SPADENet: Skill-based Player Action Decision and Evaluation for Card Games using Deep Neural Networks (Online Rummy as Case Study)

Divyansh Jain, Anurag Garg
Data Science Group, Gameskraft
research@gameskraft.com
India

***Abstract*—In Online multiplayer skill-based card games, players compete with each other intending to optimize their reward. They have to make many decisions quickly as the game progresses and new information is revealed. Representing and comprehending such intricate information, and then making an optimum decision in real-time, is an extremely challenging task that requires skill. We introduce SPADENet, a generic and improved multi-input deep neural network architecture designed to 1) comprehensively capture and combine the players' hand along with the ever-evolving visible game state outside of their hand 2) seamlessly incorporate the intricacies of different variants of any given card game 3) determine the best decision for any skill-based card games. Considering Online Rummy as an example, we demonstrate how this architecture can be easily adapted for determining the optimum drop (or play) decision across different Online Rummy variants (such as Points, Pool, Deals, etc.) and at different game stages. We observed improvement in Test F1 score across all Rummy variants with the highest being for Pool-6P from 0.767 (existing best-performing method) to 0.908 (18% increase). The odds of favorable game outcomes also increased when players follow model recommendations. We also show how comparable performance can be achieved by applying transfer learning across different variants. We further conduct a quantitative analysis of game state features outside of players' hands to comprehend their significance in influencing the decision to Drop or Play. With this architecture, player skill can be measured and Low Skilled Players can be identified for any skill-based card game. We can take appropriate measures to improve their gameplay and provide a personalized upskilling path.**

***Index Terms*—Multi-Input Deep Neural Networks, Convolutional Neural Networks, Skill Assessment, Personalized Gameplay improvement, Card Game Analytics**

I. Introduction

Online card games have gained enormous popularity due to increased digital penetration. The total revenue generated by them is expected to reach US$ 18.61 billion by 2027 with a projected annual growth rate of 13.54% [1]. These games allow players to compete against each other for a potential reward. Most of these games are skill-based and require them to make strategic decisions based on a multitude of factors.

Most card games such as Poker, Bridge, and Rummy are examples of imperfect information games [2] as each player has partial information about the game. They don't fully know the cards held by other players, however, they can develop intuition about them as the game progresses. Since each player learns about the changing "game state" simultaneously, their playing strategy and decisions also evolve in real-time. Skilled players, therefore, not only depend upon their past game learnings, but they are also able to make decisions based on how their opponents are playing. For developing a method to correctly evaluate player decisions, it becomes essential to include these factors. Representing and holistically assimilating such an evolving game state in a generic framework and then coming up with the best action is not only difficult for players but is also a challenging Machine Learning problem.

A. Main Contribution

With the above problems and observations in mind, we propose SPADENet. The key contributions of this research work can be classified into two categories:

- **Game Table Progression and Evolution:** It allows to comprehensively capture and combine the player's hand along with the ever-evolving game state representation that exists outside of their hand. Only the features that are visible to the player are considered as part of this "game state" representation.
- **Generic Game Model Architecture:** A multi-input DNN architecture that can be adapted to i) any skill-based card games and ii) different variants of the game itself.

To evaluate the architecture, we take Online Rummy as a case study and evaluate the first drop (or play) decision across its different variants (Points, Pool, Deals, etc.) and player count format (2-player, 6-player, etc). The main contribution here is in bringing a better understanding of how the Drop (or Play) decision of the player is influenced by:

- the Drop (or Play) decision of opponents on the table.
- the number of drops left with the player and the opponents.
- the card which is present on top of the Open Deck pile.
- the change in the Rummy variant.

This architecture can be extended to other skill-based card games with appropriate features. To the best of our knowledge, no prior work has been done regarding this.

979-8-3503-5409-6/24/$31.00 ©2024 IEEE
DOI 10.1109/CAI59869.2024.00233

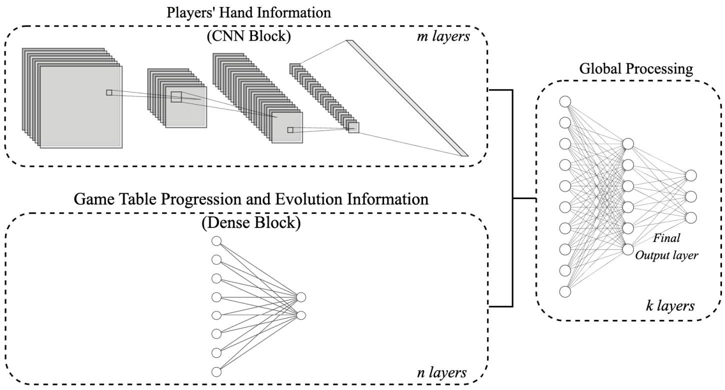

Fig. 1: SPADENet Architecture

II. Related Work

There has been growing research in understanding and modeling player behavior and building models that can learn and beat humans, in the context of multi-player games. Reinforcement learning-based methods have been used widely to develop agents for games that train a network through self-play and decide the next action using a search algorithm. For Go, which is a perfect information game, AlphaGo [3] became the first algorithm to beat a professional Go player. They employed the Monte Carlo Tree Search algorithm (MCTS) on a model that was trained using a combination of Supervised Learning on expert human moves and Reinforcement Learning through self-playing respectively. Later, the algorithm was improved in the form of AlphaGoZero [4] which completely relied on reinforcement learning and was trained entirely using self-play, skipping the need for reliance on training data. By contrast, in imperfect-information games like Poker, not all information is visible to all players which makes the game complex. Algorithms such as Libratus [5], Pluribus [6, 7] and ReBel [8] have outperformed humans in Poker. For Hearts, [9] developed an agent based on Stochastic Linear Regression and Temporal Difference Learning which outperformed human players. For Gin Rummy, [10] compared the performance of agents trained to play using 2 methods namely- Temporal Difference Learning and Co-Evolution, with the latter found to be performing better. Ref. [11] found their proposed heuristic model to perform better than an ANN model for informing draw, discard, and knock decisions in Gin Rummy. Ref. [12] developed a fast hand strength estimation algorithm and proposed a gin rummy agent by using the algorithm in conjunction with Counterfactual Regret minimization. *While the above works aim to develop AI agents that can play and achieve or surpass human-level performance, we are more interested in understanding how highly skilled players play. This will enable us to understand their playing behavior as well as develop a learning journey for upskilling other players when they play against human opponents.* CNNs have found applications in various tasks such as Image detection, classification, and segmentation across multiple domains [13]–[15].

For Online Rummy, [16] has proposed a Hand Quality estimation model (HandNet) based on CNN, which was trained on expert human decisions. They propose both - a method to create an image representation of player hand cards and a CNN model architecture to predict optimum drop or play decisions which has been trained using top skilled human player actions.

In upcoming sections, we show that this method lacks holistic game context and opponent information that is visible to the player. These are key factors in imperfect information games like Rummy. **Their work only considers player's hand in making that decision for Points Rummy variant and hence, doesn't generalize well on other Rummy variants like Pool and Deal, etc.** Drop or play decision is a key indicator in determining player skill in Online Rummy and its accurate determination can help classify their playing behavior.

III. SPADENet Architecture

We introduce a multi-input single-output Deep Neural Network, SPADENet. It is a combination of 2 separate neural networks (Fig. 1):

- Players' Hand Information (CNN Block): The first network is a CNN model that takes image representation of players' cards. It learns to identify only those card patterns or melds that are relevant to the problem. These patterns can also be common across the different variants for certain games. In such cases, transfer learning can be used for this block when training for another game variant.
- Game Table Progression and Evolution (Dense Block): The second network is a feed-forward neural network that captures game-variant-specific features. These features help us capture the nuances that, otherwise, don't get captured in the first network. Only the features that are visible to the player are included here.

Fig. 2: Game-table for Online Rummy with a *Valid Declaration* hand

These networks' final output is concatenated and passed through a Dense layer which works as a Global Processing Unit. Multi-input CNNs allow a combination of different data sources and process them separately using different network layers. They have proven to be more accurate in a variety of domains [17]–[19]. Combining the outputs of different network streams in multi-input CNNs towards the end has been shown to give better results [20].

IV. Game Of Rummy and its different variants

Rummy requires a combination of skill and strategic acumen. It generally employs 2 standard decks, each comprising 53 cards (52 cards + 1 printed joker), and is typically played between 2-6 players. The primary objective of the player is to arrange the 13-cards in their hand, by building melds - sequences and sets, leading to a valid declaration before other opponents. A Sequence is defined as a group of cards with an identical suit and sequential ranks (e.g. - $A\heartsuit, K\heartsuit, Q\heartsuit, J\heartsuit$), while a set consists of cards having the same rank but different suits (e.g. - $2\clubsuit, 2\diamondsuit, 2\spadesuit$) (Fig. 2). Each meld must have at least 3 cards. At the start of each game, a card is randomly drawn and designated as a wild joker, capable of substituting for a missing card within a meld. A sequence formed with the help of a joker (either printed or wild) is called an Impure Sequence (e.g. - $Q\spadesuit, 8\spadesuit, 9\spadesuit$, where $Q\spadesuit$ is the wild joker), else it is called a Pure Sequence. To make a valid declaration, the player must have a minimum of two sequences, with one of them being a pure sequence. The remaining cards must be organized into either sequences or sets.

The gameplay unfolds with players drawing cards from either the closed deck or the top card of the discarded pile, known as the Open Deck card. After drawing a card from one of the decks, players must strategically discard one from their hand, adding it to the discard pile and thereby determining the Open Deck Card for the next player. This sequence persists until a player makes a valid declaration, compelling all participants to strategically minimize their points.

Importance of first drop: Players can strategically Drop out of the game if they think the odds of a favorable outcome (either win or lose by less than Drop penalty points) for their hand are limited. If dropped, they forfeit the game by 15/20/25 points (depending on the Rummy variant) which minimizes their losses.

A. Points Rummy

A Points game consists of a single deal with predetermined monetary value. The player who successfully makes a valid declaration first, wins collective sum of points of all other players, while other players lose points based on the sum of their ungrouped cards. Therefore, the earlier a player secures victory and the wider the margin of their success, the greater the winnings, emphasizing the strategic importance of early and decisive wins.

B. Pool Rummy

In Pool Rummy, participants contribute a fixed entry fee to form the game's prize pool collectively. At the beginning of the game (first round), all players have a game score of 0 and they aim to minimize points and avoid reaching the elimination threshold (such as 61/101/201), until only one player remains. In each round, the winner accrues 0 points, while the other players accumulate points based on the ungrouped cards they hold. For a Pool 101 game, which has a drop penalty of 20 points, they can drop at max 5 times[1]. This variant places a premium on strategic play to achieve the dual goal of scoring minimal points and outlasting opponents. The first drop decision becomes even more crucial as it is also affected by the outcome of the previous deal and even a single bad play can lead to massive loss and elimination.

C. Deals Rummy

In the Deals game, players play a predetermined number of deals with a fixed entry fee, which forms the game's prize pool. Each player starts with the same pot. After each round, points are calculated for each player based on the sum of ungrouped cards. For the losing player(s), these points are reduced from their initial pot and added to the pot of the winning player (who declares first). At the end of all deals, the player with the highest pot wins and takes the entire prize pool as winnings.

V. Methodology

We define a High Skilled Player (hereby referred to as *HSP*) as someone who has played at least 100 cash games on high-value tables[2] and has a differential end score ≥ 3 and ≥ 20% win rate (following [16]). Analysis of their behavior for the games played showed that their first drop decision not only depends upon their dealt hand and Open Deck Card but also on other factors. For example - their tendency to drop is higher in 6-player games as compared to 2-player games. HSPs also prefer to drop more in the Pool game variant vs other games (Table I). The tendency to drop also increases as more players join and choose not to drop out of the game (Table II). For Pool games, which can run across multiple deals, HSPs tend to drop less if the opponents have more drops remaining with

[1] Drops left = floor[(Game End Score-Player's Current Game Score)/Drop Penalty]

[2] Minimum Table Buy-in Amount ≥ Rs. 500

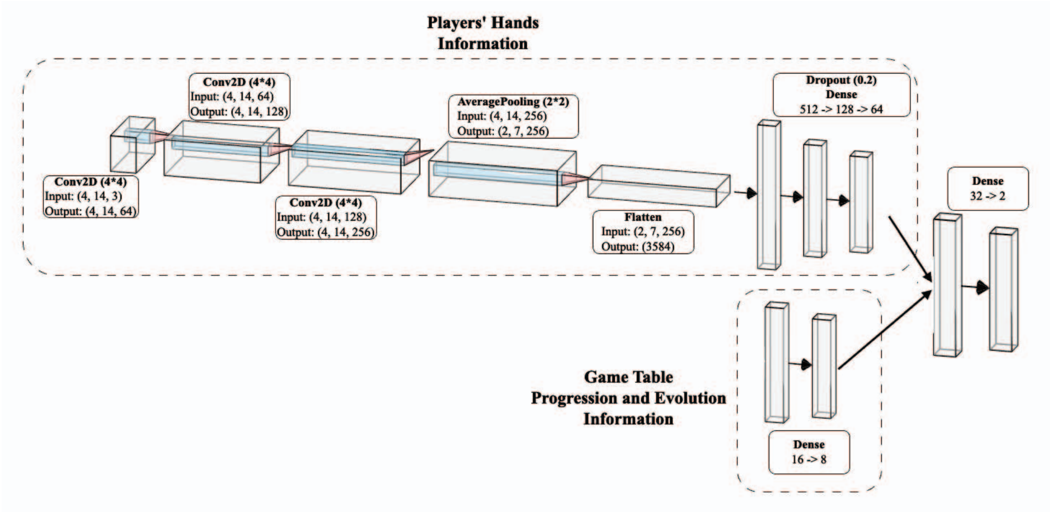

Fig. 3: SPADENet Architecture for Online Rummy

them (which is the result of their previous wins) (Table III). Previous work done till now hasn't accounted for these game variant-specific nuances. To accurately predict the best play or drop action, we need to include these attributes as well.

Turn Number	2-Player		6-Player		
	Points	Pool	Points	Pool	Deals
1	0.13	0.19	0.39	0.37	0.39
2	0.30	0.39	0.44	0.42	0.44
3	-	-	0.46	0.46	0.45
4	-	-	0.48	0.49	0.44
5	-	-	0.49	0.53	0.47
6	-	-	0.50	0.56	0.46
Overall	**0.21**	**0.28**	**0.46**	**0.47**	**0.44**

TABLE I: Avg. Drop Rates of HSPs across different Rummy variants and player counts.

Players Joined	Players Still Playing				
	1	2	3	4	5
2	0.22	-	-	-	-
3	0.22	0.33	-	-	-
4	0.26	0.37	0.37	-	-
5	0.28	0.41	0.46	0.45	-
6	0.32	0.48	0.54	0.54	0.52

TABLE II: Avg. Drop Rates of HSPs for Pool-6P games for different player count on the table.

Such a problem can be addressed by multi-input single-output Deep Neural Network. The upcoming section explains how we adapted SPADENet for different variants of Online Rummy.

VI. SPADENet for Online Rummy And its popular variants

We take Online Rummy as a case study and evaluate the first drop (or play) decision for its different variants (Points,

Drops Left (Player)	Drops Left (Opponent)					
	0	1	2	3	4	5
0	0	0	0	0	0.01	0.02
1	0.33	0.27	0.20	0.17	0.14	0.12
2	0.44	0.37	0.32	0.26	0.23	0.22
3	0.46	0.41	0.35	0.32	0.28	0.27
4	0.43	0.39	0.36	0.34	0.31	0.28
5	0.43	0.39	0.38	0.34	0.34	0.31

TABLE III: Avg. Drop Rates of HSPs for Pool-2P (101) games for player vs opponent drops left.

Pool, Deals[3], etc.) and player count format. We model this as a supervised binary classification problem since we aim to mimic the behavior of High Skilled Players. Fig. 3 shows the adapted version of SPADENet for Online Rummy.

- Players' Hands Information: The first network is a CNN model that takes image representation of players' cards as described in [16]. For the first drop decision, players only need to consider their dealt hand and since it is taken at the start of game, picked/discarded card history is not involved. This rules out need for sequence based models such as RNNs or transformers. CNNs are able to learn and capture the card relationships from the hand image representation. In previous work, only the player's hand (13 Cards) was embedded in the hand image representation. However, the Open Deck Card can also influence player Drop (or Play) behavior. So to make the model more realistic, we included Open Deck Card as well in the hand image representation.
- Game Table Progression and Evolution: The second network is a 2-layer feedforward neural network that captures Rummy-variants specific features. It also allows for

[3] First Drop is only allowed in Deals 6p (3 Deal) game.

Rummy Variant	Feature Name	Feature Description
Points-2P	Turn No.	- Player turn no (1 or 2).
Points-6P	Players joined Players playing Turn No.	- Count of players who joined the game table (2-6). - Count of other players who are still on the game table when the current player's turn arrives (1-5). - Player turn no (1-6).
Pool-2P	Player drops left Opponent drops left Turn No.	- No. of drops left with the player[1]. - No. of drops left with opponent[1]. - Player turn no (1 or 2).
Pool-6P	Players joined Players playing Player drops left Opponents' drops left Turn No.	- Count of players who joined the game table (2-6). - Count of other players who are still on the game table when the current player's turn arrives (1-5). - No. of drops left with the player[1]. - Drops left at game level when the current player's turn arrives (Min., Max., Avg., Median). - Player turn no (1-6).
Deals-6P (3 Deals)	Players joined Players playing Deal number Player rank Turn No.	- Count of players who joined the game table (2-6). - Count of other players who are still on the game table when the current player's turn arrives (1-5). - Current deal number (1-3) - Rank of the player based on their points (in decreasing order) at the start of deal (1-6) - Player turn no (1-6).

TABLE IV: SPADENet Features (Online Rummy as a Case Study)

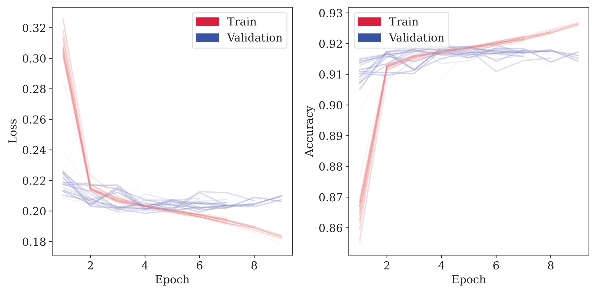

Fig. 4: Loss, Accuracy vs Epochs: Pool 6P

features that are outside of players' hands and represent the evolving game state. These features are described in Table IV.

These networks' output is concatenated to form a dense layer with 32 neurons. The final output layer is a Dense layer with 2 neurons and the softmax activation function. **Across the entire network, only the features that are visible to the player are considered.**

A. *Training and Dataset*

For preparing the training dataset, we considered the Rummy games played by the selected HSPs. This led to approximately 2,000 players, corresponding to over 580,000 deals for each game variant. 20% of these players are separately allocated to each of the validation and test sets respectively. This is done to avoid any potential bias and ensure that the drop or play behavior learned by the model on the players in the training set also generalizes well with the drop or play behavior of unseen players in the validation and test set. The training dataset comprises their initial hand, Open Deck Card along with other variant-specific attributes (from Table IV) that were visible to them, representing game states as input for these players, with their corresponding first drop (1) or play (0) action as the target attribute. We also identified HSPs using the same definition over different time periods and found the target distribution to be similar, confirming correctness of our sampling method. The model is trained using the categorical cross-entropy loss function and optimized with the Adam optimizer [21]. Training occurs over 50 epochs with a batch size of 512. To prevent overfitting, Early Stopping is implemented [22], halting training when the validation loss ceases to decrease for more than 3 epochs (Fig. 4 shows learning curves for model trained with multiple random states to avoid initialization issues). A separate model was trained for each game variant.

VII. RESULTS

To assess performance, we conducted a comprehensive evaluation of SPADE-Net, comparing its performance metrics against alternative models using the same test set. Since to the best of our knowledge, there are no public dataset and benchmarks available to compare our results with, we established a foundational Rule Model as a *baseline*, as well as also compared our results with previous work (HandNet [16]) by replicating that on our own dataset. The Rule model recommends dropping if a player doesn't possess at least 1 Pure Sequence with a free Joker (not part of the Pure Sequence) or has less than 2 Joker cards, else recommending to play. This rule emerged from our examination of actions taken by HSPs, where their action is ≥90% toward either playing or dropping.

SPADENet demonstrated superior performance over other models. We looked into how the model's behavior changes as well as top factors towards the prediction. Finally, we looked into the correlation between model predictions and game outcomes, providing valuable insights into the predictive efficacy of SPADENet. These are discussed in the following subsections.

Model Evaluation: We compare the results of our model for all Rummy variants with a domain-based Rule model

Model	AUC-ROC		AUC-PRC		F1 (P=R)	
	2P	6P	2P	6P	2P	6P
Rule	-	-	-	-	0.489	0.81
GTPE	0.633	0.601	0.527	0.544	0.313	0.529
HandNet	0.92	0.93	0.7	0.88	0.68	0.80
HandNet**	0.926	0.938	0.749	0.912	0.719	0.849
SPADENet	**0.963**	**0.962**	**0.878**	**0.948**	**0.79**	**0.89**

TABLE V(a): Rummy Variant: Points

Model	AUC-ROC		AUC-PRC		F1 (P=R)	
	2P	6P	2P	6P	2P	6P
Rule	-	-	-	-	0.59	0.795
HandNet	-	-	-	-	-	-
GTPE	0.725	0.718	0.45	0.635	0.472	0.615
HandNet**	0.881	0.874	0.689	0.803	0.661	0.767
SPADENet	**0.966**	**0.974**	**0.92**	**0.967**	**0.838**	**0.908**
SPADENet^{TL}	**0.933**	**0.942**	**0.89**	**0.954**	**0.817**	**0.892**

TABLE V(b): Rummy Variant: Pool

Model	AUC-ROC	AUC-PRC	F1 (P=R)
	6P	6P	6P
Rule	-	-	0.702
GTPE	0.543	0.473	0.477
HandNet	-	-	-
HandNet**	0.867	0.782	0.753
SPADENet	**0.896**	**0.816**	**0.795**

TABLE V(c): Rummy Variant: Deals (3 Deals)

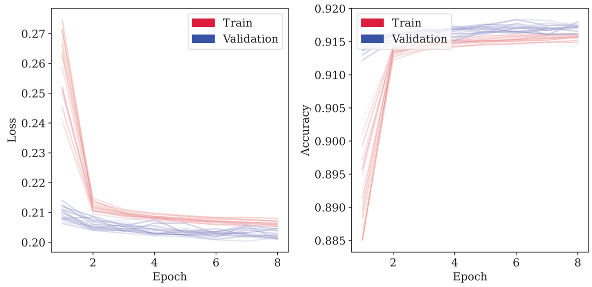

Fig. 5: Loss, Accuracy vs Epochs: Pool 6P Transfer Learning

(*baseline*) and HandNet (as reported in [16] for points 2p & 6p). For a fair comparison across all Rummy variants and to remove any resulting bias that may arise due to the underlying training data, we also trained and evaluated HandNet on the dataset on which SPADENet was trained (referred to as HandNet**). To understand the impact of Game Table Progression and Evolution alone, we trained and evaluated a model by only considering features as described in Table (IV) and excluding player hand information (referred to as GTPE). Table (V(a), V(b) and V(c)) show the results of the model in terms of metrics like AUC-ROC, AUC-PRC, and F1 score (we use the break-even point where precision = recall to decide the threshold). The HandNet** model performance is similar to the *baseline* Rule Model for Pool-6P. We observed improvement in Test F1 score for SPADENet across all Rummy variants with the highest being for Pool-6P from 0.767 (existing best-performing method HandNet**) to 0.908 (18% increase).This demonstrates the need for the inclusion of other non-player-hand features in SPADENet.

Transfer Learning: The CNN block learns the relationship between the player's hands and the Open Deck Card with the play or drop decision. Across different Rummy variants, while the game table information changes in the DENSE block (Fig. 3), the player's hand-learned features remains same. Hence, the same trained CNN block from one variant can be reused in the other variants. To validate this, we retrained another model for the Pool-6P variant by taking the CNN block from the Points-6P game. We froze the weights of the network layers for the CNN block during training while freely allowing the DENSE block to learn (referred to as SPADENet^{TL}). This method not only gave similar results (Table V(b)), but also achieved convergence faster (Fig. 4 vs Fig. 5).

Impact of Open Deck Card: We examined the influence of the Open Deck Card on both drop probability and player decision-making. To investigate this, we selected a hand at random {3♥, 6♠, 1♦, 3♠, 5♥, 10♣, 13♥, 6♦, 7♣, 1♦, 12♠, 7♦, 12♠} with the joker being 10♥ and the Open Deck Card being 4♥. If we disregard the open deck card, the model's drop probability is 0.64[4], indicating it is not playable. However, in an actual game, the player acquires the Open Deck Card and refrains from dropping, as it forms a pure sequence (3♥, 4♥, 5♥), significantly enhancing the hand and reducing hand points to 66 from 80. On incorporating the Open Deck Card in the analysis, the model's drop probability also diminished to 0.03. This confirms the players' consideration of the Open Deck Card in their strategic decisions regarding playing or dropping.

Impact of Game Variant: Players' drop behavior also changes depending upon the game variant (Table I). To understand this, we picked a random player hand { 12♠, 3♥, 12♠, 13♥, 1♦, 5♥, 11♣, 3♠, 6♦, 7♣, 1♦, 6♠, 7♦}. With joker = 10♥ and Open Deck Card = 4♥, we looked at how drop probabilities changed across game variants. We found that for the same hand, the model drop probability was 0.68 for the Points 2P game but it increased to 0.90[5] for the Pool 2P game. This is expected since players tend to take less risk in Pool games and drop more often even though they have 1 Pure Sequence in their hand.

Impact of Drops Left: For Pool games, the player drop behavior is influenced by the number of possible drops left with the player. For the above hand, we observed that the

[4]Model output is generated using Points 2P model assuming Turn = 2
[5]Model output is generated assuming drops left = 5

Player drops left	5	4	3	2	1
Drop Probability	0.90	0.85	0.80	0.67	0.44

TABLE VI: Player drops left vs Model output - Pool 2P

model drop probability decreased with decrease in drops left for a Pool-2P game (Table VI). This shows that both the model and players tend to drop less as the number of drops left with them decreases.

Feature Importance: We aim to comprehend the influence of various features and quantify their impact on model decision-making. One approach involves selecting a feature, removing it from the training set, evaluating model performance on the test set, and measuring the resulting drop in performance. However, this method is computationally intensive, necessitating model retraining for each feature, and becomes inefficient as the number of features grows.

An alternative method is to employ the feature permutation importance algorithm [23, 24]. In this approach, a feature is chosen; the test set undergoes a random shuffle for that feature, and predictions are made using the original trained estimator. By disrupting the relationship between the feature and target, any drop in model performance concerning the original predictions indicates the extent to which the trained model relies on that specific feature. Importantly, this method doesn't require retraining, making it computationally efficient.

Before calculating feature importance, we need to check and remove correlated features. If we permute any feature(s) which is/are correlated, model still can access them through the other correlated feature(s). This can result in lower feature importance value for them. We calculate Spearman's Rank Correlation [25] for features (Table IV), remove those that are highly correlated (≥ 0.9) (For Pool-6P, *Opponents' drops left* was correlated with *Player drops left*) and then retrain the model. Table VII illustrates the change in Precision and Recall for the Pool-6P model as each feature is permuted. The number of drops left for the player exerts the most significant effect (13.93% drop in Precision, 16.05% drop in Recall). Notably, for most features, there is a higher drop in Recall compared to Precision. However, the Open Deck Card feature has a greater impact on Precision. This can be attributed to including an Open Deck Card enabling players to achieve a better hand, if possible. Without this information, the model tends to recommend dropping the hand more frequently, leading to a higher number of false positives.

Analysis of Model Predictions vs Game Outcomes: We examined the impact of model predictions on game outcomes, focusing on instances where HSPs opted to play in the test set for Pool-6P games (~68,000 instances). We scrutinized the final game outcomes (Table VIII) and compared them with whether the model suggested playing or dropping. When the model advised playing (~62,600 instances), 72% of those games yielded a more favorable outcome (either the player

Feature Permuted	Precision (%)	Recall (%)
Player drops left	-13.93	-16.05
Open Deck Card	-7.01	0.66
Players playing	-5.52	-9.79
Turn No.	-5.19	-8.20
Players joined	-3.27	-7.89

TABLE VII: Feature Permutation: SPADENet Pool 6P Performance change

Model Output	Metric	Favorable Outcome		Non-Favorable Outcome
		Win	Points Lost by <DPP*	Points Lost by >= DPP*
Play	(%) Avg. Points	40 99.4	32 (-9.5)	27 (-51)
Drop	(%) Avg. Points	21 88.4	29 (-10.2)	50 (-56.8)

DPP: Drop Points Penalty*

TABLE VIII: Game Outcome when HSPs played

won or lost by less than the drop penalty points). However, when the model recommended dropping (~5,400 instances) but the players still played, only 50% of those games resulted in a favorable outcome for them. Also, the average points lost by players were lower in games where the model suggested playing (51 vs 56.8). For the games, where the model recommended playing, players also won by a greater margin (99.4 vs 88.4). Statistical validation through a one-tailed, two-sample t-test further confirmed these observations (p-value = 10^{-7}).

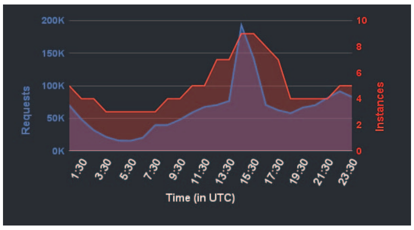

Fig. 6: Requests and Instances Count

Real-time Deployed System Performance: Deploying SPADENet in real-time to generate predictions is a complex engineering challenge due to a large number of games played (millions per day) as well as varying traffic patterns. To tackle this, we developed a model prediction service where the prediction is made available via an API endpoint, hosted on a public cloud platform. The service is hosted across a distributed compute cluster with the ability to scale horizontally based on number of requests and is replicated across multiple availability zones to reduce risk of downtime. Fig. 6 shows the count of requests received and the number of concurrent compute instances running for the service, over a 24-hour period in real-time production environment. The average p99 latency observed was 89 ms.

VIII. Conclusion & Future Work

We presented SPADENet, a generic and improved multi-input deep neural network architecture designed to comprehensively capture and combine the player's hand with the ever-evolving visible game state outside of the player's hand. This enabled incorporating the intricacies of any game variant for skill-based card games. With the example of Online Rummy, we demonstrated how it can be easily adapted for assessing first drop (or play) decisions to the different variants of the same game of Online Rummy (such as Points, Pool, Deals, etc.) and compared its performance with existing work. We showed how our model not only outperformed them but also increased the odds of a favorable game outcome. Future work includes adapting the architecture to other skill-based card games like Poker. Poker, like Rummy, requires players to form specific combinations of cards to win the game. Players make raise or fold decisions based on the cards they hold, as well as those they observe from other players. Their decision is also influenced by opponents' raise or fold behavior. So while the cards' information can be represented in the CNN block (Players' Hand Information), the other game information like opponent playing behavior, which is outside of the Players' hand and is visible to all, can be represented using the DENSE block in the SPADENet architecture respectively. Hence the CNN block will learn the relevance of the poker card melds and the DENSE block will learn the influence of opponents' decisions towards making optimum raise or fold decision respectively. With such information at hand, a holistic view of any player's skill can be derived for any skill-based card game. This can be used as input for player upskilling by building personalized learning journeys. From an Online Rummy perspective, it would mean sharing tips that consider the nuances of individual variants like Points, Pool, Deals, etc.

References

[1] Statista Market Insights, https://www.statista.com/outlook/dmo/app/games/card-games/worldwide

[2] Finite Games with Incomplete Information. In: Game Theory. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-69291-1_5

[3] Silver, D., Huang, A., Maddison, C. et al. Mastering the game of Go with deep neural networks and tree search. Nature 529, 484–489 (2016). https://doi.org/10.1038/nature16961

[4] Silver, D., Schrittwieser, J., Simonyan, K. et al. Mastering the game of Go without human knowledge. Nature 550, 354–359 (2017). https://doi.org/10.1038/nature24270

[5] N. Brown and T. Sandholm, "Superhuman ai for heads-up no-limit poker: Libratus beats top professionals," Science, 2017.

[6] Noam Brown, Tuomas Sandholm ,Superhuman AI for multiplayer poker.Science365,885-890(2019). 10.1126/science.aay2400

[7] Noam Brown, Tuomas Sandholm, and Brandon Amos. Depth-limited solving for imperfect information games. In Advances in Neural Information Processing Systems, pages 7663–7674, 2018.

[8] Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. 2020. Combining deep reinforcement learning and search for imperfect-information games. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20). Curran Associates Inc., Red Hook, NY, USA, Article 1431, 17057–17069.

[9] N. R. Sturtevant and A. M. White, "Feature construction for reinforcement learning in hearts," in Computers and Games. Berlin, Heidelberg:Springer Berlin Heidelberg, 2007, pp. 122–134.

[10] Kotnik, Clifford Kalita, Jugal. (2003). The Significance of Temporal-Difference Learning in Self-Play Training TD-Rummy versus EVO-rummy.. 369-375.

[11] Nguyen, V. D., Doan, D., Neller, T. W. (2021). A Deterministic Neural Network Approach to Playing Gin Rummy. Proceedings of the AAAI Conference on Artificial Intelligence, 35(17), 15622-15629. https://doi.org/10.1609/aaai.v35i17.17840

[12] Ahmed, A., Leppo, J., Lesniewski, M., Patel, R., Perez, J., Blum, J. (2021). A Heuristic Evaluation Function for Hand Strength Estimation in Gin Rummy. Proceedings of the AAAI Conference on Artificial Intelligence, 35(17), 15465-15471. https://doi.org/10.1609/aaai.v35i17.17820

[13] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks, Eprint Arxiv, 2013.

[14] Xiao Y, Tian Z, Yu J, Zhang Y, Liu S, Du S, Lan X. A review of object detection based on deep learning. Multimed Tools Appl. 2020;79(33):23729–91.

[15] Ker J, Wang L, Rao J, Lim T. Deep learning applications in medical image analysis. IEEE Access. 2017;6:9375–89.

[16] Sharanya Eswaran, Vikram Vimal, Deepanshi Seth, and Tridib Mukherjee. 2020. GAIM: Game Action Information Mining Framework for Multiplayer Online Card Games (Rummy as Case Study). In Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part II. Springer-Verlag, Berlin, Heidelberg, 435–448 . https://doi.org/10.1007/978-3-030-47436-2_33

[17] Yu Sun, Lin Zhu, Guan Wang, Fang Zhao, "Multi-Input Convolutional Neural Network for Flower Grading", Journal of Electrical and Computer Engineering, vol. 2017, Article ID 9240407, 8 pages, 2017. https://doi.org/10.1155/2017/9240407

[18] Raquel Sánchez-Cauce, Jorge Pérez-Martín, Manuel Luque, Multi-input convolutional neural network for breast cancer detection using thermal images and clinical data, Computer Methods and Programs in Biomedicine, Volume 204, 2021, 106045, ISSN 0169-2607,https://doi.org/10.1016/j.cmpb.2021.106045.

[19] Oktay, O. et al. (2016). Multi-input Cardiac Image Super-Resolution Using Convolutional Neural Networks. In: Ourselin, S., Joskowicz, L., Sabuncu, M.R., Unal, G., Wells, W. (eds) Medical Image Computing and Computer-Assisted Intervention - MICCAI 2016. MICCAI 2016. Lecture Notes in Computer Science(), vol 9902. Springer, Cham. https://doi.org/10.1007/978-3-319-46726-9_29

[20] Seeland M, Mäder P (2021) Multi-view classification with convolutional neural networks. PLoS ONE 16(1): e0245230. https://doi.org/10.1371/journal.pone.0245230

[21] Kingma, Diederik Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

[22] Prechelt, L. (1998). Early Stopping - But When?. In: Orr, G.B., Müller, KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 1524. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-49430-8_3

[23] Breiman, Leo."Random Forests." Machine Learning 45 (1). Springer: 5-32 (2001)

[24] Fisher, Aaron, Cynthia Rudin, and Francesca Dominici. "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously." http://arxiv.org/abs/1801.01489(2018)

[25] Spearman, C. "The Proof and Measurement of Association between Two Things." The American Journal of Psychology 15, no. 1 (1904): 72–101. https://doi.org/10.2307/1412159.