

# Hearthstone++ : Hearthstone simulator with reinforcement learning

Young Joong Kim<sup>1</sup>, ChanHo Ohk<sup>2</sup>, SeungHyun Jeon<sup>3</sup>, SungHyun Kim<sup>4</sup>, JunYoung Park<sup>5</sup>

Department of Computer Science, Hanyang University, [revsic99@gmail.com](mailto:revsic99@gmail.com)<sup>1</sup>  
Nexon Korea<sup>2</sup>, Satrec Initiative, Republic of Korea<sup>3</sup>  
Daedeok Software Meister High School<sup>4</sup>, Korea Digital Media High School<sup>5</sup>

## Backgrounds

- Hearthstone is the online collectible card game.
- AlphaGo Zero achieved superhuman performance in the game of Go by combined MCTS with self-playing reinforcement learning.
- Chance Event Bucketing with pre-sampling and Hierarchical policy helps reduce the searching space of MCTS
- Symbolic reasoning with heuristic utility function overwhelm the UCT based MCTS method in the game Hearthstone.

## Methods

- Our objective is implementing the model playing the game Hearthstone, MCTS with self-playing reinforcement learning algorithm like AlphaZero (David Silver et al., 2017)
- Now our progress is under implementing the Hearthstone simulator with C++, please note that the below is just for the plan with some experiments.
- Starting with feature construction of card data embedding, it can be represented as cluster, cost, health, attack value and preprocessed description.
- Description is preprocessed as sum of vector representation of tokenized text as word2vec, and cluster is decided by clustering algorithm with remained card features.
- The single feature extractor network  $f_{\theta}$  is combined to MCTS. It connects to high-level policy network, low-level policy network and value network guiding the MCTS to select and expand trees.
- The searching space can be reduced by grouping the policies related to card clusters and pre-sampling.

$v$  : expected value     $z$  : game result

$\mathbf{p}$  : expected policy     $\boldsymbol{\pi}$  : MCTS policy

Generate Prior and Value:  $(\mathbf{p}, v) = f_{\theta}(s)$

Generate MCTS Policy:  $\boldsymbol{\pi} = \alpha_{\theta}(\mathbf{p}, s) = \arg \max_{a \in \mathbf{p}} Q(s, a) + U(s, a)$

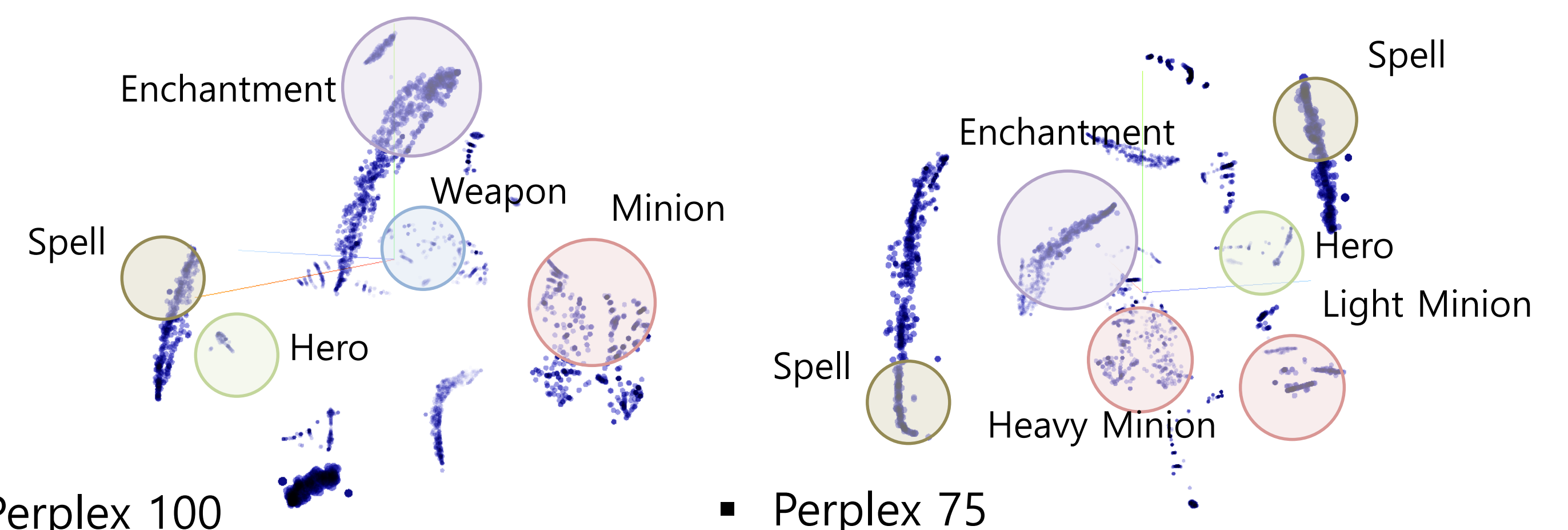
Objective :  $l = (v - z)^2 + \boldsymbol{\pi}^T \log(\mathbf{P}) + c \|\theta\|^2$

## Experimental Results

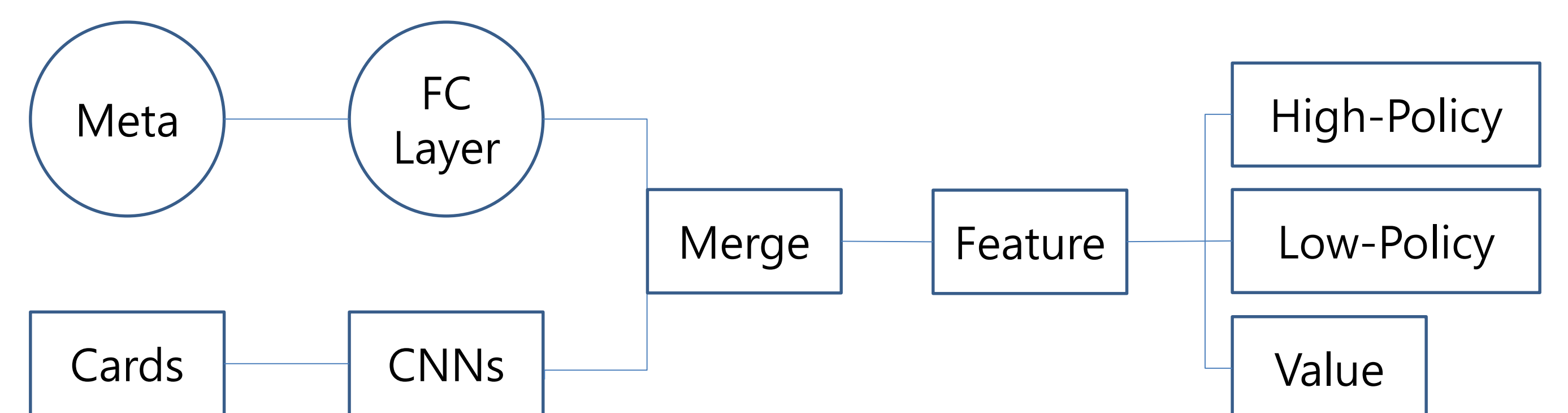
### Hearthstone++ Repository

<https://github.com/utilForever/Hearthstonepp>

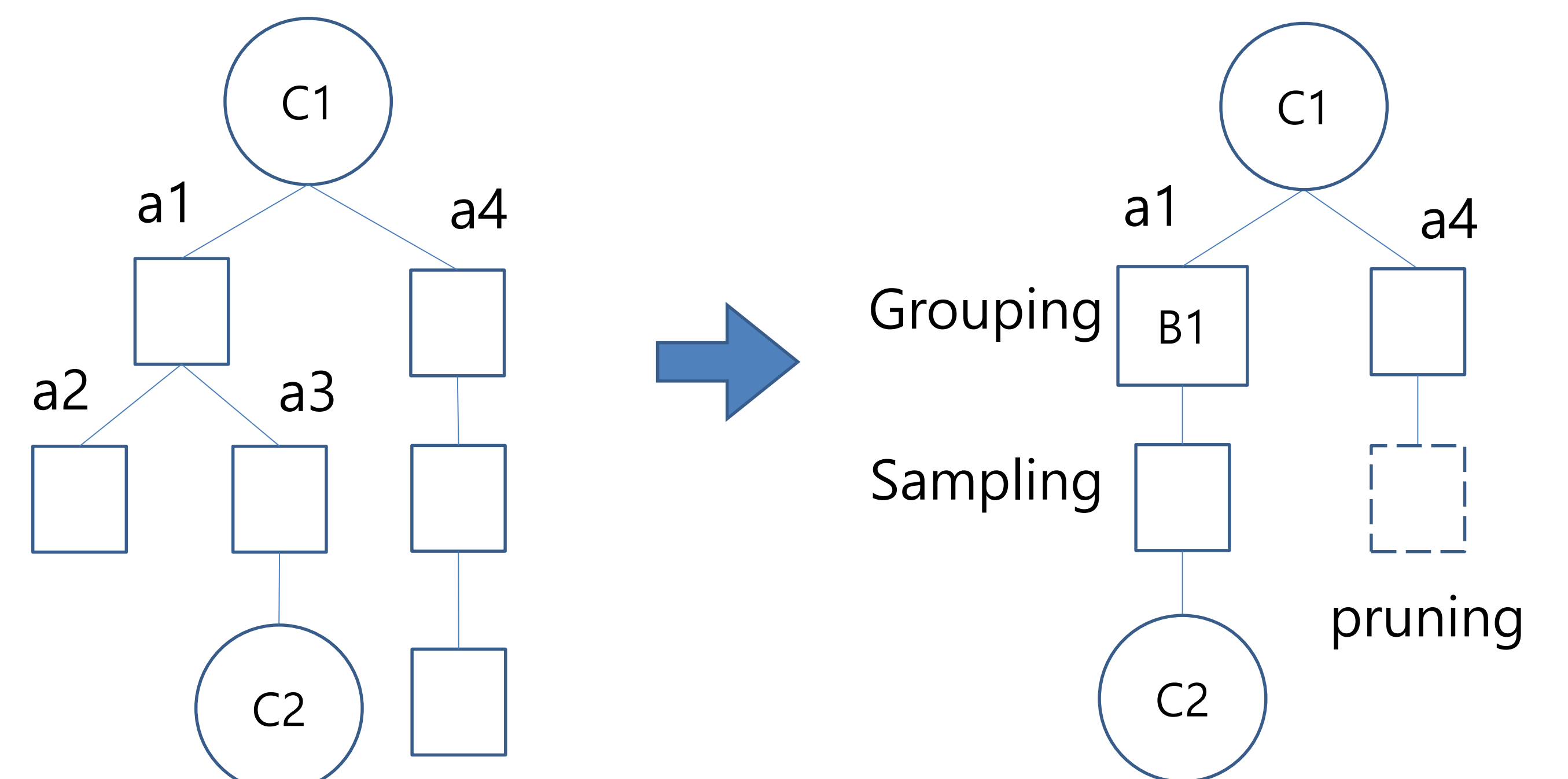
### t-SNE Visualization of Card Data Embedding



### Feature Extractor



### Reducing Search Space



## Discussion

- Searching space of MCTS can be reduced by grouping the policies related to clusters constructed by some clustering algorithms with vector representation of card data.
- Hierarchical policy allows MCTS to rapidly rollout the nodes by early cutoff on the high level policy without elaborating the target as low level policy.
- Extracted deck representation is expected to be interpretable as deck property, and model expected to have ability of anticipating the property of other's deck with visible cards although it is an imperfect information about deck.

## References

- David Silver, Julian Schrittwieser, Karen Simonyan, et al. "Mastering the Game of Go without Human Knowledge." Nature, 550:354-359, 2017
- David Silver, Thomas Hubert, Julian Schrittwieser, et al. "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm", arXiv preprint arXiv:1712.01815v1, 2017
- Shuyi Zhang, Michael Bruo, "Improving Hearthstone AI by Learning High-Level Rollout Policies and Bucketing Chance Node Events", IEEE's 2017 Conference on Computational Intelligence in Games, CIG 2017
- Andreas Stiegler, Keshav P. Dahal, Johannes Maucher, Daniel Livingstone, "Symbolic Reasoning for Hearthstone.", IEEE Transactions on Games Volume 10, 2017