat 15 Marines (ours) vs. 16 Marines (enemy).

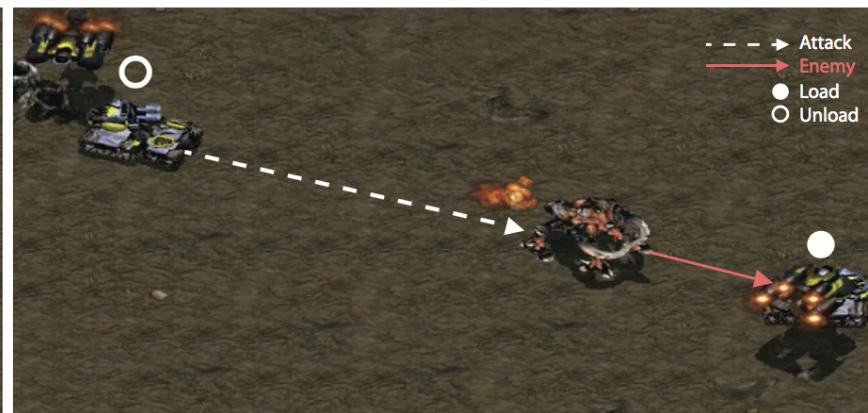
BiCNetにより学習された協調戦略

■ Collaborations between heterogeneous agents

- タイプが異なるエージェントの学習も試みた
- 2 Dropship and 2 Tank vs. Ultralisk
- Dropship: 攻撃できない。地上のUnitを空中に避難できる
- Tank: 攻撃できる地上Unit



(a) time step 1



(b) time step 2

Figure 11: Coordinated heterogeneous agents in combat 2 *Dropships* and 2 *tanks* vs. 1 *Ultralisk*.

同タイプUnitだけでパラメタ共有
することでうまく学習

目次

- マルチエージェント強化学習とは
- Multiagent Bidirectionally-Coordinated Nets (BiCNet)
 - 大域的報酬を用いたゼロ和確率ゲームとしての戦闘定式化
 - 個別の報酬を用いたマルチエージェントActor-Critic
- 実験
 - 性能比較
 - 学習した協調戦略の分析&可視化
- まとめ

まとめ

- BiCNetを用いて新たなマルチエージェント深層強化学習の枠組みを提案した
- ベクトル拡張版Actor-Critic法を構築しBiCNetと用いることで、エージェント間の協調性を学習した
- BiCNetを用いることでEnd-to-Endで効率よく協調行動を学習できることを示した
- StarCraftを用いた数値実験により、様々な戦闘で協力スキルを学習できることを示した

Appendix