Artificial Intelligence and Social Card Games: Exploring the Prospects of a Dixit-playing AI

Nils Hendrix and Pieter Spronck
Tilburg School of Humanities
Tilburg University
Tilburg, Netherlands
Email: n.j.m.hendrix@tilburguniversity.edu

Abstract – **Little research is available on designing artificial intelligence (AI) for playing social board or card games to this date. The lack of research might be related to the challenge of equipping the AI with all game-relevant information, which can be extensive in the context of social games. The purpose of the current study is therefore to demonstrate that social games can be solved by using relatively simple techniques. For this purpose, an AI was created for the social card game Dixit. In Dixit, players have to figure out the correct referent card belonging to a cryptic hint. Each card contains an illustration that can be interpreted in various ways. By using an internet-based approach, the Dixit-playing AI was able to identify the correct card with near-human accuracy on average (44.7% versus 45.6%). This finding implicates that the internet can be used to achieve human-level performance in social games.**

1. Introduction

To this date, there is little research into creating an artificial intelligence (AI) for playing social board or card games. As far it can be considered a social card game, Poker is one of the very few social games that has been the subject of AI research (e.g. Billings, Davidson Schaeffer, & Szafron, 2002; Bowling, Burch, Johanson, & Tammelin, 2015; Rubin & Watson, 2011). The lack of research might be related to the difficulties in developing a social game AI. A challenge that one could conceive of is the requirement of "common ground", which can be broadly defined as "the mutual knowledge, beliefs, and assumptions shared by the speaker and addressees" (Clark, Schreuder, & Buttrick, 1983). Common ground is expanded during daily interactions and contains a vast amount of shared information. For instance, if someone wants to order a specific bouquet at the florist's, it would require countless assumptions about what flowers look like, customs in shops, pointing gestures, natural language, et cetera. Thus, it is generally accepted that common ground is essential for mutual understanding.

As social games naturally depend on social skills and interactions, the importance of common ground in this context can be easily recognized. One particularly challenging social card game is Dixit. In Dixit, each player receives six mysteriously illustrated cards out of a set of 84 (Dixit, 2011). A subset of playing-cards is shown in Figure 1. One of the players (storyteller) gives a cryptic hint about one the cards he or she is holding and puts it face-down on the table. The other players select one of their own cards that fit the description and put it on the storyteller's card. Then, the cards are shuffled and turned over. All players, except the storyteller, have to secretly vote on the card that they think to be the storyteller's card. They receive points when they voted on the storyteller's card and when others have voted on their card. The storyteller only gets points when at least one but not all of the players has voted on his or her card. Therefore, the success of a Dixit player seems to ultimately depend on the ability to assess the boundaries of the common ground.

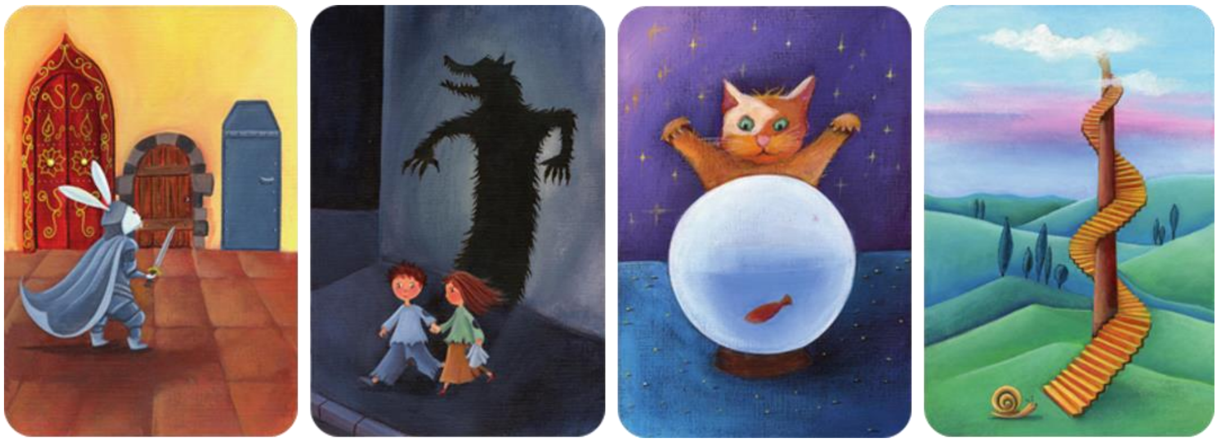
Figure 1. A subset of Dixit playing-cards. The images can be interpreted in various ways.

When designing a social game AI, a relevant question might be how to equip the AI with a common ground that is large enough to cover all game-relevant information. Since games such as Dixit require a vast amount of shared knowledge, one might consider this an impossible task at first glance. The purpose of the current study is to demonstrate that social card games can be solved by using relatively simple techniques. In this paper, a Dixit-playing AI will be developed and evaluated in a Python programming environment. The scope of the research will be limited to the AIs ability to pick out the storyteller's or target card on the basis of a given hint. It will be shown that a near-human performance can be obtained by using an internet-based approach.

2. Related work

As explained in the introduction, little research is available on developing AI's for social card or board games. In the light of the investigated internet-based approach, perhaps the most relevant, related work is the Watson technology of IBM Research (Ferrucci et al., 2010). IBM Research built a computer system, called Watson, that could compete at human-champion level on the American TV quiz show Jeopardy. In this quiz show, three players compete against each other for three rounds in which they have to quickly formulate an appropriate question (clue) for a given answer (response). For instance, when given the clue "this drug has been shown to relieve the symptoms of ADD with relatively few side effects", the correct response would be "What is Ritalin?". Players earn a certain amount of money when they have been the quickest to give the correct response, but lose that money when they answer incorrectly. The player who has the most money at the end of the final round wins the game.

The Watson AI was implemented by using IBM's DeepQA architecture (Ferrucci et al., 2010). DeepQA is a massively parallel probabilistic evidence-based architecture for handling general question-answering (QA) problems. The first step in this architecture is content acquisition, which consists of manual and automatic steps. One manually builds a baseline corpus by analysing example questions and deciding what sources might provide evidence for answering them, such as encyclopaedias, dictionaries and news articles. Then, DeepQA automatically expands the baseline corpus by extracting relevant documents from the web. As the live system does not have access to the web, this corpus forms the basis of the available evidence.

After the content acquisition, the actual QA-process consists of the following steps (Ferrucci et al., 2010). An incoming question is first analysed, that is, Watson tries to understand what is being asked and determines the next processing steps. If necessary, the question is decomposed into sub-questions. The results of the question analysis are used to produce hundreds of candidate answers by searching through the corpus and extracting answer-sized snippets from the search results. Each candidate answer (e.g. "Ritalin") that is put back into the question is considered a hypothesis. The number of hypothesises is filtered down to about one hundred, and the selection of hypothesises is then evaluated by using an evidence scoring algorithm that gathers additional supporting evidence in the process. In the end, the hypothesises are merged and ranked, and the best-supported hypothesis is selected as an answer.

The Dixit AI presented in this work is inspired by the Watson technology. To a certain extent, the challenge of unravelling hints of Dixit players could be compared to the QA-problem that is handled by the DeepQA architecture; one needs to figure out the correct referent of a given reference on the basis of trivial knowledge. Consequently, even though its workings are greatly simplified, the basic Dixit AI's decision structure is rather similar to that of Watson. As with Watson, the Dixit AI selects playing cards based on evidence gathered from data sources that are collected from the internet. Since the kinds of player strategies or hints will determine what data needs to collected for the Dixit AI, an outline of possible player strategies will be given in the next section.

3. Player strategies in Dixit

In order to assess what knowledge is required to guess the correct target card, it is necessary to map the hint strategies of Dixit players. In general, one could make a distinction between two types of hints: 1-layer (1L) and 2-layer (2L) hints. 1L-hints have a direct relation to the image depicted on the target card, and 2L-hints have an indirect relation to it and are therefore harder to interpret. Although one could give higher layered hints in theory (e.g. L3- or L4-hints), such hints are not used in practice.

1L-hints could be divided into four categories. Each category can be illustrated by means of the playing-card in Figure 2. Firstly, one can give a description of what happens on the card. For instance, "follow me, gentleman" or "the party doesn't start 'till I walk in". Secondly, a description of the entire scene or a salient entity can be given using cultural references. One may refer to famous persons (e.g. "Magritte" , "Esher", "Abraham Lincoln" – two surrealist painters and one president) or popular movies (e.g. "The Truman Show"). Thirdly, players can hide a salient entity in the hint itself, such as "cloud" in "is your file in the *cloud*?" or "stairway" in "*Stairway* in Heaven". Finally, an alternative description or synonym can be given for an entity in the image. For instance, "vampire" can refer to the man with the top hat, and "mirror" can refer to the reflection in the water.

2L-hints can be divided into two categories. In the first category, players indirectly refer to objects by using a quote, actor or other recognizable aspect from a cultural artefact. One may use the quote "Cue the sun" or the concept "Reality TV" to refer back to "The Truman Show". Other examples are "Knock Knock" ("Knocking on heaven's *door*", a Bob Dylan song) and "Cheek to Cheek" (a song from the movie *Top Hat*), which refer to a specific entity in the image. In the second category, players use a feature of an alternative description of an entity, such as "Transylvania" or "bloodsucker" for "vampire".

The described index of player strategies will determine what data will be collected for the Dixit AI. The AI should first have access to a large database of hints with the corresponding target cards, because some pop-cultural references are used fairly often (e.g. “The Truman Show”) and there are a limited number of salient entities in the images. This way, a considerable number of hints can be easily unravelled by matching them to this database. Since players may hide entities in their descriptions that can be observed on the cards, the database should be extended with “ground-truth” descriptions of directly perceivable entities. For dealing with more complex references, relevant information could be retrieved from the internet via a search engine, such as Google. Complex, but popular references may immediately pop up on the first results page, which could break down 2L-hints into 1L-hints. In the next section, it will be explained how these data were collected for the purpose of this study.

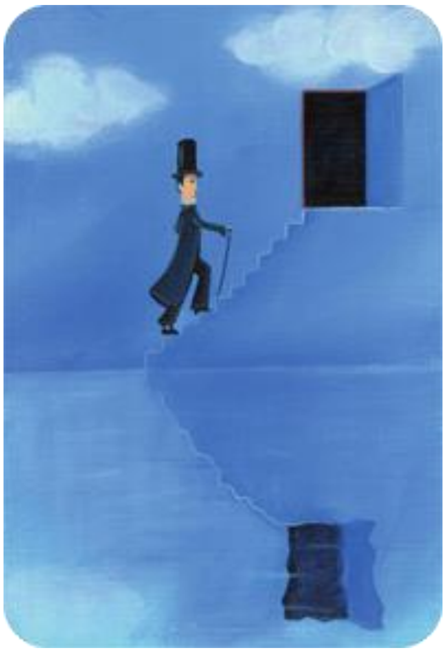

Figure 2. An example Dixit-playing card which illustrates the range of different strategies that can be used to refer to target cards.

4. Method

4.1. Data

Two datasets and four Python dictionaries were created in total. The main dataset consisted of 55,101 game rounds from 25,750 Dixit games, which were scraped from the online games portal Boiteajeux. An overview of the features of the dataset is shown in Table 1. Each row of the dataset represented a game round, and included – among other things – the given hint, the cards that were played, the target card, and the number of votes that each card received. The games in the dataset were played by four (48.1%), five (17.5%), or six (34.4%) players.

Table 1

Dataset Features

Feature	Description
game	Game number. This is an unique number.
round	Round number.
players	Number of players. The value range is 4-6. The frequency distribution is 48.1% (4), 17.5% (5), and 34.4% (6)*.
description	Description of the storyteller.
explanation	Explanation of the description of the storyteller. This explanation is seldom provided.
c[1/2/3/4/5/6]_card	ID number of the card. The value range is 1-84.
c[1/2/3/4/5/6]_picks	Number of picks or votes that the card has got.
c[1/2/3/4/5/6]_target	Whether the card is the target card (storyteller's card) or not.

Note. *: the number of players are in parentheses.

The second dataset contained all players (*n* = 4,222) that occurred in the main dataset with an average accuracy score for each player; that is, the percentage of correct identifications of the target card. The mean accuracy score of the players was 45.6% (*SD* = 16.7) overall. More specifically, the mean accuracy score was 53.1% (*SD* = 15.6), 45.0% (*SD* = 18.0), and 40.2% (*SD* = 16.4) when there were respectively four, five, or six cards on the table.

A "hint", "ground-truth", "URL" and "TFxIDF" Python dictionary were created. In the next subsection it will be discussed how these dictionaries were implemented. In the hint dictionary all descriptions were stored as keys and the accompanying target cards and their number of occurrences were stored as values. In other words, for each unique description in the dataset it was stored to which cards and how often the description had ever referred.

The ground-truth dictionary contained for each of the 84 playing cards (keys) a list of descriptive words about entities that can be directly observed on the cards themselves (values). The images of all playing-cards were scraped from Boiteajeux.net (as a part of the previously mentioned process) and were manually and personally reviewed.

The URL dictionary was created by entering every description of the dataset into the Google search engine using an automated process; the URL of the first occurring Wikipedia or Goodreads (book quotes) webpage on the first results page was captured. Hence, Wikipedia or

Goodreads URL’s were stored as dictionary keys and the corresponding target cards and their frequencies as values (the structure is similar to the hint dictionary).

The TFxIDF dictionary was generated by calculating TFxIDF values over the words from all hints, explanations and ground-truth descriptions for each playing-card. The TFxIDF value of each word was calculated by using the log inversed document frequency and a term frequency that was relative to the total number of words belonging to a given card. Thus, in the TFxIDF dictionary the cards were stored as keys and the corresponding words and TFxIDF values as values.

4.2. Model

The main task of the Dixit AI was to guess the correct target card on the basis of a given hint. Figure 3 shows the complete procedure that was used to accomplish this task. Given a certain hint, the AI first searched for a match in the hint dictionary. When a match was found, it checked whether one of the cards ‘on the table’ occurred in the dictionary entry and it picked the target card with the highest frequency if this was the case. When no match was found or no cards on the table were in the matching dictionary entry, it entered the hint into the Google search engine and retrieved the first occurring Wikipedia or Goodreads URL on the first results page. It was thought that these webpages would regularly show up in the results, as Dixit players often use popular pop-cultural references. In case a URL was retrieved, the AI searched for a match by looking up the URL in the URL dictionary and used the same decision strategy as before. When no match was found, it scraped all words from the webpage and determined a similarity score for each card on the table.

This similarity score was calculated by adding up the TFxIDF values of all words that were on the retrieved webpage using the TFxIDF dictionary entry of a given card. The idea was that the better the match between the words on the webpage and those of the card entry, the more likely that the card will be the target card. The card with the highest score was chosen by the AI. Finally, when neither a Wikipedia nor a Goodreads page was found, the same decision strategy was employed, except that the words from the hint were compared against the TFxIDF dictionary instead of the words from a webpage.

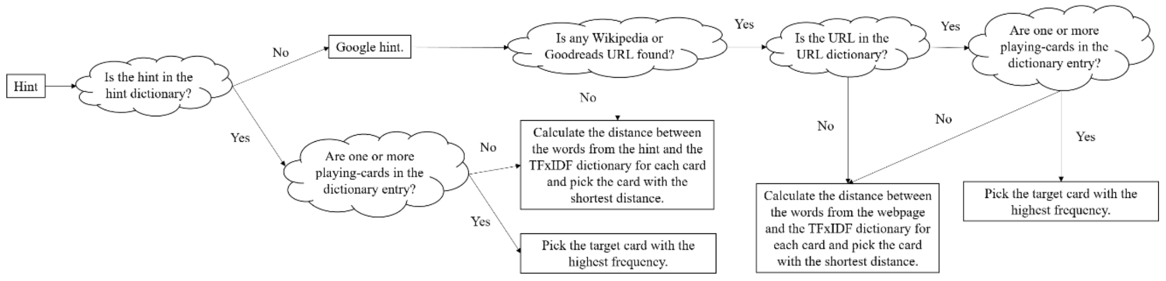
Hint
Is the hint in the hint dictionary?
No
Google hint.
Is any Wikipedia or Goodreads URL found?
Yes
Is the URL in the URL dictionary?
Yes
Are one or more playing-cards in the dictionary entry?
No
No
No
Yes
Yes
Are one or more playing-cards in the dictionary entry?
No
Calculate the distance between the words from the hint and the TFxIDF dictionary for each card and pick the card with the shortest distance.
Yes
Pick the target card with the highest frequency.
Calculate the distance between the words from the webpage and the TFxIDF dictionary for each card and pick the card with the shortest distance.
Pick the target card with the highest frequency.

Figure 3. Decision structure of the Dixit AI for guessing the correct target card.

5. Experiments and results

The Dixit AI was tested on 5,510 hints from the main dataset in a Python programming environment. The testing process was fully automated by using a custom script. Six different configurations of the AI were tested via two experiments. The accuracy scores of all configurations are shown in Table 2. In the first experiment, configurations 1-4 were tested. In these configurations the number of considered Google results was varied from 1 (first result) to 10 (all links on first result page).

On the basis of these results, a second experiment was conducted in which configuration 5 and 6 were built and tested. This time, only the words from the hint were compared to the TFxIDF dictionary and no words from Wikipedia or Goodreads webpages were scraped. The only difference between the two configurations was that the former stored only one Wikipedia or Goodreads URL from the results, while the latter used an extended URL dictionary that contained all top-3 Google links per hint.

In the end, the highest accuracy was yielded by configuration 6 (44.7%). This accuracy score was higher than that of configuration 1 (43.9%; *t*(5509) = 6.66, *p* < .001, two-tailed) and configuration 5 (44.3%; *t*(5509) = 4.81, *p* < .001, two-tailed), but lower than the mean accuracy score of the human players in the dataset (45.6%; *t*(4424) = 3.74, *p* < .001, two-tailed). Also, the accuracy score of configuration 5 was higher than that of configuration 1 (*t*(5509) = 4.59, *p* < .001, two-tailed).

Table 2

Accuracy Score per Dixit AI Configuration (n = 5510)

Configuration	Number of considered Google results	URL dictionary (normal / extended)	Accuracy per decision strategy (in percentages)				
			Hint match*	Hint TFxIDF	Web match	Web TFxIDF	Total
1	1	normal	67.8 (894)	38.6 (3742)	66.5 (272)	31.1 (602)	43.9
2	3	normal	68.7 (894)	37.3 (2848)	66.6 (524)	31.1 (1244)	43.8
3	5	normal	68.6 (894)	37.0 (2489)	64.9 (575)	31.8 (1552)	43.6
4	10	normal	68.0 (894)	37.1 (2713)	66.9 (474)	30.7 (1429)	43.0
5	3	normal	68.2 (894)	36.0 (4095)	68.1 (521)		44.3
6	3	extended	68.7 (894)	33.8 (3734)	66.2 (882)		44.7

Note. The frequency of a decision strategy is in parentheses. *: there is variation in the accuracy scores since there can be multiple frequency maxima in a hint dictionary entry; in these cases, the AI randomly picks one of the target cards with the highest frequency.

6. Discussion and conclusion

The results show that the Dixit AI presented in this paper can guess the correct target card with near-human accuracy (44.7% versus 45.6%). The performance was best when (1) removing the comparisons between webpage words and those in the TFxIDF dictionary, and (2) extending the URL dictionary. It can be noticed in Table 2 that the performance decreases when more links are considered while using webpage-TFxIDF dictionary comparisons; these comparisons are used more frequently at the expense of the more accurate hint-TFxIDF dictionary comparisons. Contrary to the expectations, it appears that webpage-TFxIDF dictionary comparisons are counterproductive.

One explanation could be that the retrieved Google results are less relevant than assumed. Whereas a Wikipedia or Goodreads webpage contains many more words than a single hint, it may be concluded that hints still contain more relevant words on average. The webpage-TFxIDF dictionary comparisons might yield better accuracy scores when scraping words from any web source that is in the top results of Google (not only Wikipedia and Goodreads), but this possibility is deemed rather small; even the web comparisons with Wikipedia and Goodreads pages taken from the first result, which should be most relevant, led to a lower score than not using such comparisons at all. Thus, it is recommended to investigate strategies other than webpage-TFxIDF dictionary comparisons in future research.

Another notable observation is that the accuracy of the hint-TFxIDF dictionary comparisons considerably drops when the extended URL dictionary is used (configuration 6). Even though the decision strategy was used about equally often, there is a difference of 4.8 percent points between configuration 1 and 6. It seems that the hints that are evaluated via the web TFxIDF strategy are more difficult to process when they are evaluated with the hint TFxIDF strategy. There might be therefore a relation between hints that generate relevant Google results and their distance to the card entries in the TFxIDF dictionary. The exact nature of this relation remains unclear, however, as its presence it rather unintuitive; one would expect that hints that generate relevant Google results refer to popular items (e.g. movie and book quotes) and that these hints should match the pattern in the TFxIDF dictionary reasonably well. This relation could be clarified in future research.

An examination of the AI performance at card-level indicated that the AI had difficulties with processing hints that contain no relevant words that can be traced back to directly perceivable entities in the images. For instance, the hint “always look on the bright side of life” was used for a variety of playing-cards without necessarily making a reference to Monty Python’s Life of Brian or referring to a specific entity. Instead, it often referred to the entire situation. In these cases, the AI should possess some context-awareness or it should simply have access to a larger database of hints that better captures these hint patterns.

The reported accuracy score could be improved in future work by adding more hints to the dataset. The URL and TFxIDF dictionary can be extended in this way, which effectively brute-forces the problem of guessing the correct target card. Although, when a performance far beyond that of human players is desired, probably a new strategy should be implemented into the AI. New strategies may involve the use of computer vision which enables the AI to directly perceive the playing cards and to make inferences about them. When a respectable performance on card guessing task is reached, the research on Dixit AI could move to the automatic generation of hints.

Altogether, the results of the current study have shown that human-level social game AI can be possibly designed by using relatively simple techniques. The internet can serve as an infinite source of data to power the performance of social game AI. The vast amount of information that is stored in common ground, which seems uncapturable at first glance, can be accessed and explored by using popular search engines, such as Google. As the fields of datamining and machine learning are rapidly developing, the research into social game AI may achieve a breakthrough in the nearby future.

References

Billings, D., Davidson, A., Schaeffer, J., & Szafron, D. (2002). The challenge of poker. *Artificial Intelligence*, *134*(1-2), 201-240. doi:10.1016/S0004-3702(01)00130-8

Bowling, M., Burch, N., Johanson, M., & Tammelin, O. (2015). Heads-up limit hold'em poker is solved. *Science*, *347*(6218), 145-149. Retrieved from http://agoptimizer.com/wp-content/uploads/2015/01/Cepheus-AI-algorithm-poker.pdf

Clark, H. H., Schreuder, R., & Buttrick, S. (1983). Common ground at the understanding of demonstrative reference. *Journal of verbal learning and verbal behavior*, *22*(2), 245-258. Retrieved from http://web.stanford.edu/~clark/1980s/Clark.Schreuder.83.pdf

Dixit. (2011). Dixit. Retrieved from http://www.bordspel.com/artikel/dixit.html

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., ... & Schlaefer, N. (2010). Building Watson: An overview of the DeepQA project. *AI magazine*, *31*(3), 59-79. Retrieved from https://xa.yimg.com/kq/groups/21607652/1092283046/name/AIMagzine-DeepQA.pdf

Rubin, J., & Watson, I. (2011). Computer poker: A review. *Artificial Intelligence*, *175*(5-6), 958-987. doi:10.1016/j.artint.2010.12.005