Algorithm of Howls

・エージェントのソースコード及びドキュメントの公開場所

https://github.com/yutian-zhao/aiwolf_java
https://github.com/yutian-zhao/aiwolf_java/tree/policy_only
https://github.com/yutian-zhao/aiwolf_python

・エージェントのコンセプト

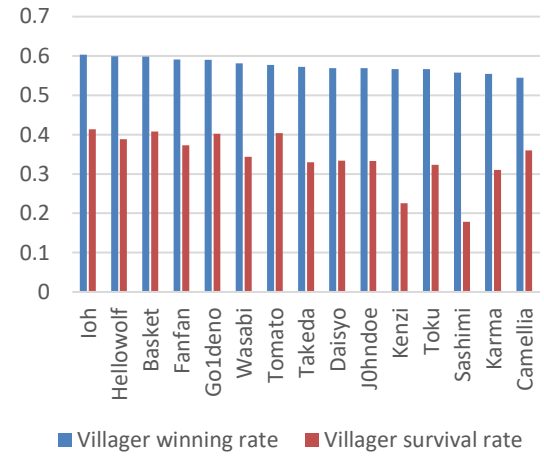
The idea of our agent is to separate the function of AIWolf into two parts: 1. Strategy; 2. Character prediction.
In order to create satisfactory agent with good performance, for strategy part, we refer to the previous winning agents and adopt multiple strategy for each character.
We modified strategy of SEER, MEDIUM, POSSESSED, VILLAGER, and BODYGUARD.
In order to have satisfactory result, we conducted multiple test running (100 * 100 sets) with past agents from 1st to 4th AIWolf International Competition.
After comparing the performance of 15 different agents, we accept various strategy for each character.
On the other hand, we find agent policies heavily depend on the role prediction based on the game status, thus the accuracy of the prediction significantly affects the effectiveness of the policy. While most existing agents rely on the Bayesian inference from previous game logs, we switch to using neural networks to perform prediction. We expect that the neural network, which is trained from large amounts of data, can generalize to unseen agents, and recognize action patterns to predict roles.

・エージェント全体の技術的特徴の解説

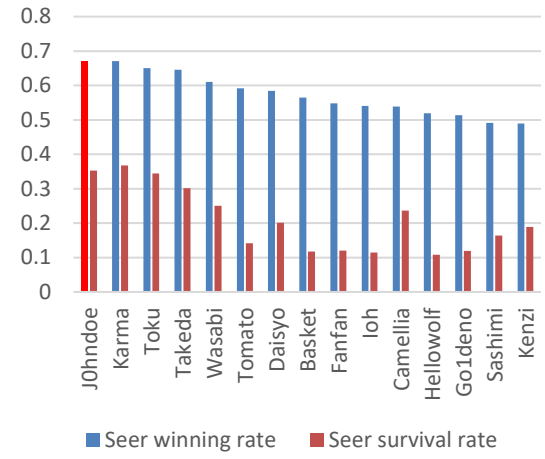
As mentioned in the concept part, we adopt multiple strategy for each character. To be more specific, we modified the Karma strategy for SEER, Fanfan strategy for BODYGUARD, Tomato strategy for MEDIUM.
For the VILLAGER, we mainly add new strategy of pretending to be SEER in 15-agent games, the idea is to protect the real SEER in the first or second round.
For the WEREWOLF, we mainly simplify the talk strategy, abandoning the possibility talk selection.
For the POSSESSED, we mainly modify the strategy while pretending to be SEER, but not CO at first round.
Overall, the positive strategy of most Takeda-based teams (e.g. CO at first round) is drastically modified of CO when WEREWOLF is detected or successfully protected, or just keep silent at the 1st round.
This negative strategy sets are expected to have better performance in 15-agent games as it adds up to the alive possibility for most characters.
Apart from that, for character prediction, LSTM is applied, trained from test running with previous winning agents from 1st to 4th International Competition.
Having mentioned in the concept section, our agent improves both the policy and the internal prediction model. To be more specific, we found that while Basket had developed a complicated action mechanism from previous agents, most decisions are made based on parameters given by humans which are not necessarily optimal. We analyzed other previous agents with good results and integrate their policies into ours. For the prediction model, we replace the original Bayesian inference

from game logs with a neural network model, which has a novel architecture that composes of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN). The model is trained on a large amount of data from previous competitions and can be used on any werewolf games with 15 or fewer plays and 6 or fewer roles. The details of the model design are described in the following section.

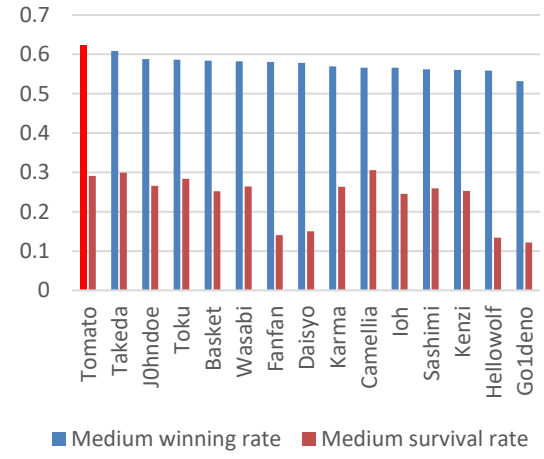
・特徴についての解説



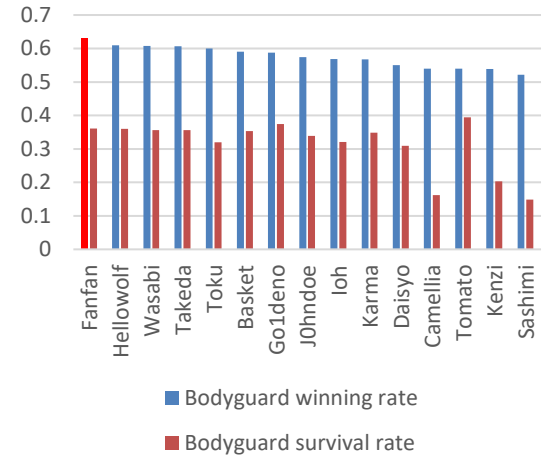
(a) Villager



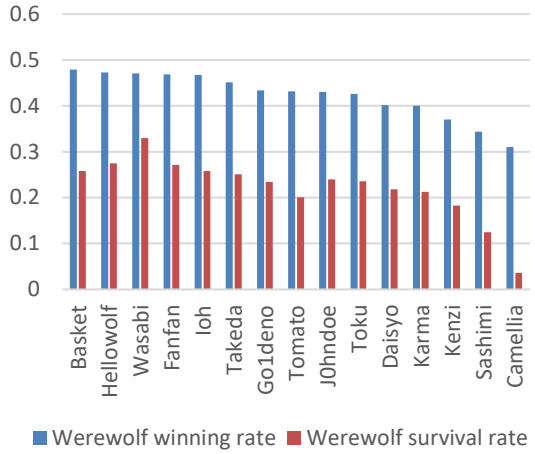
(b) Seer



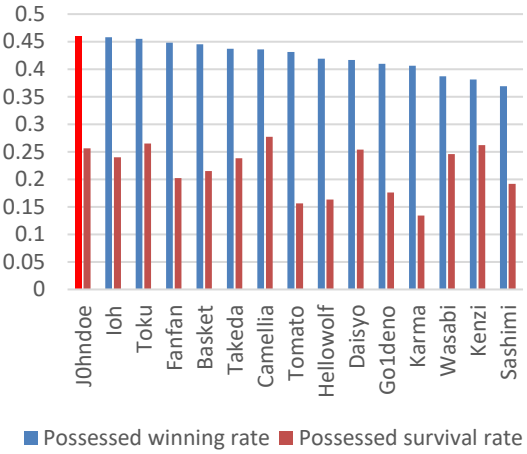
(c) Medium



(d) Bodyguard



(e) Werewolf



(f) Possessed

Fig.1 Wining rate & Survival rate of each character (100 * 100 sets)

As shown in Fig.1, the result of winning rate and survival rate have shown previous agents having different advantages for each character.

After analyzing these agents. we modified the Karma strategy for SEER, Fanfan strategy for BODYGUARD, Tomato strategy for MEDIUM. For the VILLAGER, we mainly add new strategy of pretending to be SEER in 15-agent games, the idea is to protect the real SEER in the first or second round. For the WEREWOLF, we mainly simplify the talk strategy, abandoning the possibility talk selection. For the POSSESSED, we mainly modify the strategy while pretending to be SEER, but not CO at first round.

More specific modification can be seen from the following figures.

Medium	Basket	Tomato
CO	when black is detected	at first round
Judgement	only the executed	executed + seer
Talk	random talk strategy	must talk

Bodyguard	Basket	Fanfan
CO	when success guarded	no CO
Guard strategy	村人らしさ + 3 * 占い師らしさ + 霊媒師らしさ　に勝率を補正して最も高いプレイヤーを選択	1. seer 2. medium 3. villager
Talk	random talk strategy talk about success guarded agent	vote to wolf

Seer	Basket	Karma

CO	CO at first round CO as werewolf when possessed alive (agents <= 3)	CO at first round CO as werewolf when possessed alive (agents <= 3)
Divine	Divine from 1 to the end	Divine from 1 to the end
Talk	Talk about the divined result when werewolf detected. Probability voting strategy	Talk about the divined result when werewolf detected. Vote to most likely to be werewolf.

Overall, the positive strategy of most Takeda-based teams (e.g. CO at first round) is drastically modified of CO when WEREWOLF is detected or successfully protected, or just keep silent at the 1st round.

There are previous works on using neural networks for role prediction. Among them, [2] provides a useful analysis of which information is effective in werewolf prediction, and [3] includes more complex information considering the relationships between other agents and the agent itself. It extends the model to predict all roles in the 15-person game using fully connected layers. Since role prediction happens every day except the first day, it's natural to use a recurrent architecture to take information from previous days into account. [4][5][6] use Long Short-Term Memory (LSTM), a kind of RNN. While [4][5] use ids to represent talk contents, [6] use Word2Vec to encode the talk contents. All the work mentioned only uses fully connected linear layers and tests with the same agents for training. In this work, we invent a new architecture, which not only considers sequential inputs but also pair-wise information. We test with unseen agent combinations and compare the model with the original Bayesian networks.

One important problem is how to represent game information efficiently. According to [2][3], besides facts like coming out, being alive or not, and divine results, there are actions that convey mutual relationships, like vote and agreement. This motivates us to encode the information in a 2d matrix so that mutual information between agent x and agent y can be stored in the entry in row x and column y. For simple facts, we broadcast to all columns. For example, if agent x is alive, we fill 1 in all entries in row x. We categorized the game information as 8 types as shown in Table 1.

The CNN architecture is suitable for matrix representations, which can be treated as multi-channel images. We use CNN to encode the game status and use LSTM to process the sequential information and finally use a multi-layer perceptron (MLP) to output probabilities of each role for each agent. The details are described in Appendix 1. The design of the state representation and the model architecture allows for handling any game type with 15 or fewer players and 6 or fewer roles. We can simply fill ignored index in the unused entries.

AIWolf competitions having been held for a few years, an abundant amount of game logs has been cumulated. To make our model generalizable to unseen agents, unlike most existing works which only focus on game logs from one competition, we train it on the dataset as large as possible. The data is from GAT2017, CEDEC2017, GAT2018, CEDEC2018, 1st, 2nd, and 4th international AIWolf competitions, which are public on the official website [7].

We use multi-class cross entropy loss and AdamW optimizer. Besides role prediction, we also perform vote prediction as an auxiliary task. Early stopping and validation are used to avoid overfitting. Note that the bodyguard is ignored because we found it hard to identify.

Evaluation

We test the performance of the model in the game held among 15 different agents chosen from previous competitions and this combination is never seen during training. The prediction accuracy is depicted in Table 2 for the test set, which is the same as the training set, and in Table 3 for the new testing game environments. We also compare the accuracy with the original Bayesian model as shown in Figure 1. Note here we evaluate the model in a real game environment and only choose the agent with the highest possibility. Unfortunately, we found the model is not as competitive as the original Bayesian inference. This might be because the model can't generalize to new environments as well as Bayesian inference, which can be updated after each game.

・エージェントの今後の課題

Unfortunately, because of the time limitation, the test of newly developed Howls02 agent against past agents from 1st to 4th competition is not finished.
For the future work, we are planning to conduct multiple test sets (100 * 100 sets) to clarify the performance of each character.
Currently, we found our strategy of WEREWOLF needs to be improved as it failed in many 5-agent games competing with Team Basket. Possibly it was because of the simplified talk strategy which we tend to always vote to the most likely to be werewolf agents. This should be verified and modified in the near future.

・**Reference**

[1] http://aiwolf.org/archives/2840
[2] http://aiwolf.org/

Appendix

Appendix 1. Model Architectural

CNN (size 15 × 15):
Conv2d(in=8, out=32, kernel=3, stride=2, pad=0), BatchNorm2d(32), ReLU(),
Conv2d(in=32, out=64, kernel=3, stride=1, pad=0), BatchNorm2d(64), ReLU(),
Linear(in=5*5*64, out=800).

RNN:
LSTM(in=800, out=800).

MLP (role prediction):
Linear(in=800, out=800), ReLU(),
Linear(in=800, out=6*15).

MLP (vote prediction):
Linear(in=800, out=100), ReLU(),
Linear(in=100, out=15), Sigmoid().

Table 1. State Representation

Channel	Range	Description
Known	{0,…,6}	Known roles. {1,…,6} represents 6 possible roles. Each agent only know its own role, except werewolves know its partners.
Alive	{0, 1}	Alive is 1 and Dead is 0.
Vote	{0, 1}	If agent x votes for agent b, then 1. Otherwise 0.
Skill	{-1, 0, 1}	Divine or identify results for Seer and Medium. Human is -1, Werewolf is 1, and Any is 0.
Coming out	{0,…,6}	{1,…,6} if agent x comes out as a specific role. 0 otherwise.
Estimate	{0,…,6}	{1,…,6} if agent x estimate agent y as a specific role. 0 otherwise.
Attitude	{-1, 0, 1}	A negative attitude is -1 while a positive attitude is 1. Otherwise 0.
Identified	{-1, 0, 1}	Agent x says it divined or identified agent y as Human (1) or Werewolf (-1). Otherwise 0.

Table 2. Prediction Accuracy for each role on each day on the test set

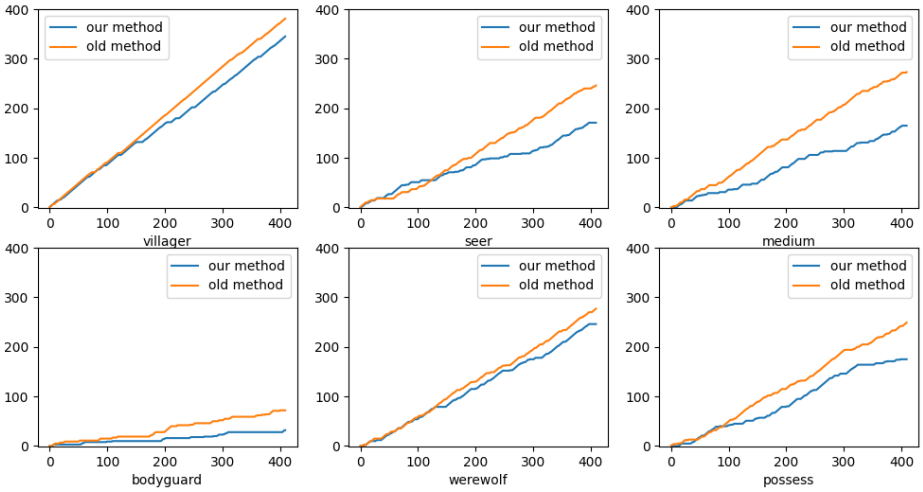
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
VILLAGER	45.8	60.7	68.8	72.7	75.1	76.7	77.9	78.3	78.2	77.9	77.5	77	76.5	76
SEER	71.2	81.1	83.7	84.7	85.1	85.2	85.2	85.4	85.3	85.2	85.2	85.1	85.1	85
MEDIUM	60	77.1	83.3	85.6	86.6	87	87.3	87.6	87.8	87.8	87.9	87.9	88	88
BODYGUARD	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WEREWOLF	92.8	91.9	91.8	92.6	93.5	94.1	94.3	94.3	94.4	94.4	94.5	94.5	94.6	94.6
POSSESSED	55.8	61.9	64.9	66.8	68	68.8	69.4	69.8	70.1	70.3	70.4	70.4	70.5	70.5

Bodyguard is not predicted. Although a 15-person game can last 14 days at most, most games end in less than 10 days.

Table 3. Prediction Accuracy for each role on each day in testing games

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
VILLAGER	44	56	60.7	61.9	63.2	62.7	64.9	66.4	67.3	67.4	67.7	67.6	67.6	67.7
SEER	32.3	41.9	35.4	39.1	38.4	39.1	41.3	41.3	41.3	41.3	41.3	41.3	42.1	42.1
MEDIUM	45.9	51	51	52.6	58.9	57.6	60.5	64.1	64.8	64.8	64.8	64.1	64.1	64.1
BODYGUARD	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WEREWOLF	85.3	81.1	81.7	82.6	85.3	84.1	85.8	85.3	85.5	85.5	85.1	84.8	85.1	84.9
POSSESSED	25.6	37.4	42.9	45.1	49	46	45.2	47.2	47.2	46.4	47.2	47.2	47.2	48

Figure 1. Accumulated number of correct predictions for days in 100 games



Test agents taken from previous competition

第4回人狼知能国際大会

No.	Name	Source code	Language
1	Basket	○	java
2	ioh	○	java
3	sUper_lL	×	
4	kgu_ryu	×	
5	takoyaki	×	
6	Hachi2	×	
7	KP22	×	
8	ice	×	
9	Ncu702	×	
10	tonkatsu	×	
11	CanisLupus	×	
12	mikami	×	
13	ichida	×	
14	daphne	×	
15	Baguette	△	java

第3回人狼知能国際大会

No.	Name	Source code	Language
1	toku/ICE	○	java
2	TOT	○	C#
3	KP22	×	
4	Syu	×	
5	CanisLupus	×	
6	Tomatoken	×	
7	SORA	×	
8	Hideto	×	
9	HALU	○	python
10	Tomato	○	java
11	OKAMI	○	python
12	karma	○	java
13	wasabi	○	java
14	Sashimi	○	java

第2回人狼知能国際大会

No.	Name	Source code	Language
1	takeda	○	java
2	otsuki	○	java
3	HALU	○	python
4	J0hnDoe	○	java
5	cube	○	java
6	daisyo	○	java
7	Tomo	○	java
8	simipu	○	java
9	Udon	○	C#
10	Tomato	○	java
11	wasabi	○	java
12	FoxuFoxu	○	python
13	PaSeRi	○	java
14	Camellia	○	java
15	Sashimi	○	java

第1回人狼知能国際大会

No.	Name	Source code	Language
1	takeda	○	java
2	hello_wolf	○	java
3	Udon	○	python
4	GO1DeNO	○	java
5	fisherman	○	java
6	fanfan	○	java
7	Tomato	○	java
8	calups	○	python
9	wasabi	○	java
10	kenzi	○	java
11	sonoda	○	python
12	cantar	○	python
13	LItt1eGirl	○	python
14	takaeye	×	
15	yskn67	○	python