Документация для хакатона по Дураку

Ты оказался в студенческом нейро-хакатоне, где тебе и твоей команде предстоит обучить нейросеть играть в карточную игру "Дурак" – и побороться за **призовой фонд в 50 000 рублей**.

Контекст

Ваша задача – обучить нейросеть побеждать предоставленную baseline-систему в 60 из 100 игр (60% винрейт в игре 1 на 1). В вашем распоряжении – **датасет из 6331 реальной игры между людьми (~300к снимков состояния)**, на базе которых вы обучаете свою модель играть 1 на 1.

В финал проходят команды, которые смогли достичь **винрейта в 60%.** Далее – все модели в финале сыграют 1000 игр против baseline-системы, – модель с самым высоким винрейтом (больше 60%) **побеждает.**

Призовой фонд составляет эквивалент **50000 рублей** в USDT, которые мы переведем на ваш TRC20 кошелек в день подведения итогов (19го апреля).

Baseline-система будет доступна 12го апреля, также, вместе с ней будет предоставлена клиентская обертка, через которую вы сможете подключаться к системе.

Даты

10.04 – Открытие хакатона
12.04 – Открытие доступа к baseline-системе
13.04 – Презентация промежуточных результатов. Митинг с модераторами
16.04 – Предварительный смотр решений
17.04 – Стоп кодинг, сдача моделей. Исходный код выгружаем на GitHub (либо передаем блокноту в Google Colab). Модели выгружаем на HuggingFace. (Обязательно предоставить код с тестированием главной модели: Подгрузка модели с HuggingFace и ее выполнение в сервисе тестирования)
17.04 - 18.04 – Оценка моделей
19.04 – Объявление победителя

Контекст/Правила/Настройки игры

Это карточная игра в Дурака. Игрока всегда 2 (игра 1 на 1), карты всегда 24. Есть как переводной так и классический (определяется параметром game_rules / game_type: 0 – классика, 1 – переводной). Игра заканчивается, когда у одного игрока заканчиваются все карты (при том, что в колоде, карт не осталось).

Структкура снимков игры

```
{
    "game_id": str,
    "trump": str,
    "timestamp": int,
    "winner": str,
    "game_rules": {
      "game_type": int,
      "player_amount": int,
      "card_amount": int
    },
    "deck": list,
    "bat": list,
    "table": list,
    "players": [{"id": str, "state": str, "hand": list},
                {"id": str", "state": str, "hand": list}]]
}
```

- game_id: уникальный идентификатор каждой игры. У разных снимков одной и той же игры одинаковый game_id, но разный timestamp
- trump: козырная карта этой игры (та, которая лежит под колодой, она всегда последняя в колоде и по ней определяется козырь)
- timestamp: порядковый номер этого снимка в конкретной игре. От 0
- winner: победитель в этой игре
- game_rules: словарь из настроек игры
- game_type: 0 – классика, 1 – переводной
- player_amount: количество игроков (всегда 2)
- card_amount: количество карт (всегда 24)
- deck: список из карт в колоде. Где первая карта – это последняя карта, и раздаются они с конца
- bat: список карт в бито. Где первая карта – это первая карта попавшая в бито, и попадают они туда с конца
- table: массив из словарей {“attack_card”: {“card”: “12H”, “user_id”: “123123”}, “defend_card”: {“card”: “13H”, “user_id”: “321321”}. В таком формате. Где attack_card это карта, которой атаковали, а defend_card – карта, которой защитились. Также user_id того, чья это карта. Если карта не отбита, то в словаре будет только attack_card.
- players: массив игроков. Каждый игрок – словарь
- id: айди игрока
- state: его стейт. Может быть attack, defend, bat, pass, take, winner, durak
- hand: список карт в руке игрока

Пример снимка в середине игры:


```
{"game_id": "8219abbc-7ec9-48f5-b9be-63374c227d6d",
    "trump": "14S",
    "timestamp": 6,
    "winner": "7279454545",
    "game_rules": {
        "game_type": 1,
        "player_amount": 2,
        "card_amount": 24},
    "deck": ["14S", "13D", "9C", "14H", "11S", "11C", "10C", "12H", "9D",
"11H"],
    "bat": ["10D", "11D"],
    "table": [
        {
            "attack_card": {"card": "10H", "user_id": "314737888"},
            "defend_card": {"card": "9S", "user_id": "7279454545"}
        }
    ],
    "players": [
        {
            "id": "7279454545", "state": "defend", "hand": ["12D", "13C",
"14C", "9H", "13S"]
        },
        {
            "id": "314737888", "state": "attack", "hand": ["14D", "12C", "13H",
"10S", "12S"]
        }
    ]
    }
```


Формат данных

Номиналы карт: [9,10,11,12,13,14] – Где 11 – валет, 12 – дама, 13 – король, 14 – туз.
Масти: [S, C, D, H] – Spades, Clubs, Diamonds, Hearts
В колоде, руках, бито карты всегда отображаются строкой номинал+масть: 11H, 12C, 14S, 9D.

Стейты:

- attack – в данный момент этот игрок атакует
- defend – в данный момент этот игрок защищается (на него нападают)
- bat – игрок прожал бито (т.к. Игра на двоих, после этого, стол очищается, карты со стола улетают в бито, новые карты раздаются из колоды, если они есть)
- pass – игрок прожал пас, когда у другого игрока прожато take. Т.е. Пас – это подтверждение, что мы не хотим ничего докинуть, после того, как оппонент решил взять карты со стола. (после этого стол очищается, карты улетают оппоненту в руку, мы добираем карт до 6)

- take – игрок прожал взять, ожидается пас от оппонента, после этого, наша рука пополнится картами со стола
- winner – стейт победителя, этот стейт появляется только в самом последнем снимке игры
- durak – стейт дурака, этот стейт появляется только в самом последнем снимке игры

Структура ответов серверу

Есть 4 основные структуры пакетов:

State: это структура которая говорят серверу, что вы хотите поменять свой стейт: нажать bat, pass или take
{“type”: “state”, “state”: “pass”}
{“type”: “state”, “state”: “bat”}
{“type”: “state”, “state”: “take”}

Attack: эта структура говорит серверу что вы хотите либо атаковать. Либо **ПЕРЕВЕСТИ,** в случае, если ваш state defend, на столе только карты одного номинала, а вы кидаете attack картой того же номинала, то срабатывает перевод. И ваш стейт меняется на attack, и все что было на столе + ваша карта – становится новой атакой
{“type”: “attack”, “move”: “10H”}

Defend: эта структура говорит серверу, что вы хотите защитить определенную карту определенной картой
{“type”: “defend”, “move”: [[“123123”, “10H”], [“321321”, “11H”]]}

Wait: эта структура говорит серверу, что ход от вас в данный момент не требуется, вы можете подождать. Т.к. Вам приходит пакет снимка на каждое изменение (включая изменение сделанное вами), то вам иногда не нужно делать ход, а можно просто отправить этот пакет
{“type”: “wait”}