Date of acceptance Grade

Instructor

Machine Learning Strategies in Cribbage

Sean R. Lang

Helsinki October 18, 2017

UNIVERSITY OF HELSINKI
Department of Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty Faculty of Science	Laitos — Institution — Department Department of Computer Science

Tekijä — Författare — Author

Sean R. Lang

Työn nimi — Arbetets titel — Title

Machine Learning Strategies in Cribbage

Oppiaine — Läroämne — Subject

Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year October 18, 2017	Sivumäärä — Sidoantal — Number of pages 0 pages + 4 appendices

Tiivistelmä — Referat — Abstract

ACM Computing Classification System (CCS):

Avainsanat — Nyckelord — Keywords

machine learning, cribbage

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Contents

1 Introduction

I just want to see if unicode characters will appear. Jag måste se om unicode-characterer funger. minä haluan katsoa kirjain öööö.

Generic introductory stuff giving a sentence or probably a paragraph about each of the sections covered.

2 Literature Review

Overview of the current literature surrounding this topic.

- Research done in cribbage
- Research done in related imperfect information games (e.g. poker)
- Overview on expert witness machine learning
- Any other topic that ends up getting used (Bayesian logic, statistics?)

2.1 Prior Cribbage Research

3 Data and Methods

Walk-through of how I went about researching the topic. (Maybe I should keep a diary or log or something so this isn't half made up at the end.) When all is said and done, include the final "output" graph in some easily viewable format (121-by-121 table of miniature bar graphs, RGB combination in each cell?).

- Creation of framework
- Creation of "expert advisor"
- Training method for listener/combiner
- Initializations for training

3.1 Methods

3.1.1 Loss Function

A good loss function is critical to the proper training of a machine learning classifier. In classroom examples in which data can be mapped into a multidimensional space, this is typically accomplished by either a euclidean distance, or some other measure of how "wrong" the prediction is based on how much it varies from what is classified as "correct." In learning to predict using expert witnesses' testimony, this is measured in terms of *regret*. However, even though it is inevitable for the human player to think back on what could have been if he or she had chosen a different arrangement of cards to keep and throw, this is not a practically applicable loss function measure for the following reasons:

1. To evaluate how each hand "would" have done in any specific point in the game, a new "branching" game would have to be run through at that specific point. Basically, this means that a giant search tree would need to be made where the branches are randomly created using draws from the deck. Therefore the notion of "regret" is not easily practically computable.

2. It is difficult to determine what a better hand would have been in any case. For example, a hand that scored higher would not always be better. Furthermore, a hand that uses a higher score may not peg as well, etc. Therefore, it is difficult to be able to determine what is "correct" to calculate a distance from that.

Therefore, it may be useful instead to create a different system of punishment and reward rather than pure Loss. There are two possible routes I can think of at this junction.

1. Use the point spread between yourself and the opponent for the round.

   The problem with this strategy is that it depends on the opponent's performance as well as your own. In the case of our simulation, this is further problematic because the opponent's playing strategy and style is not known, which means this can't be rightfully evaluated.

2. In what position was the player left after playing with this hand.

   (a) Using intrinsic values for each playing state.

   This method relies upon previous knowledge of the game in order to set up position in which the player is at an advantage or disadvantage. While this may be acceptable for this small application, it should not be used in the general case.

   (b) Using a form of back-propagation.

   Each positions value will be computed based on the amount of times a player at that position ended up winning the game, and by how much. This would require the agent to be trained in reverse, so to speak. Early games could start at scores like 119 to 115 and compute which are likely to win, then make their way backwards to lower scores. Another option would be for each agent to keep track of its own path of scores and reward/punish all at the end, perhaps with different weights.

4 Findings

What are the final results of the “experiment.”

- Where did each initialization family end up going?
- Were they different or did the agent learn a “single” strategy in general?
- All this and more will be answered . . . after the break!

5 Discussion

This is the difficult part. What does this part mean? And how does it differ from what I’ve already covered above in the Findings section.

6 Conclusion

Generic closing remarks and rephrasing of the original introduction again in parting. Because this is a thesis and likely fairly long, “In section X, we covered Y” is probably allowed and not tacky.

References

ACC URL `http://www.cribbage.org/rules`.

BDSS02 Billings, D., Davidson, A., Schaeffer, J. and Szafron, D., The challenge of poker. *Artificial Intelligence*, 134,1-2(2002), pages 201–240.

BFGL17 Brown, A., Fisher, G., Gilman, S. and Lang, S., Overhead delivery system for transporting products, April 13 2017. URL `https://www.google.com/patents/US20170101182`. US Patent App. 14/881,217.

KS02 Kendall, G. and Shaw, S., Investigation of an adaptive cribbage player. *International Conference on Computers and Games*. Springer, 2002, pages 29–41.

Mar00 Martin, P. L., *Optimal Expected Values for Cribbage Hands*. Ph.D. thesis, Harvey Mudd College, 2000.

O’C00 O’Connor, R., Temporal difference reinforcement learning applied to cribbage. Technical Report, University of California, Berkeley, 2000. URL `http://r6.ca/cs486/`.

PMASA06 Ponsen, M., Munoz-Avila, H., Spronck, P. and Aha, D. W., Automatically generating game tactics through evolutionary learning. *AI Magazine*, 27,3(2006), page 75.

Pon04 Ponsen, M., *Improving adaptive game AI with evolutionary learning*. Ph.D. thesis, TUDelft, 2004.

RW11 Rubin, J. and Watson, I., Computer poker: A review. *Artificial Intelligence*, 175,5-6(2011), pages 958–987.

SPSKP06 Spronck, P., Ponsen, M., Sprinkhuizen-Kuyper, I. and Postma, E., Adaptive game ai with dynamic scripting. *Machine Learning*, 63,3(2006), pages 217–248.