Three-Head Neural Network Architecture for Monte Carlo Tree Search

Chao Gao, Martin Müller, Ryan Hayward
University of Alberta

Deep Neural Net Architectures for MCTS

- Move prediction in Go (Clark+Storkey 2015, Maddison et al 2015):
  Single deep convolutional net, “**policy** net”

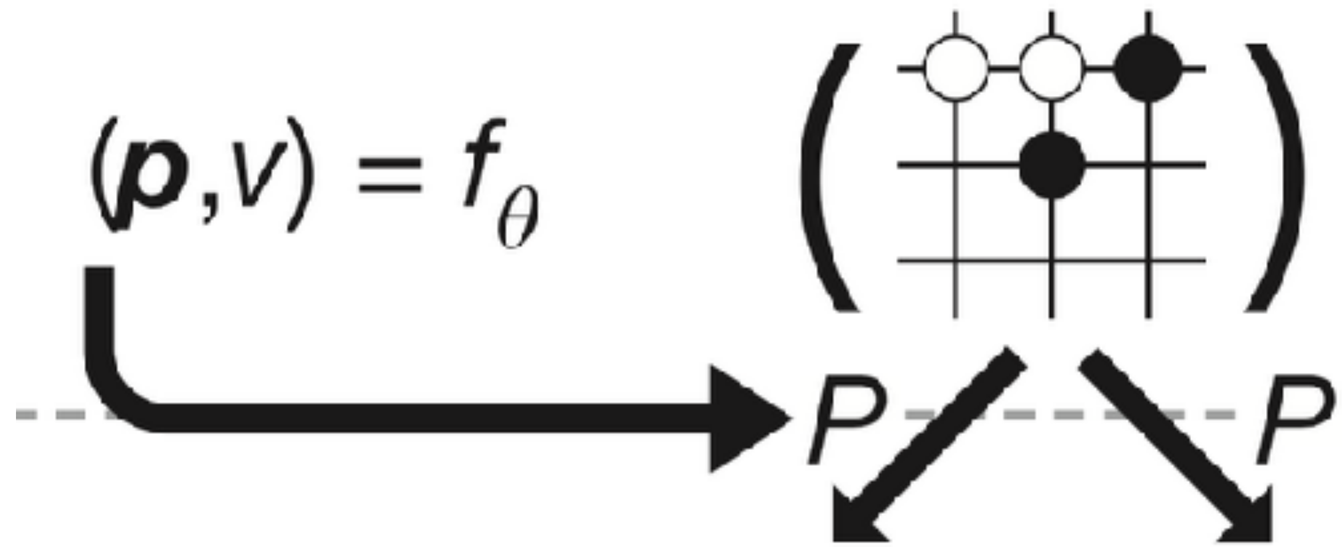
- Early AlphaGo (Silver et al 2016):
  Two separate policy and **value** nets

- AlphaGo Zero (Silver et al 2017), Alpha Zero (Silver et al 2017): Single residual net with **two heads** - policy and value

- In this work: add a **third head**

    - One-step value predictions (Q-values) for all moves

Two Head Architecture



- f .. Deep net
  with parameters $\boldsymbol{\theta}$

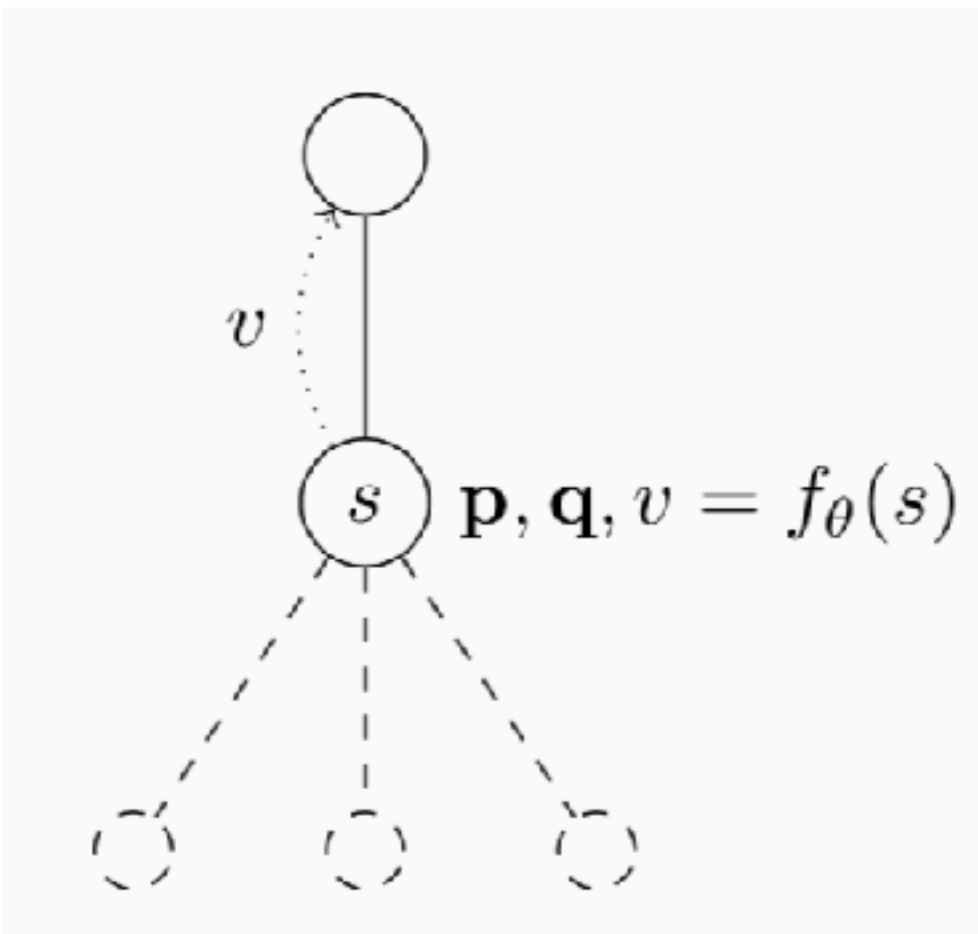
- Input: Go position

- Output (**p**,v)

- **p** = a-priori probability
  of each move being best

- v = evaluation of *current* state

Third Head for Q-Values

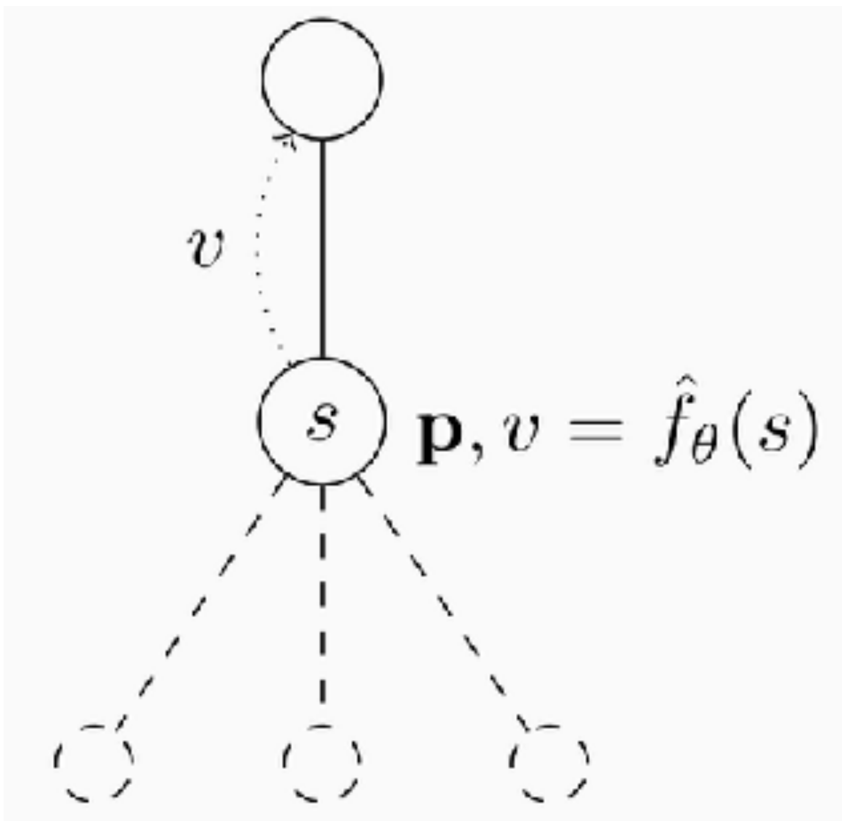


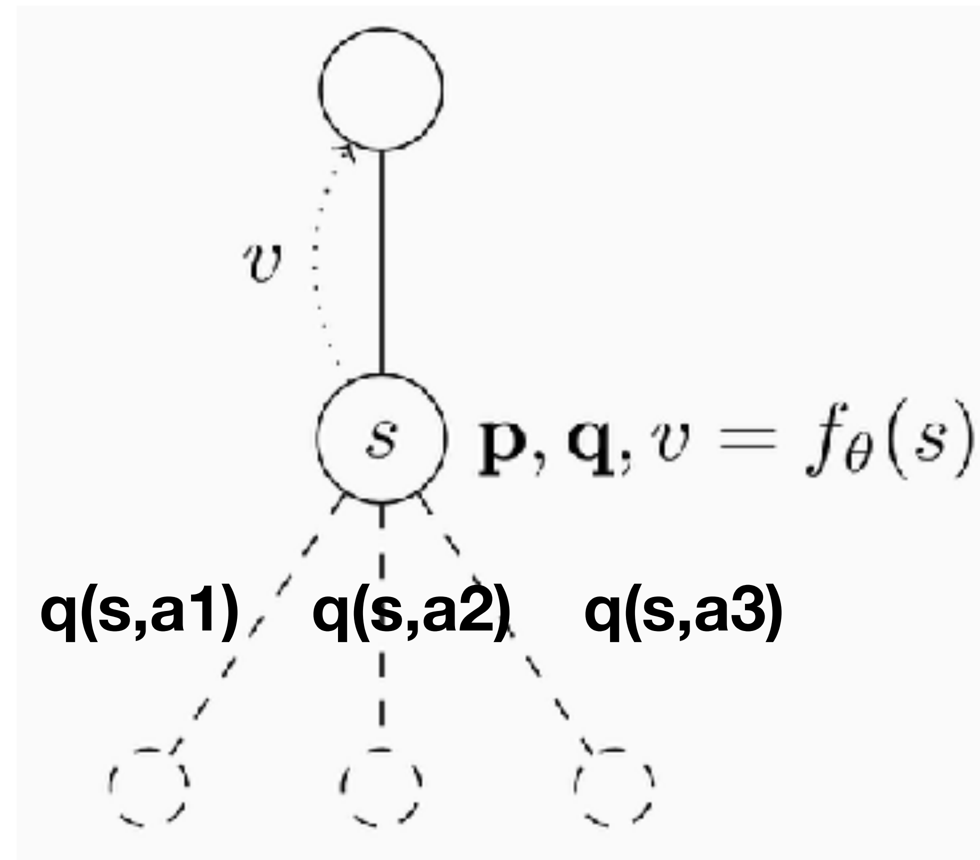
Third output **q(s, a)**:
after-state evaluation
after each legal move a

Main advantage:

- Estimated value of children immediately available…
- …before evaluating them

Use in MCTS





- **2-head**: backup value v of s
- No value estimate of children

- **3-head**: backup value v of s
- Also backup q-value of children

Relation Between v and q

- v .. evaluation from current player's point of view
- **q ..** evaluation from opponent's view
- Best move for us:
  - minimize among all q values
  - negate to change point of view to us
- Minimax consistency:
  $v(s) = - \min q(s,a_i)$
  or $v(s) + \min q(s,a_i) = 0$
- Use for learning consistent v and q estimates

Training of 2 Head Network

- Minimize loss function over labeled training data (s,a,z_s)
  - State s, action a played,
    z_s game result from current player's view
    - $z_s = +1$: win
    - $z_s = -1$: loss
- 2 head loss function (with parameters w and c)

$$\hat{L}(\hat{f}_\theta; \mathcal{D}) = \sum_{(s,a,z_s)\in\mathcal{D}} \left(w(z_s - v(s))^2 - \log p(a|s) + c||\theta||^2 \right)$$

Training of 3 Head Network

- Three changes to loss function:

1. Replace v-loss with average of v- and q-loss

$$(z_s - v(s))^2 \rightarrow \frac{1}{2}(z_s - v(s))^2 + \frac{1}{2}(z_s + q(s,a))^2$$

2. Add AND constraint: if s is loss, all actions lose

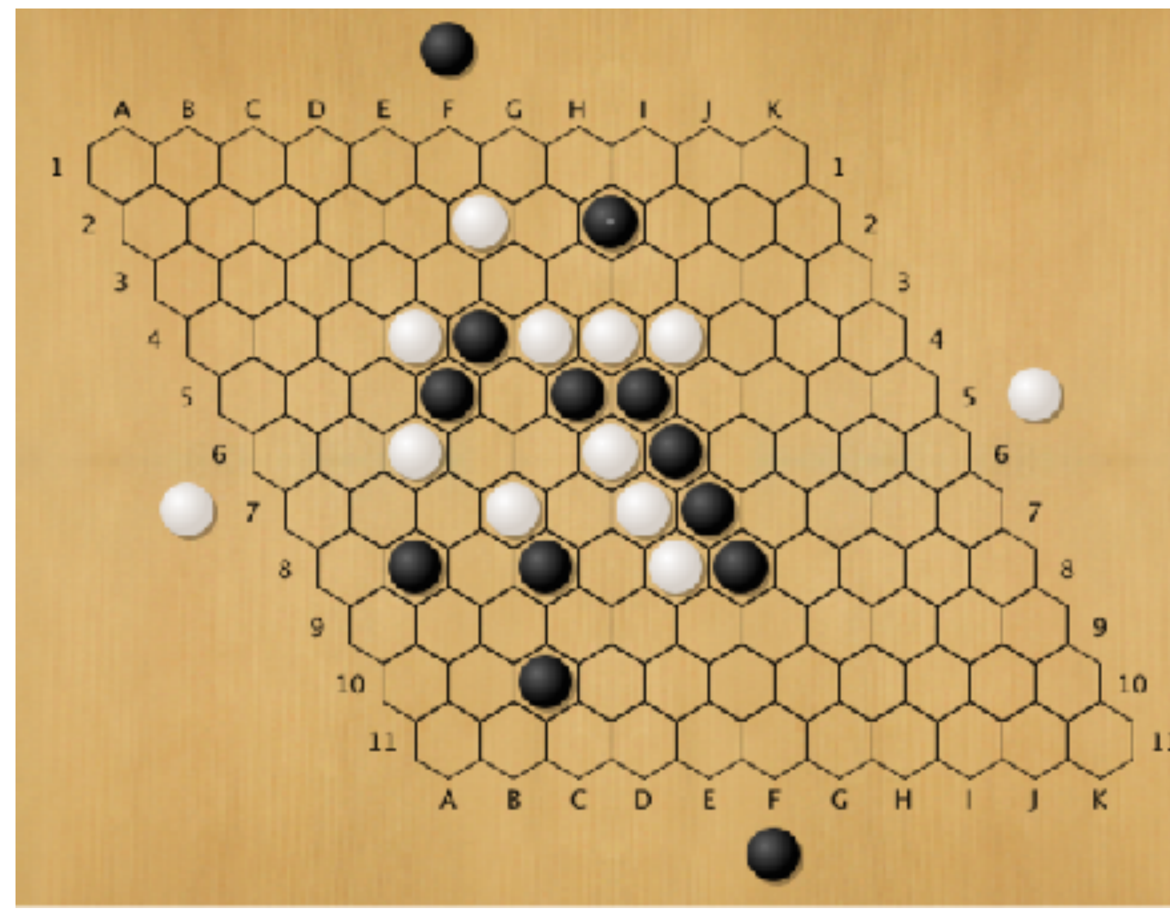
$$L_Q(f_\theta; \mathcal{D}) = \sum_{(s,a,z_s)\in\mathcal{D}} \frac{\max(-z_s, 0)}{|\mathcal{A}(s)|} \sum_{a' \in \mathcal{A}(s)} (z_s + q(s,a'))^2$$

3. Add minimax consistency loss

$$L_P(f_\theta; \mathcal{D}) = \sum_{(s,a,z_s)\in\mathcal{D}} (\min_{a'} q(s,a') + v(s))^2$$

Game of Hex

- Classic abstract board game, invented in 1940's
- Goal: connect your two sides
- Theorem: exactly one player will connect
- Some similarities to Go
    - Simpler rules
    - Deep, difficult game



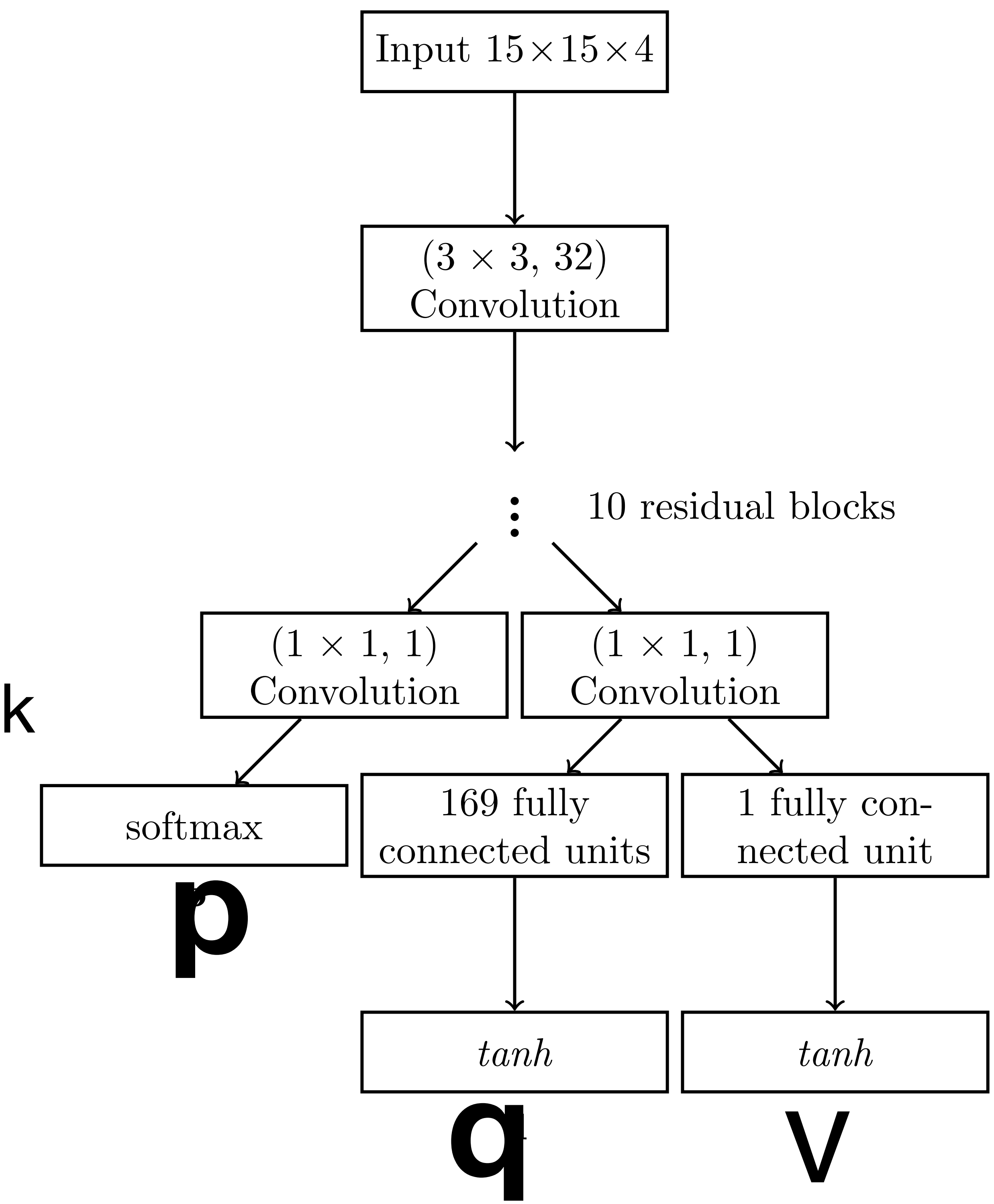
Neural Net for Hex

- Input: 13x13 Hex board padded with extra borders

- Residual net, 10 blocks

- 3 heads for **p**, **q**, v

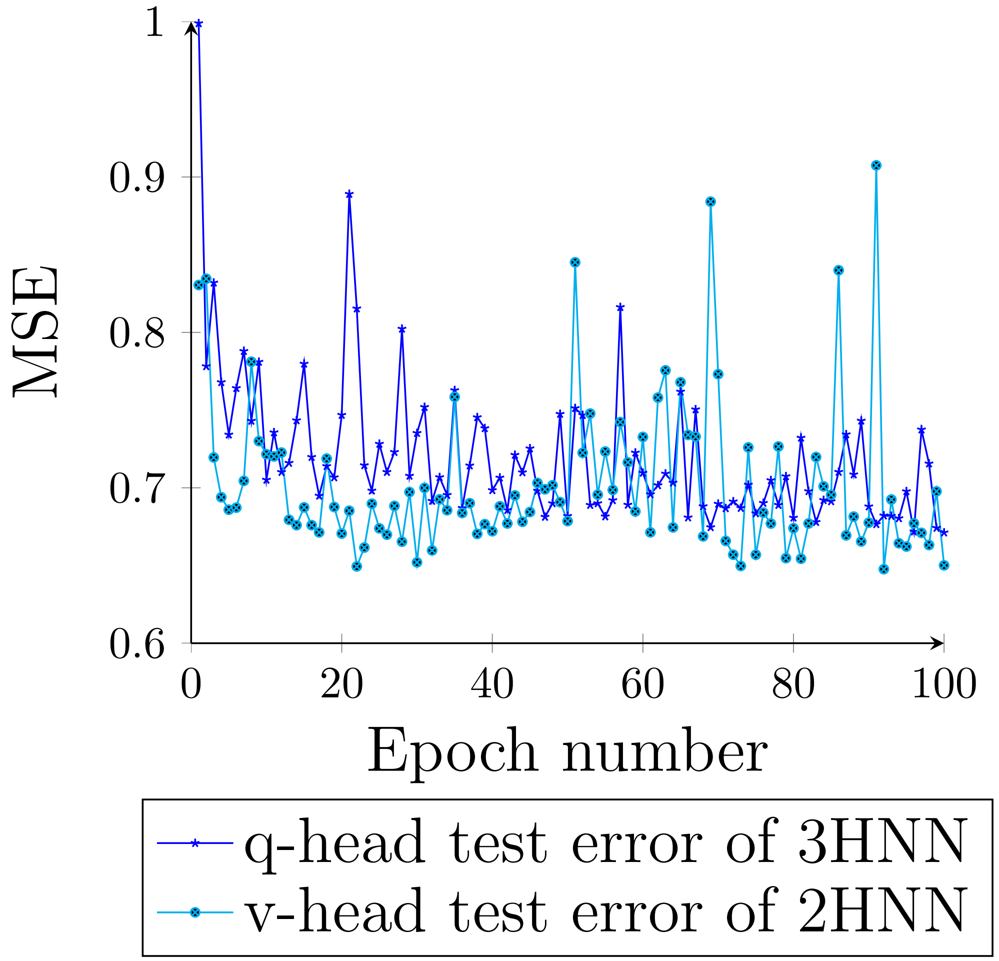
- Compare with 2 head network - without the q head

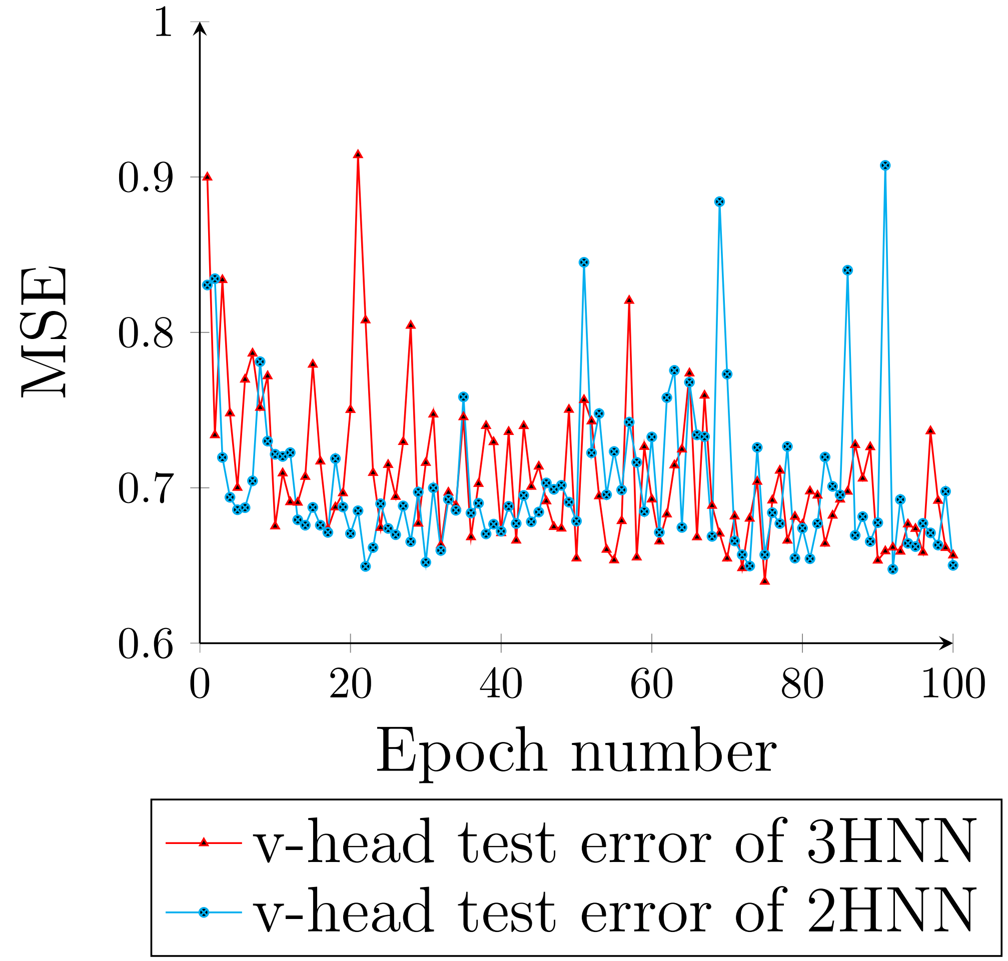


Hex Training Data

- Self-play 13x13 Hex games
  - From previous strongest program MoHex 2.0
  - About 10^6 positions
- Labeled by game outcomes z
- Data augmentation: for lost positions, all actions lose

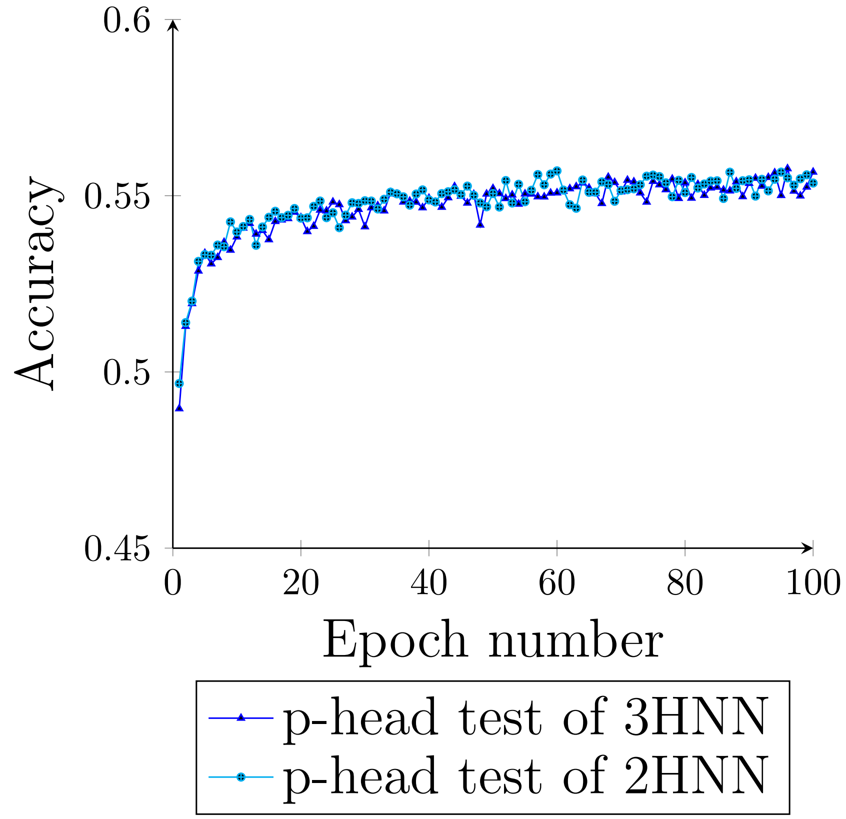
Test Errors 2 vs 3 Heads





- q error comparable to v error - very good news! 1-step predictions as good as direct evaluation
- v errors comparable with 2 and 3 heads

Policy Move Prediction



- Top-1 move prediction of policy head
  - Is the highest probability move the same as in test data?
- Again, 2 and 3 head nets are very similar

Play against Previous Mohex-CNN

Player	Player as black	Player as white	Overall winrate
MoHex-3HNN	76.5%	70.6%	73.5%
MoHex-2HNN threshold 0	65.9%	57.6%	61.8%
MoHex-2HNN default threshold	69.4%	56.5%	62.9%

- Integrated new nets with MoHex' Monte Carlo Tree Search
- Played against last year's MoHexCNN
- Iterate over all opening moves - many are very lopsided
  - 73.5% is a large score in this test

2 vs 3 Heads

- **64.1% wins** for 3 heads
  against strongest version of 2 head

Combining q and v

- Idea: v and q estimate for different
  but *closely related* states
- Use minimax consistency arguments
  - Combine v and q into a single estimate v’
- Two versions of this idea in the paper
- Both win 55-58% against plain v

Advantages of Three Heads

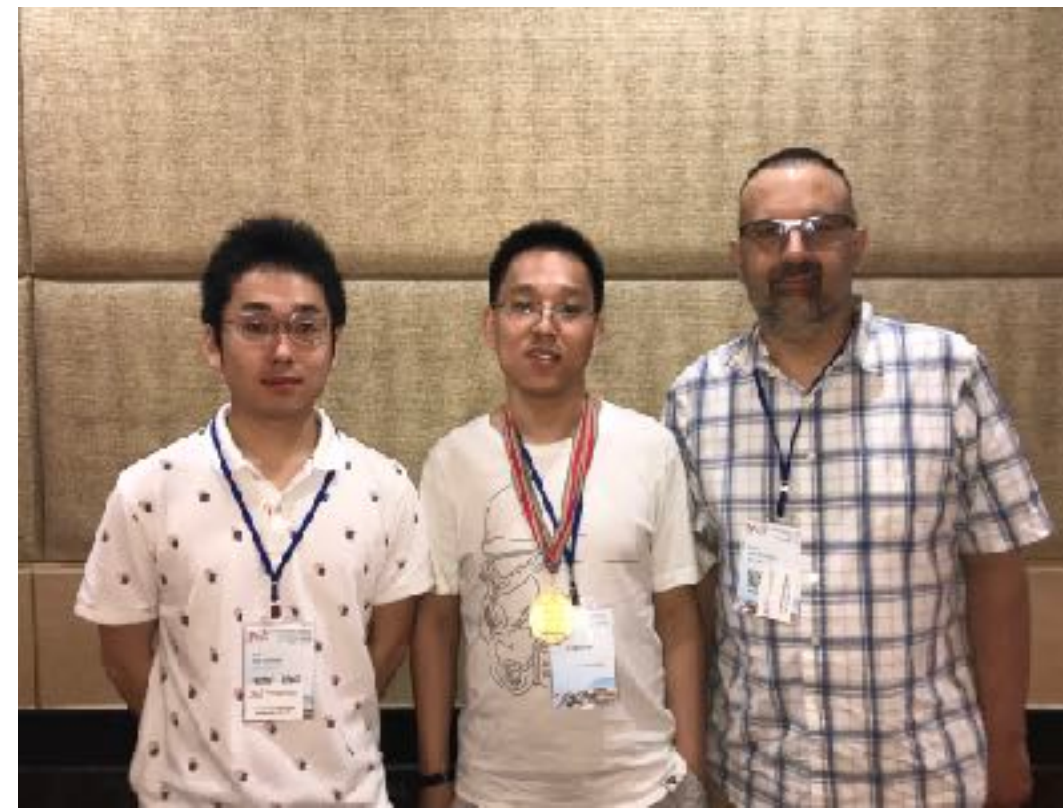
- **Many more state evaluations** *in the same time* due to q-values
- Slightly stronger evaluation by combining v and q
- Some advantages during learning - see paper

Alpha Zero Style Training

- Early result, not in paper
- 3 head architecture also works well
  with Alpha Zero approach
- Continuously improve **p, q**, v by self-play
- Warm start with best version above
- After 400,000 training games, “significantly stronger”
- Why not train from zero knowledge? Practical reasons

2018 Computer Olympiad

- Two strong Hex entries this year, MoHexCNN (Gao/Ualberta), EzoCNN (Takada)
  - MoHexCNN:
    3-head plus Alpha Zero style training
  - EzoCNN: CNN, trained 4-5 months by selfplay, 10 million games
- Both win over 80% against MoHex 2.0
- Two board sizes: 11x11 and 13x13
- MoHexCNN won both matches **5-0**



Kei Takada (Ezo),
Chao Gao (MoHex),
Ryan Hayward (MoHex)

Summary

- 3 head architecture, learns q-values as well
- Game-independent idea, applied to Hex
- More data efficient, sees one step further “for free”
- Also works well with Alpha Zero self-play training
- Far surpasses previous best Hex programs