CAN THE BEST JASS AIS BEAT THE TOP HUMANS?

ROMAN MARTINEZ

Bachelor Thesis

Research Center for Digital Sustainability
Faculty of Science
University of Bern

December 2022

SUPERVISORS:
PD Dr. Matthias Stürmer,
Joel Niklaus (Assistant)

ABSTRACT

The performance of Artificial Intelligence (AI) in imperfect information games is not at its peak. In the case of the Swiss card game Jass, previous work showed that the best bot at the time could compete with active amateur human players with over 10 years of experience on average[6]. Since the human vs. AI experiments in the previous paper are scarce, these will be the focus of this paper to continue the research on AI in cooperation games.

The agent used implements a Determinized Monte Carlo Tree Search (DMCTS) algorithm for the card selection and a Deep Neural Network (DNN) for the trump selection. In this paper, we first look at the current state of research according cooperation games and the previous state according the Jass game.

Our main contributions are the implementation of an optimised Graphical User Interface (GUI) and the connection with the currently strongest Jass agent. Furthermore, we provide details according to the conducted experiments such as setup, performance and limitations to help future researchers.

The sobering truth of the research question is that the best Jass agent cannot consistently win against well-practiced Jass teams. Potential reasons for this and a statement about the agent's performance level are described in this paper.

ACKNOWLEDGMENTS

First, I would like to express my sincere thanks to Joel Niklaus, who made this work possible for me as a supervisor and was always able to give me advice in difficult situations. I would also like to thank Prof. Dr. Matthis Stürmer as head of the Research Center for Digital Sustainability (RCDS).

Furthermore, I would like to thank Prof. Dr. Thomas Koller from the HSLU for providing the Jass agent and answering my countless emails. Next, I would like to thank all the participants in the experiments who pushed the Jass agent to the limit with their Jass skills.

In this context, I would also like to thank the company IPT, who sponsored the Jass Challenge Event at the University of Bern, which was organised as part of the work. (I am also thankful to Neville auf der Maur for spontaneously providing a mobile WLAN router).

I would also like to thank my parents and my two brothers, who always believed in me and supported me throughout the years of my studies.

I am also grateful for all the friends I have made throughout my life and for my girlfriend, who has always been able to motivate me. (Not to forget thanks to my cat as a permanent lap warmer during the writing of this thesis).

With the completion of this thesis, a new chapter in my life begins and an old one ends. I could never have written the chapter I am closing without all of you. Thank you so much!

CONTENTS

LIST OF FIGURES

LISTINGS

ACRONYMS

AI	Artificial Intelligence
API	Application Programming Interface
CNN	Convolutional Neural Network
DMCTS	Determinized Monte Carlo Tree Search
DNN	Deep Neural Network
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ISMCTS	Information Set Monte Carlo Tree Search
JS	JavaScript
JSON	JavaScript Object Notation
JSX	JavaScript XML
MCTS	Monte Carlo Tree Search
RB	Rule-based
REST	Representational State Transfer
STD	Standard Deviation
T-DMCTS	Time-based Determinized Monte Carlo Tree Search
UCB	Upper Confidence Bound
UML	Unified Modeling Language

Part I

THEORETICAL PART

1 INTRODUCTION

In this chapter we give a brief overview of the card game Jass and outline the contribution this thesis brings to the research on AI in cooperation games.

1.1 SWISS CARD GAME JASS

Jassen[1] is described as the "people's sport" No. 1. The game is traditional and multifaceted. It is widespread across all Swiss regions, languages and cultures[5]. In this paper we deal with the so called Schieber, a special variation of the Jass game.

In this often-played variant, the exactly 4 players each receive 9 cards of the 36-card deck. The deck is divided into 4 different colours with 9 cards each.

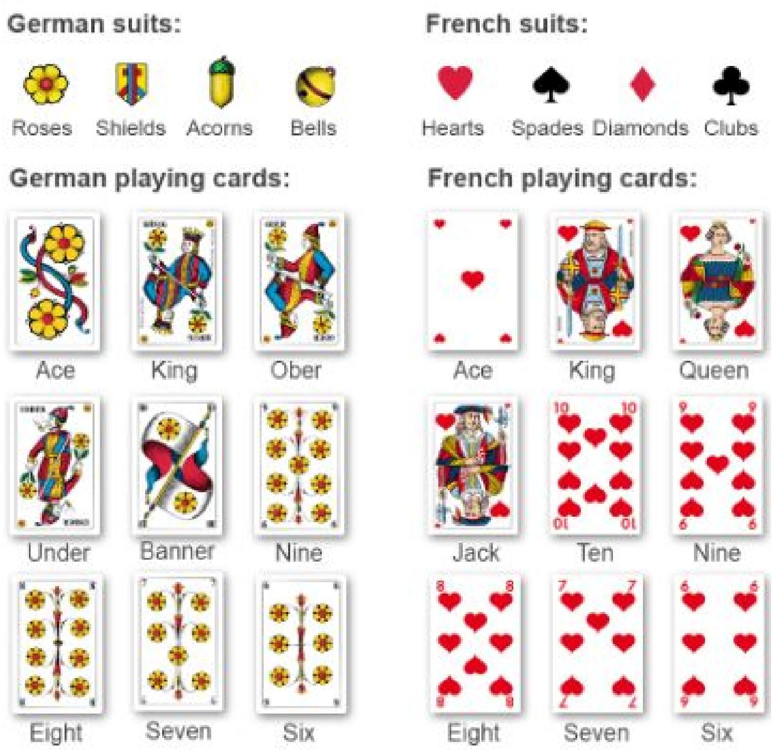


Figure 1: Presentation of the Cards. Note that there are two types of cards. The French and the German cards. Depending on the geographical location, one or the other is more popular in Switzerland.

1 This is the name given for playing the card game Jass in Switzerland.

The 4 players are divided into teams of two and sit opposite each other. The Schieber is then played in counter-clockwise rounds. Each player lays down one of his cards and whoever has given the best card gets the trick. The value of the trick is equal to the point value of those four cards on the table. The last trick of a round counts an additional 5 points. After the 9 tricks, the round is over and the 157 points are divided between the two teams.

At the beginning of each round, a trump is chosen by one player. In the next round, the trump indicating player is changed counter-clockwise. The value of the cards in this round depends on the chosen trump option. There are three scenarios for trumps: First, the player can declare one of the four colours to be the trump. Secondly, the player can declare "Obeabe" or "Undeufe" and the last possibility is the "Schiebe" from which the game variant got its name.
In the case of a Schiebe call, the player passes the responsibility of the trump decision to their partner, now the partner can not call Schiebe again but has to pick one of the 6 trump options.

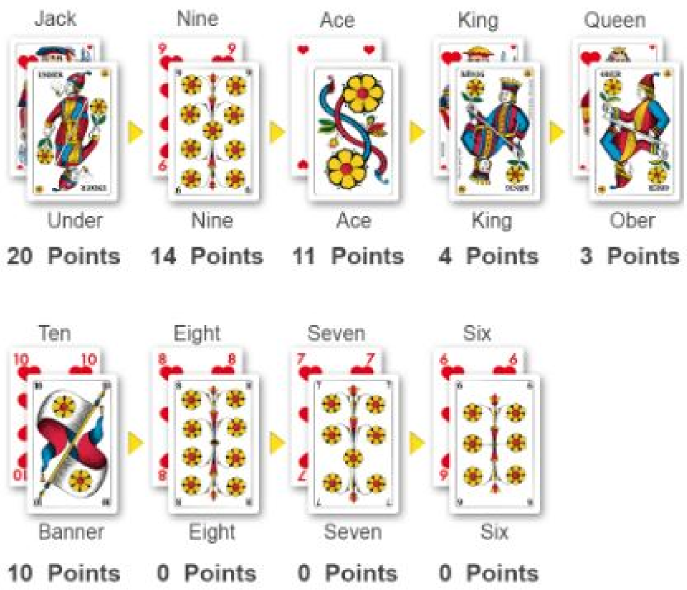


Figure 2: Example of the card values in the case of a heart/rose trump selection.

More exact card values for other colours and trumps can be found on the Swisslos webpage. A set of rules can also be accessed there[2].

2 https://www.swisslos.ch/en/jass/informations/jass-rules/principles-of-jass.html.

1.2 MOTIVATION

Games with hidden information make it difficult for an AI to calculate good moves. Unlike chess, where every possible move is visible to both players, in a imperfect information game like Jass, a player only knows his own cards.
It is particularly difficult to plan over several moves, since the number of possible card distributions and moves of the opponents is overwhelming.
However, hidden information as a challenge for AI has already been mastered in other games such as Texas hold'em poker[1]. In this example, the AI plays alone against all the opponents and, over time, defeats them.

Cooperative games like Hanabi are another challenge for AI [7]. In the game of Hanabi there is also hidden information, but the players exclusively try to work together to win the game. So Hanabi belongs to the category of cooperative games with imperfect information. In the Jass variant Schieber, the aspects of cooperation and competition are combined by having two teams that play against each other. Additionally in the Schieber we have to deal with hidden information. This makes it even more difficult for the AI to perform well, which is why this game is particularly suitable for testing and challenging the best algorithms.

In a previous paper, it was shown that the best AI can perform at a similar level as well-rehearsed amateur teams[6].
In experiments, 6 human teams were able to score 49.5% ± 14.2% of the points against a Time-based Determinized Monte Carlo Tree Search (T-DMCTS) agent.
The aim of our work is to conduct new experiments with strong human teams against the current strongest agents and then also to analyse potential reasons for the performance of the agents and the limitations of the experiments.

2 BACKGROUND

2.1 GAME THEORY

In this chapter, some background information on game theory is given to be able to compare the Jass variant Schieber with other games. There are various aspects with which games can be distinguished from a theoretical point of view.
In the following section, the most relevant classifications for our game are presented.

2.1.1 *Imperfect Information Games vs. Perfect Information Games*

A very important distinction in card games is the amount of game information available to players.
There are the games with perfect information, e.g. chess or mill. Games with more than two players can also have perfect information, e.g. Mensch ärgere dich nicht (Ludo). In this example, there is even a random factor built into the game with the dice. In perfect information classification, the only relevant factor is whether the players can always see their own and all other players positions.

In games with imperfect information, on the other hand, something is withheld from the player and thus the player must make assumptions and try to evaluate the situation. The player does not know for sure what the other players have. Games like Rock, Paper, Scissors and most card games like Poker or Uno are imperfect information games.

2.1.2 *Cooperation Games vs. Competitive Games*

Another important theoretical distinction in games is the amount of cooperation. There are purely cooperative games in which the players try to achieve a good result together.
For example, Hanabi is a game that includes imperfect information, so the players must cooperate strategically. Furthermore there is no competition among the players in the Hanabi game. In cooperative games with competitive aspects, players can join in a coalition to increase their chances of a good outcome. E.g. Monopoly or Catan. Games in which there are teams by default also belong to the cooperative games.
In contrast, there is the class of purely non-cooperative games in

which each player independently decides on a strategy. For example, chess or poker.

2.1.3 *Strategic Games vs. Non-Strategic Games*

There are more categories that can be used to classify a game. However, these factors have a lesser influence.
In a strategic game, each player has an influence on the outcome of the game. The non-strategic games are often called games of pure chance like rolling a dice and the higher number wins.
In between are the strategic games with a luck-based aspect, such as the previously mentioned Ludo.

2.1.4 *Sequential Games vs. Simultaneous Games*

Sequential games have a regulated flow in which only a certain player is allowed to make a play. Therefore, in these games it is not a matter of reaction and speed.
The contrary to this are simultaneous games in which various players can be active at the same time, e.g. the "doubling" in Uno.

2.1.5 *Constant-Sum Games vs. Non-Constant-Sum Games*

Constant-sum games are characterised by the fact that no matter how the game develops, the payout of all players remains at a constant level. In the special case of zero-sum games, this value is zero.
A simple example of this is the rock, paper, scissors game, because in a draw, the value of the game is zero and no one benefits. If one player wins, the other loses and the theoretical payoff is again zero.
In contrast, consider non-constant-sum games. Players can profit depending on their strategy.
A well-known example of a non-constant-sum game is the Prisoner's Dilemma.

2.1.6 *Finite Games vs. Infinite Games*

As a final classification, a distinction can be made between finite and infinite games. While finite games reach a predetermined end state, infinite games can last forever.
The vast majority of games are finite in nature, but we can consider a situation in games that have options to steal points from opponents that could theoretically create an infinite cycle.

2.2 THE CARD GAME JASS

Now that we have laid out some important aspects of game theory, we turn to the concrete example of Schieber Jass.
This variant of Jass belongs to the games with imperfect information. It is a aggregation of cooperative and non-cooperative games, as it combines both aspects. Schieber can also be classified as a sequential game with a constant sum.
In the variant played by the experiment participants, there are exactly 157 points in each round and the game has a finite outcome after totally 16 rounds. The reason for this is, that a tournament-like setup was chosen for the experiments in order to increase the significance of the evaluation.

The strategic Schieber game also presents itself with a portion of luck in the card distribution. We were able to eliminate this factor through so-called double rounds. How this looks like exactly is mentioned in more detail within chapter 5: Experimentation Setup.

Figure 3: The player "Roman" makes clubs trump here. Afterwards, the computer plays the "6 of hearts" to his right and thus has no trump in his hand. The longer the round lasts, the more information a player can collect, depending on the cards played. The partner "Nadja" and the second computer opponent call. Now the player "Roman" must play again depending on the information he got from this trick.

To get to the bottom of the Schieber variation on a mathematical level, we like to refer to figures from a previous paper. In a search tree used to algorithmically find the best move, there are $36! = 3.72e41$ possible paths. In Schieber there are $2.13e19$ possible card distributions. If we now only consider the allowed moves and multiply them by the possible card distributions, we still get $1.16e28$ game states, which is an enormous number of possible states that an algorithm has to consider in order to select a card[6].

Comparing the Schieber Jass to similar card games is not trivial, as the number of players and the gameplay differs. In Skat as one of the most familiar games, there are around $8.2e16$ different card distributions[3]. How many possible states there actually are is not mentioned.

In case of heads-up limit Texas hold'em poker, the number of possible states is roughly $10e13$. This simple version was solved close to perfection. In HUNL(heads-up no-limit) poker, on the other hand, there are $10e161$ possible states[1], many of which hardly differ because of the numerous branches created by slightly different bidding[1].

2.3 AI PERFORMANCE

In order to further advance the development of AI, it is necessary to constantly compare and evaluate the state of the art algorithms. It is popular to test AI against other AI to get a picture of their performance. This method can also generate a lot of data at little cost in a short time.

Ultimately, we want to see how the AI performs against the human. In direct comparison, the following classifications can be made: sub-human, par-human, high-human and super-human AI[6]. In other words, these terms represent the categories: worse than the average player, similar to most players, better than the majority of players and definitely better than any player.

We would expect an AI to be able to perform in the high-human/super-human range else it is not impressive enough.

If a game has a low complexity, e.g. Tic-Tac-Toe, an AI cannot reach the super-human level, because it is always possible for an human to play a draw since the strategy is too trivial. In many games with more complexity and perfect information like chess, the AI has already reached a super-human level for decades[2].

How the development of AI behaves in other games especially with imperfect information is shown in the following chapter on related and prior work.

1 For example, it is almost the same whether someone bids \$100 or \$101
2 https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/

3 PRIOR WORK

In this chapter we present the related work. We also discuss the most relevant state-of-the-art algorithms for the Schieber Jass game. Finally, the approach used in this thesis is put into perspective.

3.1 RELATED WORK

In this section we refer to relevant papers regarding AI vs. Human research in card games. To start with, we consider a game called Hanabi, where tests with a Rule-Based AI and an Other-Play[1] AI were made[7]. The results were comparable, but the experience for the player was very different in the feedback. Despite the fact that the Other-Play AI performed better, the participants liked to play with the Rule-Based AI over the Other-Play AI.

The game Hanabi is not very similar to Jass, as all participants work together to achieve a good result. However, this makes Hanabi a highly cooperative game and this aspect is also important in the Schieber variant of Jass.

The game of Solitaire, which is a 1-player game, is also not directly similar to Jass. In an experiment with Solitaire, the AI used was as well able to outperform human capabilities and thus reaching a super-human level[3]. In the paper there was used a deterministic Markov decision problem with more the 52! (Approximate $8.0e67$) states. This huge number got reduced with determinism, still it takes close to two hours to finish one game.

The card game Skat comes closer to the Schieber Jass. There are also Skat-like variations that are played in Switzerland under a different name[2]. Unfortunately, the paper does not test the AI in a direct bot against human experiment. However, they did a selection of characteristic games and put them in comparison in different aspects[3]. The AI seems to beat the human significantly in Nullspiel and Null Ouvert. This is the conclusion after evaluation of around 2.5 million games.
However, a statement about the AI level would be difficult because we don't know exactly how good these players were, as the paper leaves open how the AI would perform in a tournament with the best skat

1 The Other-Play AI is designed to avoid the creation of "secretive" conventions that can result from self-play training. This AI assumes, that its teammates are also optimized for zero-shot coordination. However this is not what a human Hanabi player actually does.

2 https://www.jassverzeichnis.ch/index.php/blog/77-bieterjass-jassen-zu-dritt.

players. Therefore, the high-human level is appropriate for the time being.

Another example for an AI performing on high-human level is the Big2AI 4.0[2]. The used AI consists of a Monte Carlo Tree Search (MCTS) algorithm with a framework designed to predict multiple movements of multiple opponents. In experiments human vs. AI only the expert players were able to finish with positive points.

In heads-up no-limit Poker, the AI Liberatus outperformed the best players in the world. The AI used implements a Monte Carlo Counterfactual Regret Minimisation algorithm.
With this feature Liberatus can exploit the human's ability over time to find vulnerabilities through a self-improvement module and even use the found tactics as a universally valid strategy. The experiment in this case was carried out in a match with four top HUNL(heads-up no-limit) specialists and the AI won by a clear margin[1]. This performance was measured in milli–big blinds per game (mbb/game) and the AI defeated the humans in this experiment by 147mbb/game. In this experiment, many well-known professionals were recruited, and a large prize pool was offered as an incentive. Liberatus was able to defeat all participants and therefore reached a super-human level.
There are more related card games in which AI agents are being tested[3].

3.2 prior work

The prior work dealt with the development of the optimal algorithm for the Schieber variant. The different approaches were compared in bot vs. bot experiments and the best agents were finally tested in human vs. bot experiments.

3.2.1 *Rule-Based Systems*

A Rule-based (RB) approach is not real artificial intelligence, as the intelligence is contained in the knowledge base. It cannot think for itself, but only carries out what a human has given it. Nevertheless, rule-based AI can also be used in more complex games. In Hanabi, the experiment participants even preferred the RB agent because it is more comfortable to play with[7].

In Schieber Jass, an RB agent has already been tested. It performed better than an agent that plays random cards, but its performance remained below that of the MCTS variants and DNN[6]. Still RB ap-

3 See: Arneson, R. B. Hayward, and P. Henderson, "Monte Carlo tree search in hex," IEEE Trans. Comput. Intell. AI Games, vol. 2, no. 4, pp. 251–258, Dec. 2010. G. Tan, Y. He, H. Xu, P. Wei, P. Yi, and X. Shi, "Winning rate prediction model based on Monte Carlo tree search for computer dou dizhu," IEEE Trans. Games, vol. 13, no. 2, pp. 123–137, Jun. 2021.

proaches can help and support specific parts also in MCTS variants for example.

3.2.2 *DNN Agents*

In the previous work on Jass AI, a DNN agent was trained[4]. The input for the DNN with six convolutions layers is a 4x9 matrix with information about the current game state.
This game state consists of data about the current round. This includes the tricks played in the right order, as well as the current player, the dealer and the hand of the current player. This DNN for card play was then also used in some DMCTS variants[5].
There was developed an individual DNN for the trump selection, as only the hand and position of the player is relevant in this case. It was noted that the precision of the network had certain diminishing returns and did not improve after a specific point by increasing the depth. In a direct comparison against other agents, the DNN variant could defeat the RB agents and had a similar performance as the DMCTS agents.

In the trump selection phase, the DNN approach delivered the best results among all agents. It is mentioned that a possible reason for this is that the MCTS approach very rarely calls Schiebe in comparison to the DNN agent[6].

3.2.3 *Monte Carlo Tree Search Systems*

The last approach presented is the MCTS algorithm. This method is used in many of the studies discussed in section 3.1: Related Work. For example, in the papers on heads up no limit poker, Big2 but also in Skat, where an MCTS approach created by Kupferschmied and Helmet is applied [4].

In the previous work about Schieber Jass, Determinized MCTS methods DMCTS and Information Set Monte Carlo Tree Search (ISMCTS) are implemented[6]. The MCTS algorithms are tested with different numbers of iterations. Similar to the DNN approach, increasing iterations and determinations above a certain threshold does not yield any additional benefits.
As already known, the MCTS methods were outperformed by the DNN agent in the trump selection. In the card play phase, however, the Time-based Determinized Monte Carlo Tree Search was able to keep up with the Convolutional Neural Network (CNN) agent and therefore outperform the RB approach. DMCTS and CNN are only outper-

4 On a dataset from Swisslos consisting of 1.8M played rounds.
5 Probability Determinized Monte Carlo Tree Search and Probability Information Set Monte Carlo Tree Search.

formed by a cheating MCTS algorithm, which, as the name suggests, has forbidden information about the cards of the other players[6].

3.2.4 *Prior Experiments*

In addition to the bot vs. bot experiments, there were also human vs. bot experiments conducted in the previous work on the Schieber Jass game.
The aim was to assess the strength of the bots, but also to get feedback from the players on how it feels to play with a bot. Experiments were conducted with two humans vs. two bots, as well as human and bot vs. two bots. In contrast to the bot vs. bot experiments, single rounds were used, as the human player could remember the cards when played with double rounds.[6] The participants were asked to fill out a feedback form after the experiment to gain knowledge about how they rate the bot.

The limitations mentioned are the procurement of large amounts of test results but also the question of the strength classification of the human players.
In the human vs. bot experiments there were 6 human teams playing 136 rounds against the T-DMCTS bot team. The humans were experienced players and scored 49.5% ± 14.2% of the overall points against the T-DMCTS players.
In these experiments, it is concluded that the AI does not have a clear upper hand against average human players but can compete with them. The variance was very high because no double rounds were made. We will not go into details about the human-bot paired experiments, as only human vs. bot experiments were done in this paper, as these are more significant for the strength classification of the AI.

3.3 CURRENT AGENT

The bot used in this thesis is the same DMCTS agent that was shown in the previous section 3.2.4 Prior Experiments. The DMCTS agent uses a configuration of 1000x1000 where the number indicates the amount of determinitazions and rollouts. These settings proved to be the most effective in the previous work and were therefore retained.

The agent used in this work uses a CNN approach for trump selection instead of the RB method. In terms of technical context, the current agent is implemented in C++[7] and uses the Upper Confidence Bound Upper Confidence Bound (UCB) algorithm in the MCTS implementation to ensure that possible variants that have not yet been

6 This gives a huge randomness factor to the experiments. In section 5.1.2 Double Rounds we introduce a possible way to implement the double rounds into human vs. bot experiments.

7 The previous agent was implemented in Python.

visited are selected first.
The use of this algorithm is considered state-of-the-art for MCTS methods and is implemented in many other research on AI[4].

Part II

PRACTICAL PART

4 IMPLEMENTATION

This chapter explains the steps involved in the programming part of this thesis.

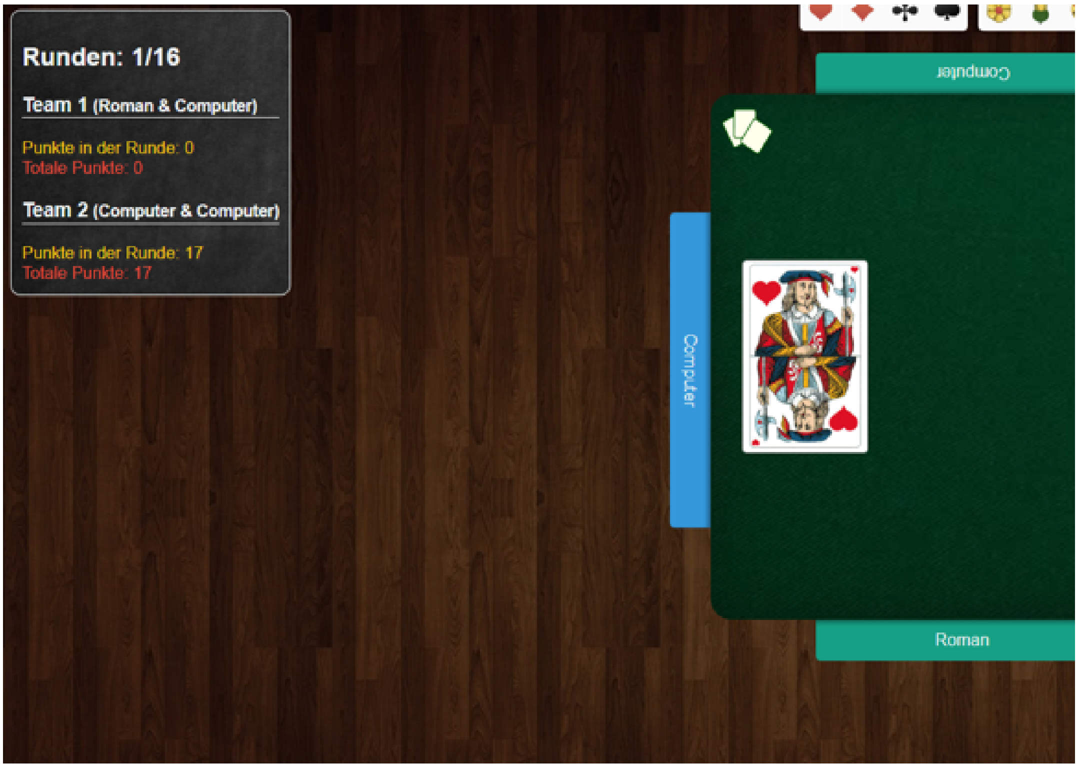
4.1 INTERFACE UPDATE

The git repository of the Jass server used in this work is a fork of an already existing Jass server[1]. Since the experiments are the focus of the work, we used an existing project that already implements a GUI. For the use of our experiments, we adapted the GUI in several places. By playing with the GUI, some revealing factors can be eliminated. There can be no cheating when shuffling the cards and the counting of the card values is always correct. It also prevents misplaying and pre-playing of cards.

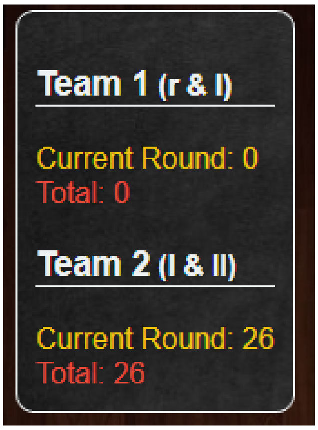
The codebase of the project is written in JavaScript (JS) and for the client-side GUI, JavaScript XML (JSX) was used. Since our experiments have a round-based system, we adapted the score screen accordingly. To achieve this, we implemented round indicators on the server side, which can be queried by the client side.
The language change from English to German is also indicated, as it was expected that the majority of the experiment participants would have an easier time with the German language. Therefore, the change from English to German was also made in all other parts of the GUI to be consistent.[2]

1 https://github.com/roman1489-afk/jass-server-ui.

2 See: Figure 4.



(a) New environment with updated score screen on the top left corner of the GUI. It shows the current round and Teams.



(b) Old score screen in English without round indicator.

Figure 4: Jass GUI changes.

The old interface contained a "suggestion engine" in the form of a clickable button that suggests a possible move to the player. This was deactivated with the intention that players might be distracted by it. In its place, the logo of the company IPT was used, as their sponsorship enabled a Jass Challenge Event at the university of Bern where the current agent was tested live[3].

(a) IPT advertisement in the GUI. (b) Old suggestion engine in the GUI.

Figure 5: Further GUI changes.

4.2 CARD DISTRIBUTION

In the preparations for the experimental setup, it was first decided not to distribute the cards randomly, but to prepare them in advance in certain compositions[4].

These card settings also had to be built into the code. To achieve this, we adapted the creation of the deck on the server side. Instead of the random card distributions that used the .shuffle function, the cards are distributed manually for each round in our implementation.

Listing 1: Manual card distribution.

```
switch (rounds) {
   case 1:
      // game 1, round 1
      helpSort(range(0,26),
          [22,21,17,35,11,7,16,24,22,27,15,35,33,27,
      24,20,22,28,24,34,24,24,22,3 ,31,27,29]);
      break;
   case 2:
      helpSort(range(0,26),
          [32,24,12,4,28,22,25,17,21,12,10,14,18,26,
      24,18,28,31,22,33,26,34,27,23,27,27,26]);
      break;
```

3 See: Section 5.3.1 Jass Challenge Event or Appendix A for more information about the event.

4 See: Section 5.1.3 Pre-set Card Discussion.

```
(...)
```

This shows two cases of card distributions in the code. Only 27 of the 36 cards had to be assigned to a specific player because the last 9 were then automatically assigned to the fourth player.
Another feature that makes up our experimental setup are the double-rounds[5]. For the realisation of this approach, we have modified the manually defined decks with a helper function.

Listing 2: Helper function double-rounds.

```
/**
 * Helper function to shift the existing deck, so that the hands
     of players are switched
 * @param {number} numbersOfShifts indicates the number of
     switches
 * @returns {*} the deck in correct order, that players will get
     different hands
 */
function shiftDeck(numbersOfShifts){
   return deck.cards.push.apply(deck.cards, deck.cards.splice(0,
       numbersOfShifts));
}
```

It takes a pre-set card deck and shifts it counter-clockwise. The 8 pre-set card decks are used twice, and this results in the 16 rounds per game.

4.3 INSERT NEW AGENT

Another task during the implementation was to replace the existing bot with Thomas Koller's new agent.
To realise this, according to the Application Programming Interface (API) documentation, we required a well-defined and formatted game state in JavaScript Object Notation (JSON) format, which is sent to the specific endpoint of the agent server using a Representational State Transfer (REST) API request.

4.3.1 *REST-API*

REST API, like other APIs, is used to exchange data between client and servers. REST typically uses the Hypertext Transfer Protocol (HTTP) protocol and transmits the data in the earlier mentioned JSON format. The REST API used was able to process the requests very quickly and sometimes provide a response in less than one second. This allowed the bots to play faster than some human participants would have been comfortable with.
For the concrete implementation, two different request types both

5 See: Section 5.1.2 Double Rounds.

using the same game state were needed. The requests are triggered when a corresponding message is recognised by the server side. The more frequently used endpoint was the card selection endpoint and the less frequently used the trump selection endpoint.
In a game with two agents playing 16 rounds, card requests had to be executed exactly 288 times and trump requests 8 times respectively.

Listing 3: API card request.

```
/**
 * Fetches an API card request to a bot.
 * @param {JSON} currentState this is the gameState of the
     current Game
 * @returns {Promise<*>} if successful we get a card, chosen from
      the bot
 */
async function fetchApiCardRequest(currentState) {

    try {
        let response = await fetch('http://****-*****.****.ch
            /*********/****************',  {
            method: 'POST',
            retry: 5,
            pause: 2000,
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(currentState),
        });
        let data = await response.json();
        return data;
    } catch (e){
        console.log(e);
    }
}
```

The code shows the async function for the card-request.[6] The trump request has a similar structure. Since the server has no caching, the 5 retries at intervals of two seconds attempt to recover from a possible interruption in the WLAN connection. This could lead to a loss of the connection and the game flow. Finally the game will be terminated undesirably.
Before the response gets passed on to the client side, it must be formatted correctly[7]. As an example, we will take a response from the agent *"HK"*, which indicates that the agent would like to play the king of hearts. At the server side, however, the cards are specified as JSON-formatted objects. Here the king of hearts looks like this: *{number: 13, colour: 'HEARTS'}* .

6 The exact URL of the endpoint is censored.
7 See: Appendix B.

This conversion in the other direction is also necessary when creating and updating the game state[8].

4.3.2 *Game State*

The game state mentioned in the previous section is the essential component of the REST-API requests. The response of the agent depends on this data. A not well formatted JSON game state will cause the agent to crash. Furthermore, the correctness of the game state is important, as even small changes will influence the card selection and trump selection.

The code below shows an example of a game state in JSON format. There are static entities that are used for information purposes only. These include "version", "jassTyp" and "gameId". Other entities such as "trump", "dealer" and "forehand" are set at the beginning of a round and remain fixed until the end of the round. The remaining entities vary with each turn and therefore need to be adjusted regularly.

Listing 4: Game state.

```
{
    "version": "V0.2",
    "trump": 5,
    "dealer": 3,
    "currentPlayer": 2,
    "playerView": 2,
    "forehand": 0,
    "tricks": [
      {
        "first": 2
      }
    ],
    "player": [
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": [
          "DJ",
          "D9",
          "D10",
          "HA",
          "H7",
          "H6",
          "S8",
```

8 See: 4.3.2 Game state.

```
        "SJ",
        "D6"
      ]
    },
    {
      "hand": []
    }
  ],
  "jassTyp": "SCHIEBER",
  "gameId": 0
}
```


To ensure that the game state is always correctly available, the newly created game state class is supplied with update messages from several locations on the server side depending on what changes happen in the game. The game state is then passed to the REST-API function described in the previous section as "currentState".
One difficulty was the order of the player IDs. In our server environment, the first player gets the ID: 0 and then it increases counterclockwise to ID: 3. In the agent's system, IDs 1 and 3 are swapped. This must be taken into cosideration when converting between the notation systems.
In the game state, we store the card hands and the cards that have already been played. The inverted conversion mentioned in the section 4.3.1 REST-API is carried out here.

4.3.3 *Program Flow*

At the end of the implementation chapter, we would like to present a Unified Modeling Language (UML)-like diagram to visualise the process and the possible branches in the game flow[9].

The diagram shows the *Jass server* as the core of the project with server and client implementation. A new game gets started from the Jass server as soon as two *players* and two *bots* are connected. After this is setup, the game begins with card distribution and the corresponding game flow.
When a *player* performs an action, it affects the *new game* and the *Jass server*. Both are responsible for ensuring that the GUI and the *game state* are updated correctly at all times. When a *bot* performs an action, a well-formatted request is made to the agent via the REST-API. The request uses the provided *game state* to build its body. The destination of the request is marked as black box, because the *agent's* details are not the main part of the work.

9 See: Appendix C

After the *agent* has completed a card selection or trump selection, the response is returned to the *bot*. After reformatting, it is again the task of the *new game* and *Jass server* to carry out the corresponding updates.

5 EXPERIMENTAL SETUP

In the preparations for the experiments, various factors were taken into consideration in order to achieve the best possible results.
The precautions taken serve to optimise the significance of the experimental results but also the best possible performance of the participants. For the experiments we have chosen a round-based scheme, as this provides better comparability. The number of rounds was set to 16[1]. This way, each player can trump four times.

5.1 MINIMIZE RANDOMNESS

In section 2.2 The Card Game Jass we declared the Schieber variant as a game with strategy but also a portion of luck. Since luck factors are very detrimental to the significance of the experiments, we tried to eliminate them.
The participants in the experiment were deliberately not informed about all these modifications. They knew that there were no multipliers, but they did not know that some of the cards were not distributed randomly and that the double round system was used.

5.1.1 *No Multipliers*

As in most Schieber tournaments, the multipliers were omitted in our experiments. The Schieber is played without "Stöck"- and "Weispoints"[2]. All six possible trump variations are counted with the same factor. This results in a total of exactly 157 points for each game, including 5 points for the last trick.

There was only one change in case of a "match". This is the term used when a team claims all the tricks in a round and reaches the maximum score of 157 points. In this situation, the team was awarded a match bonus of 100 points. This measure was intended to give teams more incentive to play in a coordinated manner.
In fact, we could observe how a team was excited when they succeeded to get a match. However as the agent does not count on this reward of 100 points, but only tries to make the maximum points from the cards, we manually deleted the match bonus in the evaluation of the experiments.

1 This number was set after testing how low a game of 12/16 or 20 rounds last. 16 rounds proved to be feasible in terms of time.
2 In some variant of the Jass game one can get additional points by having a special constellation of hand cards.

5.1.2 *Double Rounds*

The concept of double rounds was copied from the previous work[6]. As mentioned in section 3.5 Prior Experiments, the concept was only used in the bot vs. bot experiments and not in human vs. bot experiments. However, the double rounds are the factor with the most potential to minimise randomness, as in a game no matter what happens both teams always have the same conditions with this concept. We have come to a solution using permutation in a way that the double rounds are not obvious.

For example, the cards of the first round are not played again in the next round, but later in round 10. Therefore, we developed a concept in which the first 8 rounds were individual and rounds 9-16 were the mirrored copies in a different order.
As the experiment participants did not know this, it was never noticed by anyone. Even with enlightened knowledge, it takes an incredibly good memory to gain an advantage and remember certain rounds.

5.1.3 *Pre-set Card Discussion*

The final arrangement was to not distribute the cards randomly, but to prepare 8 card arrangements and then fit them into 16 rounds using the concept of double rounds. The card distributions include card hands in which the players are challenged with problems that can be solved by the unwritten rules of the Jass game.
As examples, they test whether a player can pass to a teammate by correctly "verwerfen[3]" in an obeabe game or whether a player makes a trump when having nell and trump ace "zu fünft[4]" . The idea behind this encompasses two benefits.

First, we hoped to eliminate rounds that were too easy. That means rounds in which it is obvious even to an average player which trump leads to the best result and seven or more tricks are directly in hand[5]. In this case it would be impossible for a bot to make a difference. Even if all the cards were shown to the bot, the bot could only play equally well.
In hindsight, this first advantage was misjudged. If the cards are distributed randomly, there are by chance practically no distributions that are too easy on all the rounds. Additionally the pre-set rounds played into the favour of an experienced team, as a lot could be achieved with cooperation. Probably more than with randomly dis-

3 A concept where a player recognises on which colours his partner has good cards depending on the previously played cards. Can be translated with "Discarding".
4 The player has the trump 9, Ace and 3 other cards that are not any special, like 6,7,8,10 or Queen.
5 Consider, for example, a hand with 7 cards of hearts.

tributed cards. Cooperation, as will be shown in the experiment evaluations, is a deficit of the agent.

The second advantage we saw in the uniform evaluation. Since all experiments had exactly the same conditions, there should be better possibilities to make comparative evaluations. The second advantage proved to be very useful for better comparisons. Because the individual moves of the games were recorded, it is easier to understand why the agent in a certain round was bad in all the experiments.

5.2 USABILITY TESTS

Before the experiments were carried out, we set up two usability tests in which certain technical problems were eliminated and we also added a 1-second delay to all decisions of the bot, as the enormous speed of the bot was described negatively in the feedback of the tests.

A second adjustment was again a delay as soon as all cards were played. The cards were now left on the table for two seconds before they were cleared away. This created a more natural-looking Jass environment. The GUI does offer players the option to view the last trick, but with this variation, participants rarely needed this option.
Viewing the last trick is not actually allowed in tournaments, but since there is no possible intervention in the online environment to prevent the trick from being taken, it is still necessary. The option gives especially inattentive players an advantage over a real tournament situation, but it does not provide more advantageous information.

5.3 IN-PERSON EXPERIMENTS

The experiments could be conducted in-person by placing both participants at a table and giving them a configured laptop as a game device.
This meant more work for the participants to attend but also for the host to organize in comparison to the remote experiments. The advantage of this variant was the possibility for the participants to get to know each other better.
It was noticeable that the partners coordinated better before the game started and were able to benefit more from non-verbal communication due to their presence in the same room. Verbal communication related to the game was prohibited during the experiment.

5.3.1 *Jass Challenge Event*

A special form of in-person experimentation was the Jass Challenge Event at the university of Bern[6]. In order to promote the project of

6 See: Flyer Appendix A.

the Jass agent and at the same time to receive more experimental data from dedicated Jass players, this challenge was brought to life. In comparison to the other experiments, the participating teams did not only want to defeat the bot but were aiming to play out as much difference as possible. In the end, it was the highest positive difference that decided the winning teams of the event. The incentive to do well was further increased by distributing a prize money of 550.- among the best teams[7].
This may have influenced some of the teams psychologically to master their coordination in the best possible way. From several teams a lot of coordination-talk was observed before the start of the matches.

5.4 REMOTE EXPERIMENTS

As the effort was lower, many experiments were conducted remotely by having a zoom meeting or phone calls. The connection to the Jass server could be enabled remotely through an internal link. In this variant, other topics than Jass were rarely exchanged, and it was generally quieter in comparison to the in-person experiments.
There was a host who was present and checked that people were not communicating verbally. For optimal performance of the Human teams, the in-person variant is preferable.

7 The sponsor IPT made it possible for the Jass Challenge Event to take place in this form.

Part III

EVALUATION

6 EXPERIMENTAL RESULTS

In this chapter, the results of the experiments are presented and discussed. In the course of the work, there were two series of experiments. All experiments were logged in JSON files. This enabled us to analyse the points scored based on this data. It also made it possible to look for specific outliers and to analyse the corresponding course of the game.[1]

6.1 EXPERIMENTAL ROUND 1

The first round of experiments had all the participating teams play with the same cards. The first round included many in-person experiments, including all the experiments from the Jass Challenge Event. In all experiments the concept of double rounds was used.
The data set includes 12 teams, which played 16 rounds each. This results in 192 rounds.

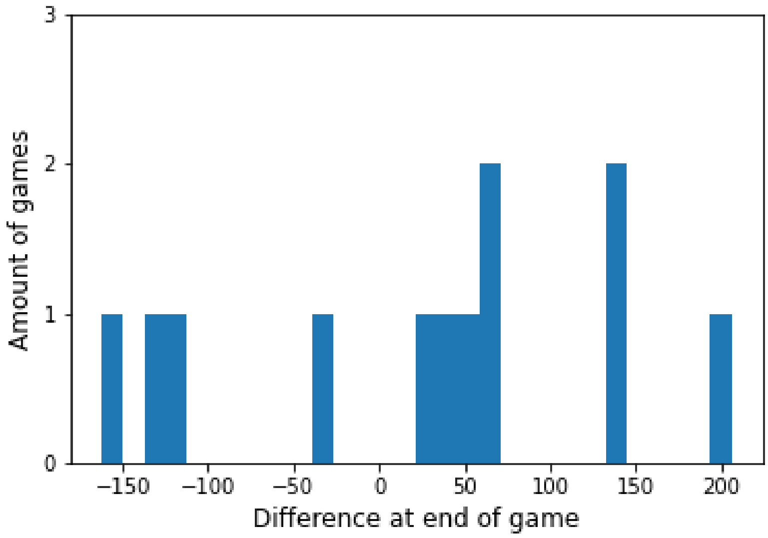
6.2 EXPERIMENTAL ROUND 2

In the second round, the cards were randomised, and the majority of experiments were done remotely. In all experiments the concept of double rounds was used.
The data set includes 13 teams, which played 16 rounds each. This results in 208 rounds.

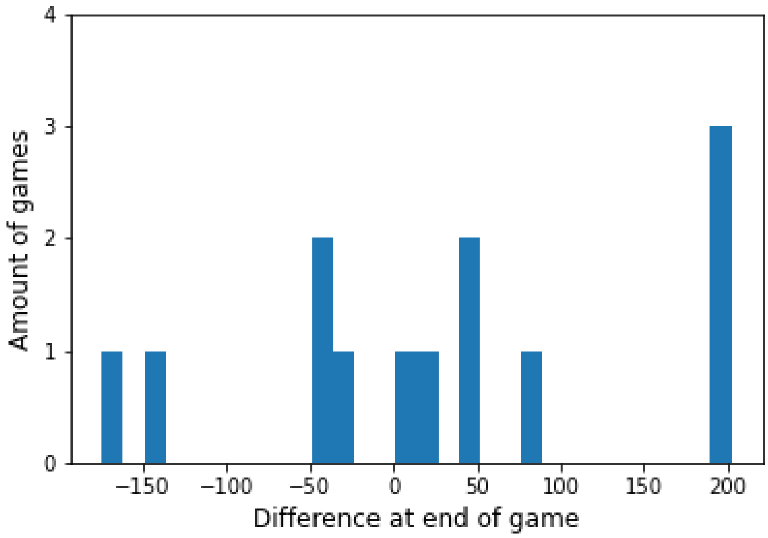
6.3 RESULT COMPARISON

When considering the results, the difference we talk about is from the point of view of the human teams. A difference in the positive range means that the human team has performed better.
A difference in the negative range indicates a better performance of the bot team.

1 Find all the results and logs under: https://github.com/roman1489-afk/jass-server-ui/tree/develop/experiments



(a) Experimental round 1.

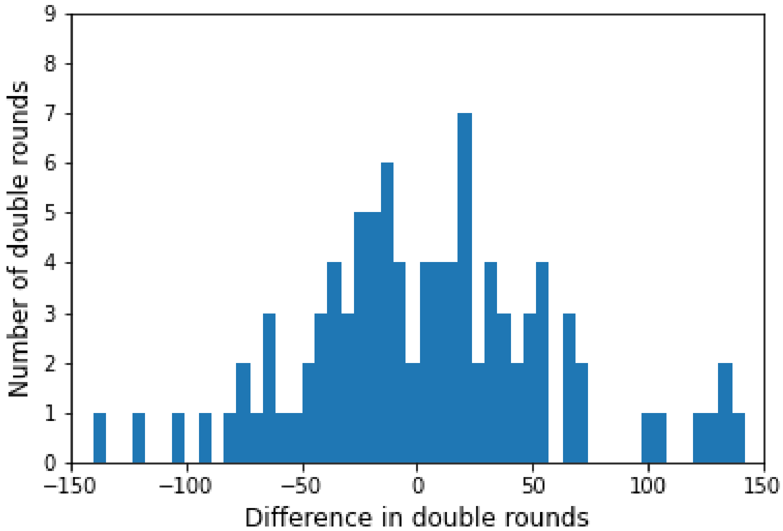


(b) Experimental round 2.

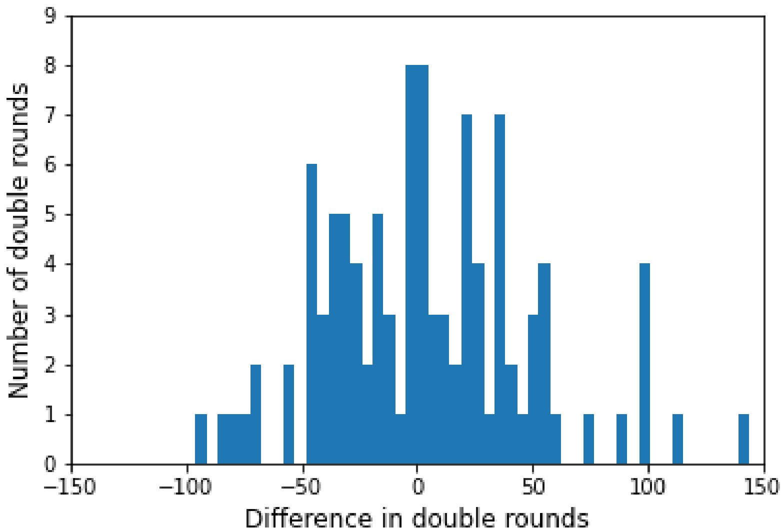
Figure 6: Result of all conducted experiments.

The first comparison of results shows the difference at the end of the 16-round games in histogram format. In the histogram (a), the human teams won 8 of the 12 games. The scatter of the results is very high, which can be explained by the limited size of the data set when only looking at the games.
In the histogram (b), the human teams won 7 of the 13 games and again there are many outliers. Purely in terms of victories, the human teams were able to beat the bot team. If we look at the exact points scored in all games, the statistics again tend towards the human teams. They were able to score 50.48% ± 17.53% of the points

in the experimental round 1. The large Standard Deviation (STD) can be explained by the nature of alternating trumps in the Jass game, as many of the results range between 110-140 and 30-50 points. While the mean value is 79.25 points.
The experimental round 2 was similar, with the human teams scoring 50.57% ± 14.27% of the points with a mean value of 79.39 points. However, the STD is somewhat smaller here. A reason for this is found in the next histograms.



(a) Experimental round 1.



(b) Experimental round 2.

Figure 7: Result of all played double rounds.

The second comparison deals with the double rounds. We look at all double rounds of all teams in the respective experimental rounds. Since in the individual double rounds both teams make a trump once with the same cards, it should be possible for the bot not to grant the human teams any above-average double rounds. However, the two histograms show us the opposite. The mean difference of experimental round 1 over all double rounds is 3.975 points and in experimental round 2 it is 3.59 points.

A double round consists of two rounds at 157 points, i.e. 314 possible points. There are some double rounds in which differences of more than 100 points are recorded. The outliers in the negative range are cases where the human teams had poor coordination or dared to take a risky trump selection which did not pay off. Some of the cases where the points are massively in favour of the human teams are mentioned separately in section 6.2 Result Discussion, as it is crucial for the future agent development to know what led to such enormous differences.

Due to the larger amount of data for the double rounds, we achieve a recognisable Gaussian normal distribution in both histograms. Compared to the first experimental round, the differences in the experimental round 2 are clearly narrower in the range -50 to +50 and many results at the zero point. The STD confirms this observation, as it is 54.57 points in experimental round 1 and only 44.58 points in experimental round 2. It is possible that the random card distributions led to more moderate rounds because the rounds less often advantaged one team.

To address the case, that certain teams could be responsible for only the negative values and certain for only the positive values, we conducted two-sided t-tests for all the teams.
As it turned out, the described scenario did not happen. Two-sided t-tests were made for each of the 25 games (12 in experimental round 1 and 13 in experimental round 2). In which we took the differences in the double rounds as data. They all fulfilled the normal distribution requirement, but no empirical t-value was found in all 25 games that would satisfy a significance level of 0.95.

Therefore, the null hypothesis: *"There is no difference between bot and human"* was kept as correct in all cases. This means that the best human teams also lost certain double rounds and the human teams that could not beat the bot overall also won in certain double rounds.

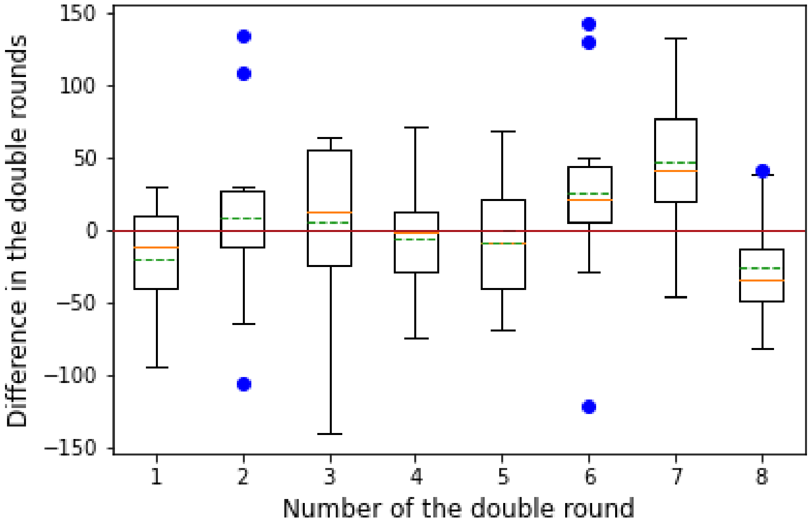


Figure 8: Experimental round 1.

The last diagram shows a box plot of experimental round 1. In this case, the pre-set card distribution discussed in section 5.1.3 Pre-set Card Discussion comes in handy. Only in this case is it possible to generate such a box plot.
We see all 8 double rounds of experimental round 1. Since all double rounds were played by all 12 teams, we have 12 data sets in each box. We are again interested in the differences of the double rounds.

The box plot clearly shows that not in every double round the teams performed equally. For analysing the bot, it is again very useful because we can look at the outliers. We also see that double round 6 and especially double round 7 seemed to have been very favourable for the human teams, as the boxes are entirely in positive range. However, double round 8 seems to have been very difficult for the human teams. It therefore appears appropriate to look at these rounds in more detail in section 6.2 Result Discussion.
In order to classify this result statistically, we also made two-sided t-tests. The 12 differences from the corresponding double rounds served as data.

For rounds 1-6, we must leave the null hypothesis *"There is no difference between bot and human"* as it is, because the empirical t-value does not reach the critical t-test for a significance of 0.95. So, there is no statistical difference between the human and bot team in these rounds. In round 7, however, the empirical t-value clearly exceeds the threshold, and we can say that there is a difference in favour of the humans between the human and bot team. The p-value in this case is actually 0.010, i.e. the significance is substantially higher than 0.95.
In the case of double round 8, on the other hand, it can be shown with

a significance level of over 0.95 that there is a difference in favour of the bot team.

6.4 result discussion

The statistical evaluations cannot determine a clear superiority of the agent but neither of the human teams. In the conducted experiments, we found that the human teams were able to defeat the agent more often and score more points in the overall game as well as in the individual double rounds.
If we base ourselves on the experiments, we can conclude that the agent is not on a super-human level. The level high-human level is debatable - there are well-founded pro and contra arguments.

From a pro point of view, it can be mentioned that the participants in the experiments were strong and experienced players, certainly above the average player. According to the feedback forms, the participants rated themselves on average with 3.6/5 points in relation to their jass skills. In feedback questionnaires, people rated themselves modestly, for example, no participant rated themselves with a score of 5.
Most of the teams have played Schieber in a few games or regularly with each other and have well over 10 years of experience with the game of Jass.
The bot was not able to win consistently against his opponents. It is possible that the AI could have done so with many more experiments[2]. Statistically, we cannot substantiate this. It is therefore possible that the agent could be set at a high-human level as we could expect him to beat an average human team.

In the following, we would like to discuss some examples in which large differences in scores were found. In some places the agent has a clear deficit, which occurs repetitively. This fact and the defeat over the entire experiments are arguments against placing the agent on a high-human level.
Because double rounds were used in all experiments, we can make direct comparisons between the human teams and bot teams in all situations. For each of the situations, the decisive card hands and game sequences are provided in JSON format[3].

For the following example situations, Appendix D contains all relevant player hands and game progressions in JSON format that are considered in the according situation.

2 Consider the Situation 3 later in this section as a reason for this argument.
3 See: Appendix B for naming conventions.

Situation 1:

The bot makes diamonds trump. In the first trick, 3 players come out with trumps. In the second trick, only the bot and its partner are calling trumps. The decision to play diamonds (trump) again in the third trick is not correct at all, because it pulls a trump from the bot's partner. This could have been prevented by simply counting the trumps. A beginner's mistake.

Situation 2:

The bot plays forehand obeabe. With this hand its certainly an option, but there is the risk of having only 3 safe tricks.[5] The majority of human teams play hearts trumps forehand, which is certainly the less risky option.
After the two safe tricks with the aces, the bot decides to play out the *"HJ"*, here the game slips into the hands of the human team and the bot even has to throw away his third ace at the end. The bot team only scores 43 points on their own trump.

Situation 3:

With this hand, one must call a trump after schiebe. The bot decides to play undeufe and scores 88 points. The human team calls clubs trump and scores 144 points. In the game, both teams made good moves. In this case, the bot cannot be blamed for a faulty or risky play style. There is no really good variation to play in this situation.
This situation shows that the decision about the trump can make a huge difference of many points. Perhaps the bot is right in this situation and in most cases of the remaining card distributions undeufe is more advantageous, but not in this one.
This example shows the difficulty in assessing the experiments, as in some cases a statistically better decision may lead to a worse result. To get around this, far more experiments would be needed.

Situation 4:

The human team plays undeufe and scores 130 points. The bot team also plays undeufe but cannot pass the game properly after the clear tricks and scores only 80 points. The human team indicates which colour is to be played by discarding[4] other cards correctly.
The bot team lacks this form of communication in a situation where it is quite possible to get more out of the game. Especially in the case of

4 Verwerfen.

obeabe and undeufe, this example of the bot's communication deficit is not an isolated case.

Situation 5:

The bot calls schiebe with this hand and then is gets called heart trump from the partner, which leads to a fatal 57-point result. After the decision to schiebe, the partner's heart decision is fine.
It is therefore questionable whether a schiebe call on a three-coloured[5] hand is really appropriate in this situation.
The human team, for example, plays undeufe and scores 107 points with its own hand containing 4 safe tricks and a little help from partner. In order to call undeufe, there should actually be 5 safe tricks in the hand, so it is debatable what is really the best decision but schiebe certainly isn't here. As in Situation 3, this can be a statistically better decision with a worse result.Nevertheless, the decision of the bot violates widespread Jass rules.

Situation 6:

With this hand both teams make undeufe trump, which is certainly the best choice. Both teams manage to make the 6 safe tricks and successfully pass at spades, as partner has the *"S6"* there.
But the difference is that the human team starts with the *"C6"* and partner knows that his *"C7"* and *"C8"* is bock[6] and keeps it. The bot plays all the diamond cards directly and then the *"C6"* in the 6th trick. The partner of the bot thereby discards the *"C7"* and *"C8"*.
The difference in points is that the human team makes a match and takes 157 points, while the bot team only makes 109 points.

In the previous section it was mentioned during the box-plot discussion, that the double rounds 7 and 8 are looked at more closely. This is done in the following.

Experiment 1, double round 7:

This is a difficult round. There is no obvious best variation. The human teams that call schiebe and make hearts trump do best. They score around 120 points. Other human teams make clubs forehand trump leaving an even game with 70-80 points.
The bot, on the other hand, only manages to score 50-60 points with its own trump. The agent always behaves in exactly the same way. First calls schiebe and then makes spades trump. This decision is

5 Three-coloured hand is a case, where a players hand has only 3 of the 4 existing colours. In many cases after a schiebe call, the partner will call trump on the colour that is missing.
6 Bock is called to cards, that are guaranteed winning a trick when played.

absolutely devastating for the course of the game, but we can show that it is not the trump selection but the subsequent card selection that is the problem.

The bot prefers to come on *"SJ"*, *"S8"*, *"S7"* instead of *"HA"*, *"HK"*, *"H9"*. The decision is understandable, because of a lot of bock cards and the trump jack in the hand. In fact, this trump decision could be converted into a better result. The trump selection decision was good.
But the bot then starts the round by playing the *"H6"*! This means that they don't come into the game and the opponent scores the first 10 points. In the second trick, the bot, who had called for a spades-trump decides to "stechen"[7] while only having 3 trumps to get into the game, which is not successful because the current hand of the bot is missing on the numerical trump advantage.
The bots now make the following two trump tricks with Jack and then Nell, but now the opponent still has a trump in his hand and can "stechen" the *"HA"* Bock and take control of the game.
The unwritten Jass rules state that you always open with the most valuable trump so that your partner can see what trump cards your opponents have[5]! This would be the *"S9"*. A scenario where the first bot plays the trump nell (*"S9"*) ends up with 77 points in the worst case and more often around 100 points.

This explains the big difference in favour of the human teams in double round 7. The bot team does not play an optimal game here, while the trump selection is appropriate.

Experiment 1st round 8:

Intended in this double round is that after calling schiebe the partner makes undeufe as trump. With coordination a solid 7-8 tricks game is feasible. Two human teams achieved a match in this round!
The bot team always plays the same pattern and scores many points on average. They make a mistake of throwing away the *"H6"* at the end. Nevertheless, most of the human teams do not play perfectly either and thus slightly worse than the bots. The human teams have often made the correct pattern with schiebe and undeufe trump. Some teams also did forehand trumps but this led to a bad result.

Depending on what player 1 plays out on the first move, more or less points can be scored in the end. The bot team does not play perfectly here, but better than the most teams that did not have a very good coordination going. Therefore, this double round swings to the bot team's advantage. It would be possible to score more points than the bot team but the challenging card distribution in this round has cost the most human teams many points on average.

---

7 Play a trump card on a non-trump trick in order to gain it guaranteed.

In the result discussions, we have seen various aspects that place the bot around a high-human level. It also becomes clear through the examples discussed why the bot does not manage to get to a clear high-human level or even above.

7 LIMITATIONS

In this chapter, closely related to chapters 5: Experimental Setup and 6: Experimental results, we want to provide an overview of the limitations and challenges that arise when trying to test a bot using human experiments for the Schieber variant of Jass.

An obvious difficulty in human experimentation is the recruitment of participants. There is the possibility of directly approaching people from the close environment, which proves to be effective. However, this method brings a limited number of participants to the experiment. Public announcements such as our Jass Challenge Event help to find people who are enthusiastic about the topic and catch some with attractive prizes.
Having a lot of experiment data is necessary because of situations mentioned in section 6.4 Result Discussion. It is possible that in a tournament where the participants not only want to win against the bot, but also fight against other participants to win a prize, the concentration is higher.
We leave open how the differences between in-person and remote experiments influenced the performance. In this case, the human players performance was high enough to win against the bot. It is important that the human performance would rise to a maximum when the bot performs better.

Increasing the number of experiments would increase the significance, but within the framework of the study, a time limit had to be set. The organisational effort to find time-slots when the two participants and a host are present is considerably high.
To generate more data, having a team of researchers would be beneficial. One option to generate large amounts of data would be to embed the bot on a platform such as Swisslos and thus receive big data in a short time.
However, this has the disadvantage that no information about the strength of the human players would be available.

Finally, there is a limitation in the significance of human vs. bot experiments in Schieber Jass.
As shown in section 6.4 Result Discussion with situation 3, small decisions can lead to considerable differences. Even though the agent may be statistically right, it potentially performs worse in the actual case. This is one reason why it would take a lot more experiments to address such scenarios correctly.
It would not be too bad if the games were not so close. In section 6.3 Result Comparison we saw that the difference in points is only

50.48% and 50.57% respectively. The fact that it is so evenly matched demands even more perfect significance, because otherwise it is difficult to make a clear statement about the bot level.

One problem in the nature of the Schieber game is that above a certain player level many moves are straightforward. In a lot of cases, it is clear what trump must be decided and what moves carry a big part of the game. Of course, important decisions are made later in the game in which a more experienced player or a perfectly playing bot can choose the statistically better one.
However, the effect of a decision has rarely a huge impact for the game, but provides the team with e.g. 14 points more. If a better player executes four such decisions better but loses 60 points in a trump decision due to a statistically 50/50 chance situation, this is not reflected in the point difference.

The random factor of the card distribution of this coordinately demanding game makes it even more difficult for the agent to be consistently better than a good human team. A good human team does not make any obvious mistakes and plays close to the optimum in terms of coordination.
The advantage of the bot in this game is probably the usually better-founded knowledge of the possible card distributions and the perfect memory of the played cards.
In situations that look like a 50/50 chance decision for many players, but one of the decisions prevails in the average outcome, the bot could be at an advantage in those cases. However, we have seen that the fine-tuned communication between the bots is sometimes lacking despite these advantages.

The accumulation of these factors lead to a great difficulty when experimenting in the area of human vs. bot with the Schieber card game.

8 CONCLUSION

After the introduction to the Swiss card game Jass and the motivation behind this thesis, a background on game theory and AI performance was given. Before the main part of the thesis is explained, we discuss the related work and especially the prior work, which is very closely related to this study.
The focus of the work are the human vs. AI experiments. Chapters 4: Implementation and 5: Experimental Setup are used to describe the approach in detail. These are decisive for the experimental results, which are discussed in chapter 6. With this work, we were able to find out that the AI does not yet perform optimally in the field of Schieber Jass and, depending on the argumentation, is just slightly below or at the high-human level.
The strong human teams were able to defeat the agent in the end. We provide explanations for this humbling result from an experimental and game-theoretical point of view.

Looking to the future, there is certainly potential for improvement in the agent, which would have to be addressed in order for the agent to have a chance of upgrading its performance. It might be worth considering features of the agents Liberatus and Big2AI to improve the Jass agent. If the coordination is optimised, the agent should make a very tough opponent for the best human teams.
The experiments with the double rounds that were carried out as part of this work seem to be very suitable. The tool of pre-set cards and the number of rounds played per game are debatable for future experiments.

Part IV

APPENDIX

A

Figure 9: The flyer from the Jass Challenge Event at the university of Bern. 10 Teams competed for the winner prizes. The event attracted 26 people over the whole duration.

B NAMING CONVENTIONS JSON

Naming conventions for the game state and REST-API calls:

Listing 5: Naming conventions.

```
DIAMONDS    = 0                  # Ecken / Schellen
D           = DIAMONDS
HEARTS      = 1                  # Herz / Rosen
H           = HEARTS
SPADES      = 2                  # Schaufeln / Schilten
S           = SPADES
CLUBS       = 3                  # Kreuz / Eichel
C           = CLUBS
OBE_ABE     = 4                  # Top-Down
O           = OBE_ABE
UNE_UFE     = 5                  # Bottom-Up
U           = UNE_UFE

A = Ace/Ass
K = King/Koenig
Q = Queen/Dame
J = Jack/Bauer
```

C

GAME FLOW UML

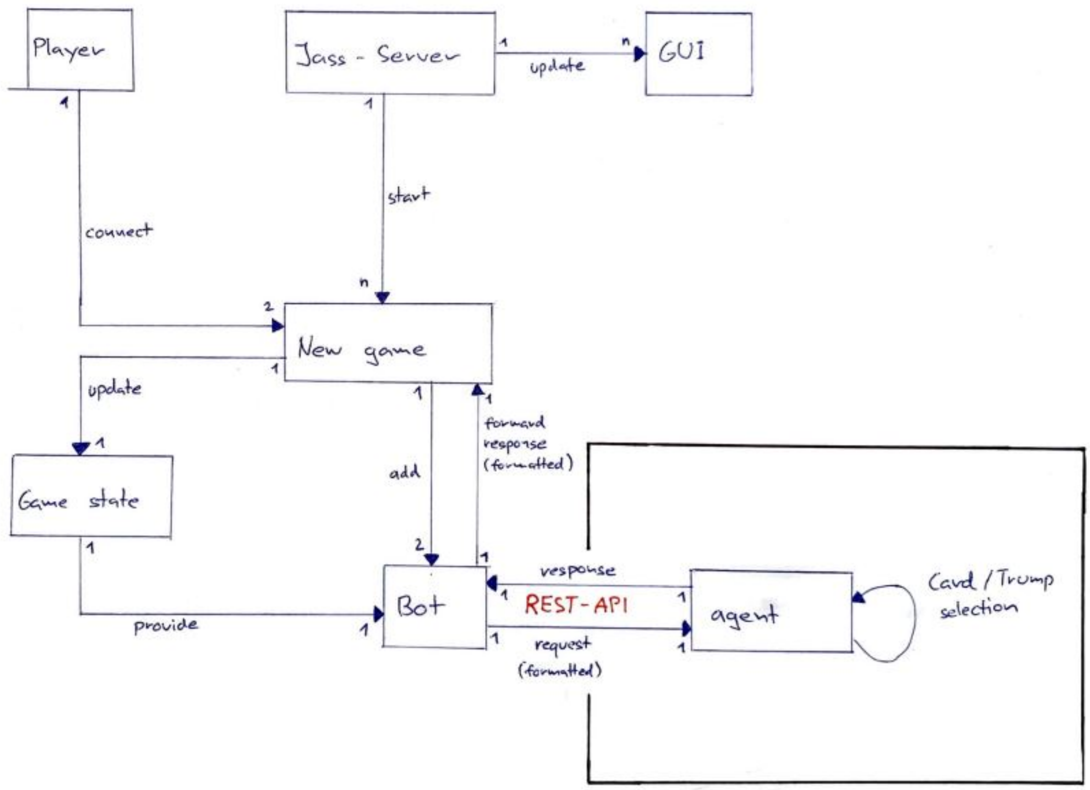


Figure 10: UML-like diagram for the game flow of the Jass server with the embedded bot connection.

D

EXAMPLE SITUATIONS

Listing 6: Situation 1: Game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
  "data": {
    "version": "V0.2",
    "trump": 0,
    "dealer": 0,
    "currentPlayer": 3,
    "playerView": 3,
    "forehand": 0,
    "tricks": [
      {
        "cards": [
          "DJ",
          "D6",
          "D10",
          "S6"
        ],
        "points": 30,
        "win": 3,
        "first": 3
      },
      {
        "cards": [
          "D8",
          "H6",
          "DA",
          "S8"
        ],
        "points": 11,
        "win": 1,
        "first": 3
      },
      {
        "cards": [
          "D9",
          "CQ",
          "DQ",
          "H7"
        ],
        "points": 20,
        "win": 1,
        "first": 1
```

```
},
{
  "cards": [
    "C6",
    "H9",
    "CJ",
    "CA"
  ],
  "points": 13,
  "win": 2,
  "first": 1
},
{
  "cards": [
    "C7",
    "C10",
    "S9",
    "C9"
  ],
  "points": 10,
  "win": 1,
  "first": 2
},
{
  "cards": [
    "CK",
    "SQ",
    "H8",
    "SJ"
  ],
  "points": 9,
  "win": 1,
  "first": 1
},
{
  "cards": [
    "C8",
    "S10",
    "SA",
    "HJ"
  ],
  "points": 23,
  "win": 1,
  "first": 1
},
{
  "cards": [
    "S7",
    "HQ",
    "SK",
    "H10"
  ],
```

```
          "points": 17,
          "win": 3,
          "first": 1
        },
        {
          "cards": [
            "DK",
            "HA",
            "D7",
            "HK"
          ],
          "points": 19,
          "first": 3
        }
      ],
      "player": [
        {
          "hand": []
        },
        {
          "hand": []
        },
        {
          "hand": []
        },
        {
          "hand": []
        }
      ],
      "jassTyp": "SCHIEBER",
      "gameId": 0
    }
  }
```


Listing 7: Situation 2: Starting hand.


```
"hand": [
  "HQ",
  "HJ",
  "C8",
  "H8",
  "SJ",
  "SA",
  "CA",
  "S6",
  "HA"
]
```


Listing 8: Situation 2: Game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
```

```
"data": {
  "version": "V0.2",
  "trump": 4,
  "dealer": 0,
  "currentPlayer": 0,
  "playerView": 0,
  "forehand": 1,
  "tricks": [
    {
      "cards": [
        "HA",
        "H6",
        "S10",
        "H7"
      ],
      "points": 21,
      "win": 3,
      "first": 3
    },
    {
      "cards": [
        "SA",
        "S8",
        "S7",
        "SQ"
      ],
      "points": 22,
      "win": 3,
      "first": 3
    },
    {
      "cards": [
        "HJ",
        "HK",
        "C7",
        "H9"
      ],
      "points": 6,
      "win": 2,
      "first": 3
    },
    {
      "cards": [
        "SK",
        "C9",
        "D8",
        "S6"
      ],
      "points": 12,
      "win": 2,
      "first": 2
    },
```

```
{
  "cards": [
    "DJ",
    "D6",
    "DA",
    "C8"
  ],
  "points": 21,
  "win": 0,
  "first": 2
},
{
  "cards": [
    "DK",
    "H8",
    "C10",
    "D9"
  ],
  "points": 22,
  "win": 0,
  "first": 0
},
{
  "cards": [
    "DQ",
    "SJ",
    "H10",
    "CJ"
  ],
  "points": 17,
  "win": 0,
  "first": 0
},
{
  "cards": [
    "D10",
    "HQ",
    "S9",
    "CQ"
  ],
  "points": 16,
  "win": 0,
  "first": 0
},
{
  "cards": [
    "D7",
    "CA",
    "C6",
    "CK"
  ],
  "points": 15,
```

```
        "first": 0
      }
    ],
    "player": [
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      }
    ],
    "jassTyp": "SCHIEBER",
    "gameId": 0
  }
}
```


Listing 9: Situation 3: Starting hand.


```
"hand": [
  "S10",
  "H10",
  "DK",
  "H7",
  "D7",
  "C9",
  "S8",
  "S7",
  "C6"
]
```


Listing 10: Situation 4: Human team game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
  "data": {
    "version": "V0.2",
    "trump": 5,
    "dealer": 3,
    "currentPlayer": 0,
    "playerView": 0,
    "forehand": 1,
    "tricks": [
      {
        "cards": [
          "H6",
          "SQ",
```

```
      "H10",
      "HA"
    ],
    "points": 24,
    "win": 2,
    "first": 2
  },
  {
    "cards": [
      "S6",
      "S10",
      "DK",
      "SA"
    ],
    "points": 25,
    "win": 2,
    "first": 2
  },
  {
    "cards": [
      "S7",
      "DJ",
      "CQ",
      "SJ"
    ],
    "points": 7,
    "win": 2,
    "first": 2
  },
  {
    "cards": [
      "S8",
      "C7",
      "DQ",
      "SK"
    ],
    "points": 15,
    "win": 2,
    "first": 2
  },
  {
    "cards": [
      "S9",
      "D7",
      "CJ",
      "DA"
    ],
    "points": 2,
    "win": 2,
    "first": 2
  },
  {
```

```
      "cards": [
        "HJ",
        "D10",
        "H7",
        "H9"
      ],
      "points": 12,
      "win": 0,
      "first": 2
    },
    {
      "cards": [
        "H8",
        "HQ",
        "CK",
        "D8"
      ],
      "points": 23,
      "win": 0,
      "first": 0
    },
    {
      "cards": [
        "HK",
        "C10",
        "CA",
        "C8"
      ],
      "points": 22,
      "win": 0,
      "first": 0
    },
    {
      "cards": [
        "C9",
        "C6",
        "D9",
        "D6"
      ],
      "points": 22,
      "first": 0
    }
  ],
  "player": [
    {
      "hand": []
    },
    {
      "hand": []
    },
    {
      "hand": []
```

```
        },
        {
          "hand": []
        }
      ],
      "jassTyp": "SCHIEBER",
      "gameId": 0
    }
}
```


Listing 11: Situation 4: Bot team game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
  "data": {
    "version": "V0.2",
    "trump": 5,
    "dealer": 2,
    "currentPlayer": 0,
    "playerView": 0,
    "forehand": 1,
    "tricks": [
      {
        "cards": [
          "S6",
          "SQ",
          "H10",
          "SA"
        ],
        "points": 24,
        "win": 1,
        "first": 1
      },
      {
        "cards": [
          "S8",
          "S10",
          "HK",
          "SJ"
        ],
        "points": 24,
        "win": 1,
        "first": 1
      },
      {
        "cards": [
          "S7",
          "DJ",
          "DK",
          "SK"
        ],
        "points": 10,
```

```
    "win": 1,
    "first": 1
  },
  {
    "cards": [
      "S9",
      "C7",
      "DQ",
      "DA"
    ],
    "points": 3,
    "win": 1,
    "first": 1
  },
  {
    "cards": [
      "H6",
      "D7",
      "H8",
      "HA"
    ],
    "points": 19,
    "win": 1,
    "first": 1
  },
  {
    "cards": [
      "D9",
      "D8",
      "CJ",
      "C10"
    ],
    "points": 20,
    "win": 0,
    "first": 1
  },
  {
    "cards": [
      "D6",
      "C9",
      "HQ",
      "CA"
    ],
    "points": 14,
    "win": 0,
    "first": 0
  },
  {
    "cards": [
      "D10",
      "CQ",
      "H9",
```

```
          "HJ"
        ],
        "points": 15,
        "win": 0,
        "first": 0
      },
      {
        "cards": [
          "C8",
          "H7",
          "C6",
          "CK"
        ],
        "points": 23,
        "first": 0
      }
    ],
    "player": [
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      }
    ],
    "jassTyp": "SCHIEBER",
    "gameId": 0
  }
}
```


Listing 12: Situation 5: Starting hand.


```
"hand": [
  "SJ",
  "SQ",
  "D7",
  "C7",
  "D8",
  "SK",
  "D6",
  "C9",
  "DJ"
]
```


Listing 13: Situation 6: Starting hand Player.

```
"hand": [
  "D6",
  "D7",
  "SK",
  "D8",
  "SQ",
  "S8",
  "C6",
  "DK",
  "D10"
]
```

Listing 14: Situation 6: Starting hand Partner.

```
"hand": [
  "S6",
  "CJ",
  "S10",
  "H9",
  "C7",
  "C8",
  "H7",
  "HA",
  "DQ"
]
```

Listing 15: Experiment 1. Round 8: Starting hand player.

```
"hand": [
  "SQ",
  "C10",
  "DK",
  "CJ",
  "D9",
  "C7",
  "H6",
  "S9",
  "SA"
]
```

Listing 16: Experiment 1. Round 8: Starting hand partner.

```
"hand": [
  "DA",
  "D8",
  "SK",
  "S10",
  "S8",
  "S7",
```

```
  "S6",
  "CK",
  "C6"
]
```


Listing 17: Experiment 1. Round 8: Game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
  "data": {
    "version": "V0.2",
    "trump": 5,
    "dealer": 2,
    "currentPlayer": 2,
    "playerView": 2,
    "forehand": 0,
    "tricks": [
      {
        "cards": [
          "C7",
          "CA",
          "CK",
          "C9"
        ],
        "points": 4,
        "win": 1,
        "first": 1
      },
      {
        "cards": [
          "CJ",
          "CQ",
          "C6",
          "HK"
        ],
        "points": 20,
        "win": 3,
        "first": 1
      },
      {
        "cards": [
          "S6",
          "SJ",
          "SQ",
          "HA"
        ],
        "points": 16,
        "win": 3,
        "first": 3
      },
      {
```

```
  "cards": [
    "S8",
    "DQ",
    "SA",
    "C8"
  ],
  "points": 19,
  "win": 3,
  "first": 3
},
{
  "cards": [
    "S7",
    "DJ",
    "S9",
    "HJ"
  ],
  "points": 4,
  "win": 3,
  "first": 3
},
{
  "cards": [
    "S10",
    "HQ",
    "DK",
    "D10"
  ],
  "points": 27,
  "win": 3,
  "first": 3
},
{
  "cards": [
    "SK",
    "H10",
    "D9",
    "H8"
  ],
  "points": 22,
  "win": 3,
  "first": 3
},
{
  "cards": [
    "DA",
    "D6",
    "C10",
    "D7"
  ],
  "points": 21,
  "win": 2,
```

```
        "first": 3
      },
      {
        "cards": [
          "H9",
          "H6",
          "H7",
          "D8"
        ],
        "points": 19,
        "first": 2
      }
    ],
    "player": [
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      },
      {
        "hand": []
      }
    ],
    "jassTyp": "SCHIEBER",
    "gameId": 0
  }
}
```


Listing 18: Experiment 1. Round 7: Starting hand player.


```
"hand": [
  "SK",
  "S9",
  "D9",
  "HJ",
  "HQ",
  "H6",
  "CJ",
  "C9",
  "C7"
]
```


Listing 19: Experiment 1. Round 7: Starting hand partner.


```
"hand": [
  "SJ",
  "S8",
```

```
  "S7",
  "HA",
  "HK",
  "CK",
  "D10",
  "D6",
  "H9"
]
```


Listing 20: Experiment 1. Round 7: Game progression.


```
{
  "broadcast": true,
  "messageType": "BROADCAST_GAMESTATE",
  "data": {
    "version": "V0.2",
    "trump": 2,
    "dealer": 2,
    "currentPlayer": 0,
    "playerView": 0,
    "forehand": 0,
    "tricks": [
      {
        "cards": [
          "H6",
          "H7",
          "H9",
          "H10"
        ],
        "points": 10,
        "win": 2,
        "first": 1
      },
      {
        "cards": [
          "D7",
          "D9",
          "DK",
          "S7"
        ],
        "points": 4,
        "win": 3,
        "first": 2
      },
      {
        "cards": [
          "SJ",
          "S6",
          "SK",
          "SA"
        ],
        "points": 35,
```

```
    "win": 3,
    "first": 3
  },
  {
    "cards": [
      "S8",
      "SQ",
      "S9",
      "H8"
    ],
    "points": 17,
    "win": 1,
    "first": 3
  },
  {
    "cards": [
      "C7",
      "C8",
      "CK",
      "S10"
    ],
    "points": 14,
    "win": 2,
    "first": 1
  },
  {
    "cards": [
      "CQ",
      "C9",
      "C6",
      "D6"
    ],
    "points": 3,
    "win": 2,
    "first": 2
  },
  {
    "cards": [
      "C10",
      "CJ",
      "CA",
      "D10"
    ],
    "points": 33,
    "win": 0,
    "first": 2
  },
  {
    "cards": [
      "DA",
      "HK",
      "DJ",
```

```
        "HJ"
      ],
      "points": 19,
      "win": 0,
      "first": 0
    },
    {
      "cards": [
        "D8",
        "HA",
        "DQ",
        "HQ"
      ],
      "points": 17,
      "first": 0
    }
  ],
  "player": [
    {
      "hand": []
    },
    {
      "hand": []
    },
    {
      "hand": []
    },
    {
      "hand": []
    }
  ],
  "jassTyp": "SCHIEBER",
  "gameId": 0
 }
}
```

BIBLIOGRAPHY

[1] Noam Brown and Tuomas Sandholm. “Superhuman AI for heads-up no-limit poker: Libratus beats top professionals.” In: *Science* 359 (2018), pp. 418 –424.

[2] Lien-Wu Chen and Yiou-Rwong Lu. “Challenging Artificial Intelligence With Multiopponent and Multimovement Prediction for the Card Game Big2.” In: *IEEE Access* 10 (2022), pp. 40661–40676.

[3] Stefan Edelkamp. “Challenging human supremacy in Skat.” In: *Twelfth Annual Symposium on Combinatorial Search*. 2019.

[4] Sebastian Kupferschmid and Malte Helmert. “A Skat player based on Monte-Carlo simulation.” In: *International Conference on Computers and Games*. Springer. 2006, pp. 135–147.

[5] Ernst Marti. *Schweizer Jass-Führer*. Kümmerly+Frey, 1995.

[6] Joel Niklaus, Michele Alberti, Rolf Ingold, Markus Stolze, and Thomas Koller. “Challenging Human Supremacy: Evaluating Monte Carlo Tree Search and Deep Learning for the Trick Taking Card Game Jass.” In: *ICAISC*. 2020.

[7] Ho Chit Siu, Jaime Peña, Edenna Chen, Yutai Zhou, Victor Lopez, Kyle Palko, Kimberlee Chang, and Ross Allen. “Evaluation of Human-AI Teams for Learned and Rule-Based Agents in Hanabi.” In: *Advances in Neural Information Processing Systems* 34 (2021).

Erklärung

gemäss Art. 30 RSL Phil.-nat.18

Name/Vorname:	Martinez Roman
Matrikelnummer:	19-104-355
Studiengang:	Computer Science

Bachelor ☑ Master ☐ Dissertation ☐

Titel der Arbeit:	Can the Best Jass AIs Beat the Top Humans?
LeiterIn der Arbeit:	PD Dr.Matthias Stürmer Assistant Joel Niklaus

Ich erkläre hiermit, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäss aus Quellen entnommen wurden, habe ich als solche gekennzeichnet. Mir ist bekannt, dass andernfalls der Senat gemäss Artikel 36 Absatz 1 Buchstabe r des Gesetzes vom 5. September 1996 über die Universität zum Entzug des auf Grund dieser Arbeit verliehenen Titels berechtigt ist.
Für die Zwecke der Begutachtung und der Überprüfung der Einhaltung der Selbständigkeitserklärung bzw. der Reglemente betreffend Plagiate erteile ich der Universität Bern das Recht, die dazu erforderlichen Personendaten zu bearbeiten und Nutzungshandlungen vorzunehmen, insbesondere die schriftliche Arbeit zu vervielfältigen und dauerhaft in einer Datenbank zu speichern sowie diese zur Überprüfung von Arbeiten Dritter zu verwenden oder hierzu zur Verfügung zu stellen.

Ort/Datum

Unterseen, 19.12.2022

Unterschrift

R. Martinez