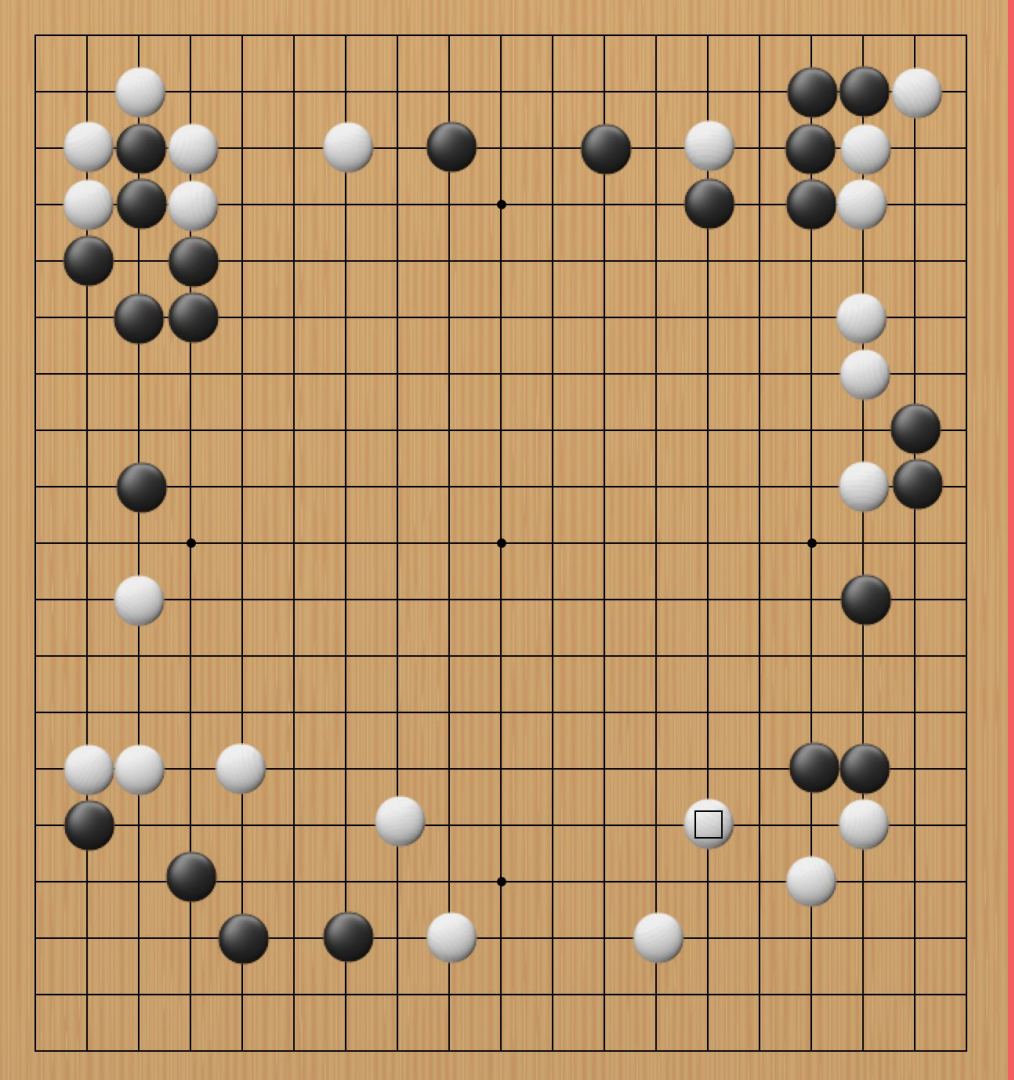


“BetaGo”: Predicting Professional Players’ Moves in Go Game

Molly Zhang

1 minute guide to Go Game

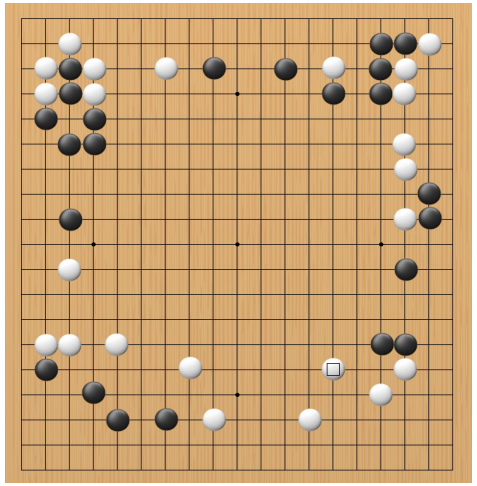
Three basic Rule:

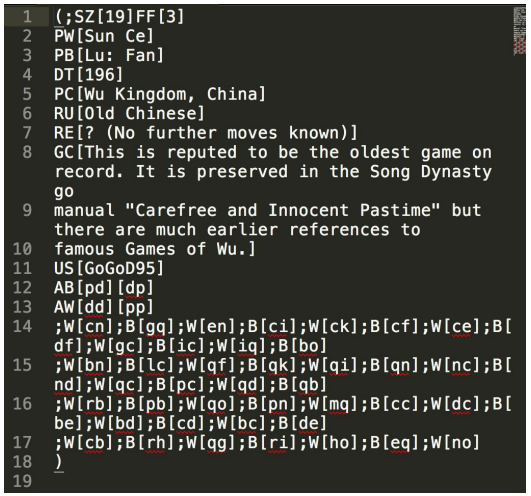
- Black and white stone alternately put a stone on board (at intersection of two grid lines)
- If stone(s) are surrounded by opponent stones, it's removed from board

- Goal of game:
  whichever side that occupied more territory on board wins

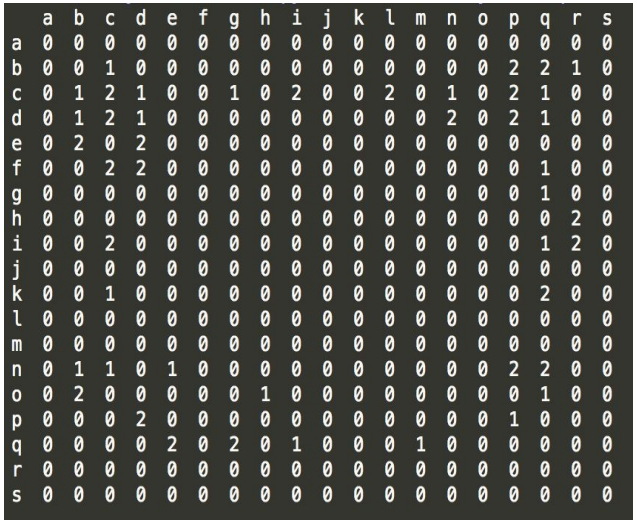
Example: demo game

Parsing Go game into Matrix





```
(;SZ[19]FF[3]
PW[Sun Ce]
PB[Lu: Fan]
DT[196]
PC[Wu Kingdom, China]
RU[Old Chinese]
RE[? (No further moves known)]
GC[This is reputed to be the oldest game on record. It is preserved in the Song Dynasty go
manual "Carefree and Innocent Pastime" but there are much earlier references to famous Games of Wu.]
US[GoGoD95]
AB[pd][dp]
AW[dd][pp]
;W[cn];B[gq];W[en];B[ci];W[ck];B[cf];W[ce];B[df];W[gc];B[ic];W[iq];B[bo]
;W[bn];B[lc];W[qf];B[qk];W[qi];B[qn];W[nc];B[nd];W[qc];B[pc];W[qd];B[qb]
;W[rb];B[pb];W[qo];B[pn];W[mq];B[cc];W[dc];B[be];W[bd];B[cd];W[bc];B[de]
;W[cb];B[rh];W[qg];B[ri];W[ho];B[eq];W[no]
)
```



	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	2	1	0
c	0	1	2	1	0	0	1	0	2	0	0	2	0	1	0	2	1	0	0
d	0	1	2	1	0	0	0	0	0	0	0	0	0	2	0	2	1	0	0
e	0	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	0	0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
g	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
h	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
i	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0
j	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
k	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
l	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
n	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	2	2	0	0
o	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
p	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
q	0	0	0	0	2	0	2	0	1	0	0	0	1	0	0	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Go game board

Game recorded in “SGF” format

Board as 19x19 matrix

n number of matrices from each game, n = total_steps_in_game

Data Summary

- Downloaded ~86,000 professional games in .sgf format from gogodonline.co.uk (for $15...)
- Each game as on average 200 moves, therefore giving a total of 200 x 86000 = 17 million board positions and corresponding moves

Question

- Given a board, with my stones and opponent stones on it, where should I go next?

Overall Results:

- Best overall prediction accuracy over entire game is ~4%
- V.S. Accuracy of random guess (~0.42%)
- V.S. Accuracy of google AlphaGo (~50%)
  - What google DeepMind did and I didn't:

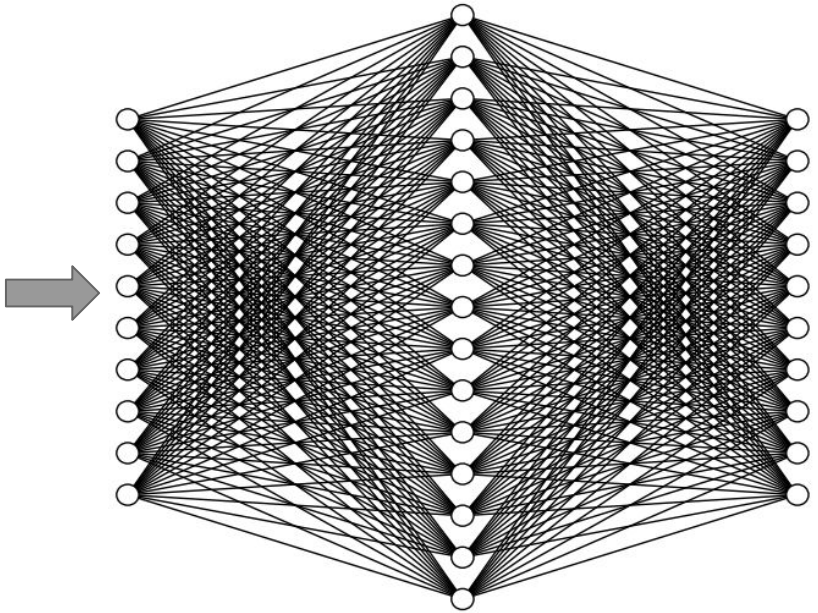
	BetaGo	AlphaGo
Training Size	1 million	30 million
Input layers (feature layer)	1	48
Conv Layers	1	many
Number of GPU and CPU	1 and 8	280 and 1920
Level of developer smartness	Okay	Very High

- However, my prediction accuracy of first 10-20 moves are high (~40%)

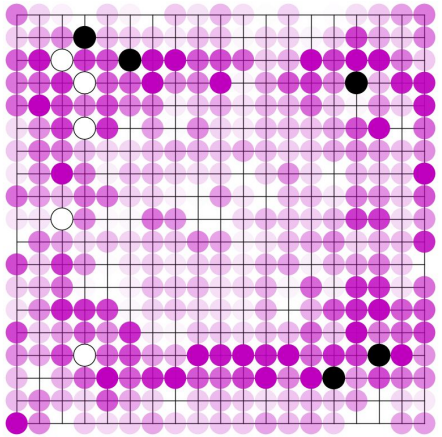
Neural Net Models 1:
One input layer, one hidden layer (ReLu activation) and one softmax output layer

Input:
19 x 19 Board
position matrix

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	2	1	0
c	0	1	2	1	0	0	1	0	2	0	0	2	0	1	0	2	1	0	0
d	0	1	2	1	0	0	0	0	0	0	0	0	0	2	0	2	1	0	0
e	0	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
f	0	0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
g	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
h	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
i	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	0
j	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
k	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
l	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
m	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
n	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	2	2	0	0
o	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
p	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
q	0	0	0	0	2	0	2	0	1	0	0	0	1	0	0	0	0	0	0
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



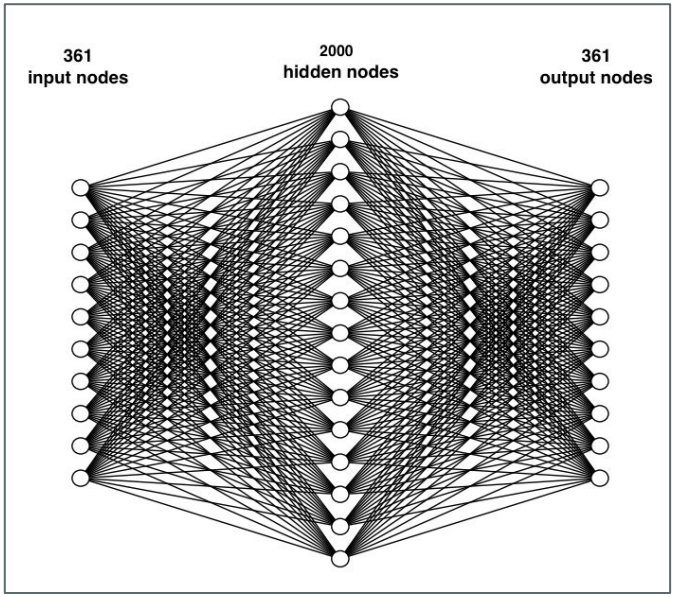
Output:
Probability distribution
of next move



Neural Net Models 1:
Basic NN with one hidden layer and one softmax layer before output

Experimented with:

- Different number of hidden nodes
- Accuracy decay with more stones on board
- Predicting accuracy scaling with increasing training data
- Regularization by dropping out
- Activation function: Sigmoid v.s. Rectifier

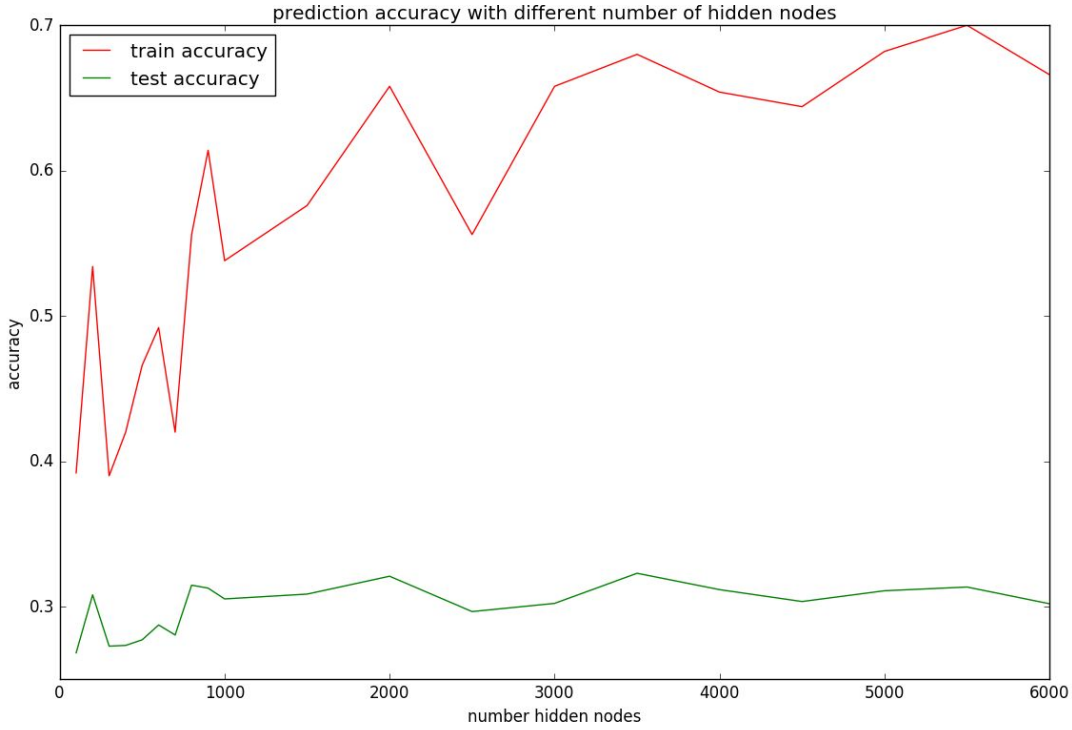


Neural Net Models 1: hidden nodes on test accuracy and overfitting

Take away:

- Lots of overfitting
- Best hidden nodes number: ~2000
- Tensorflow limits max node number to 6000

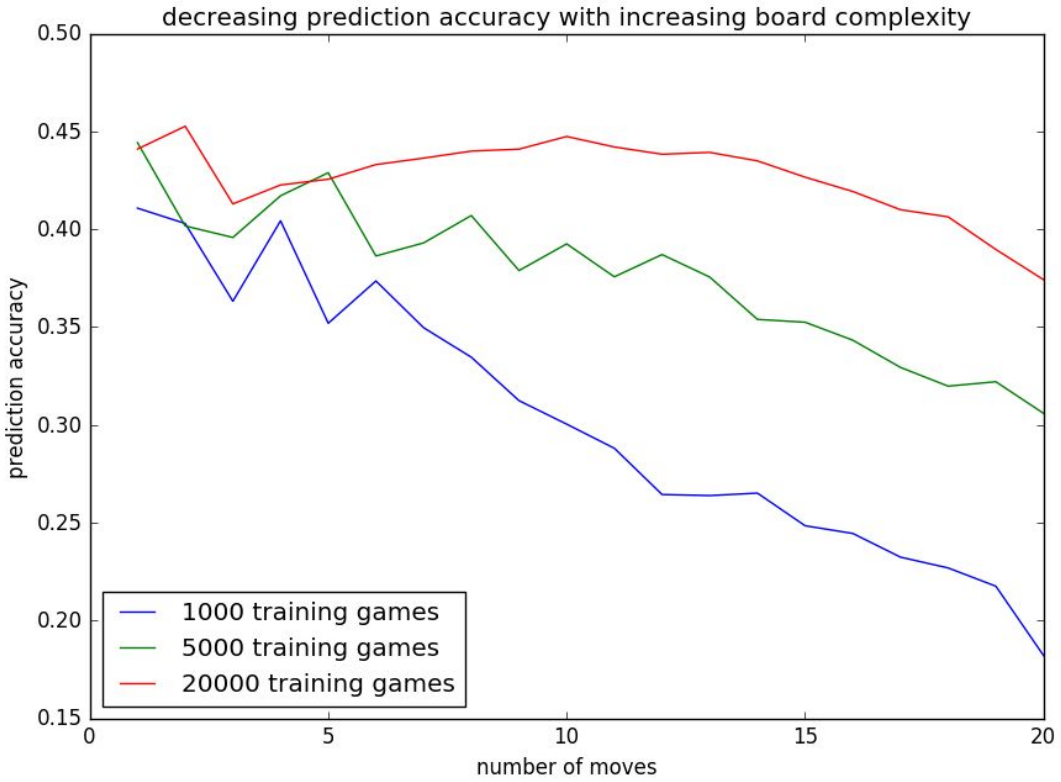
Note:
Only 1000 training examples are used for speedy experiment (thus the lower accuracy and the high overfitting)



Neural Net Models 1: Accuracy Decay with number stones on board

Take away:

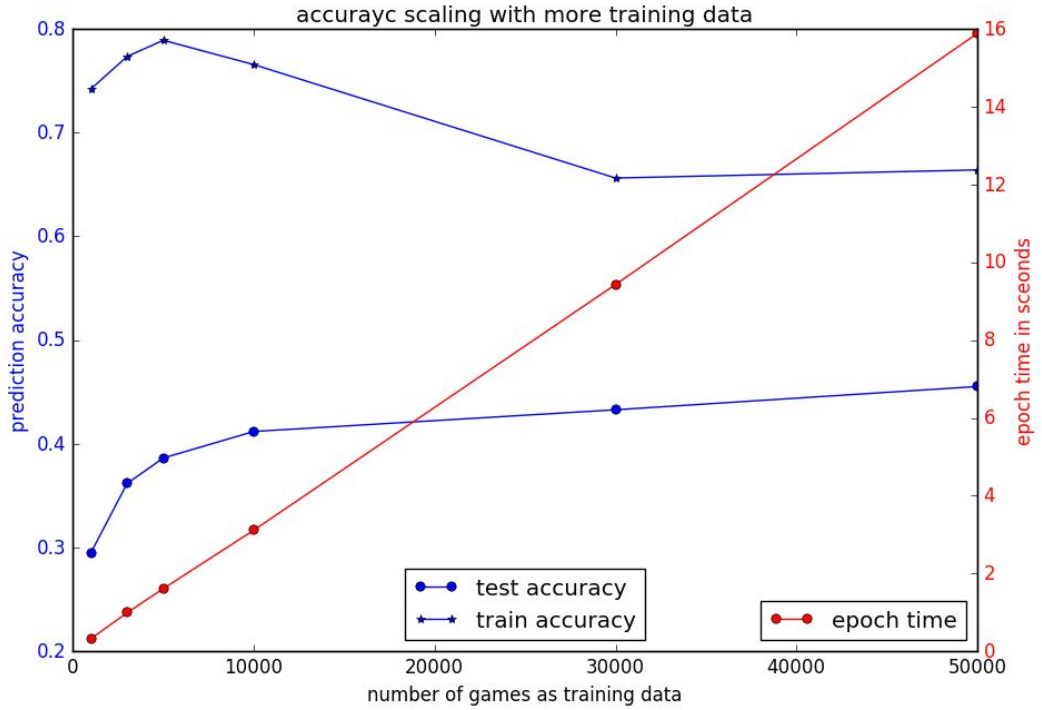
- As game progresses, it gets harder to predict
- More training data makes the accuracy drop slower



Neural Net Models 1:
Accuracy improvement and training slowdown with more training data

10,000 games is sweet trade-off between:

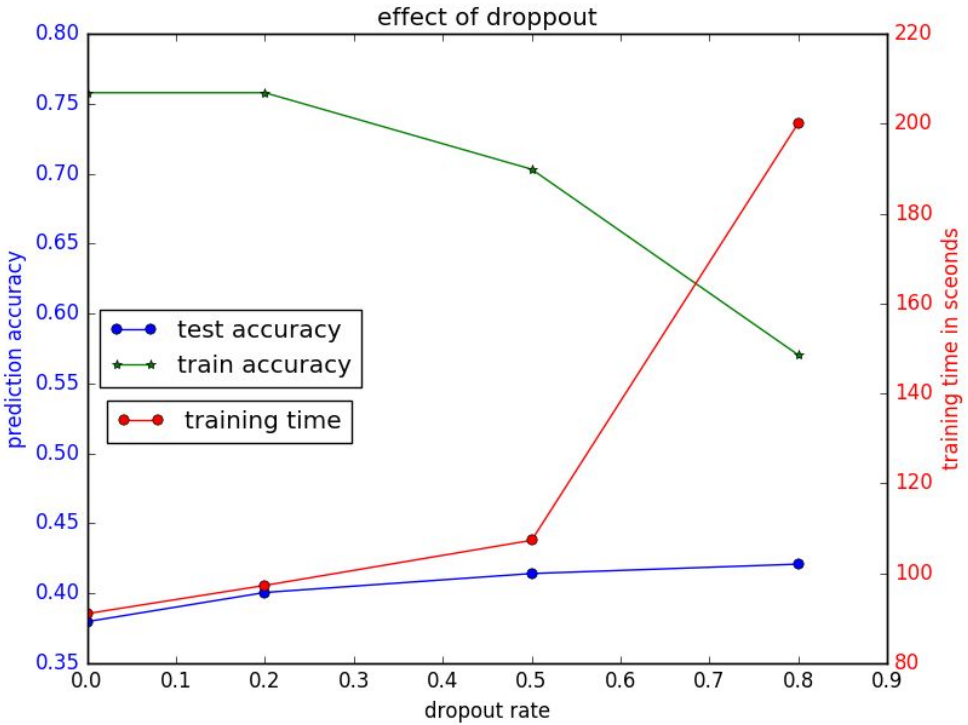
- Higher test accuracy
- Speedy epoch time



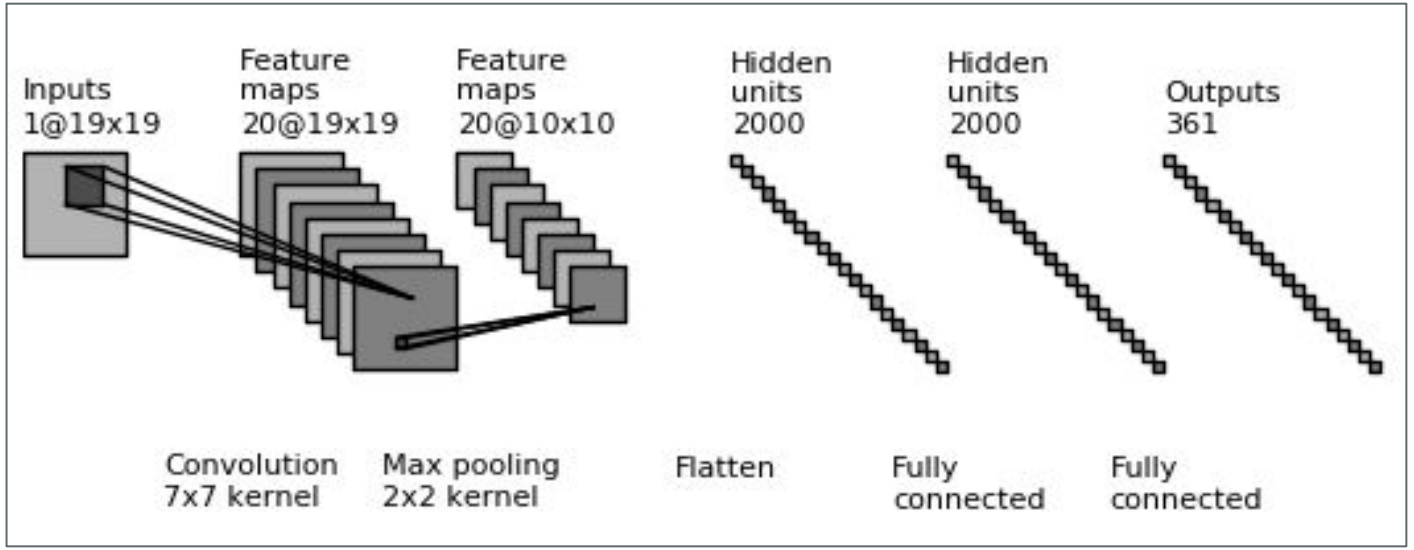
Neural Net Models 1: Effect of regularization by dropping out

Take away

- Dropout does reduce overfitting
- Dropout increase training time
- Dropout improves test accuracy
- Picking dropout rate = 0.5



Neural Net Models 2:
convolutional layer + fully connected layer + softmax output

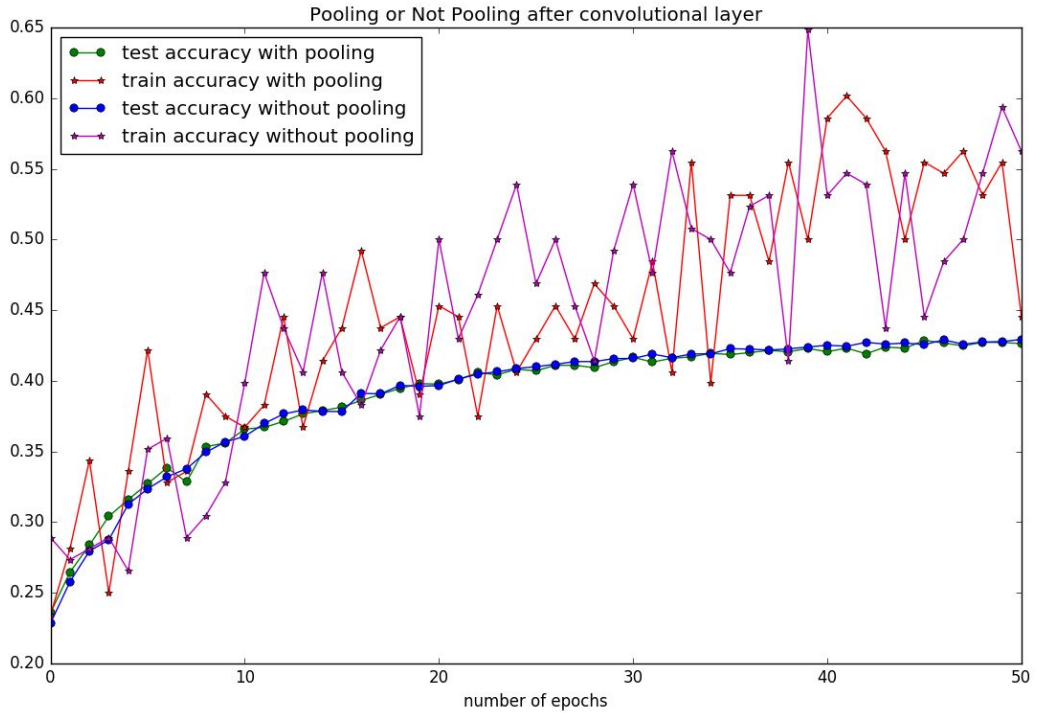


Result:
Accuracy: 43.3% (comparable to without convolutional layer)
Epoch time: 13 seconds (twice of NN model 1)

Neural Net Models 2: Pooling, or No Pooling?

Take away

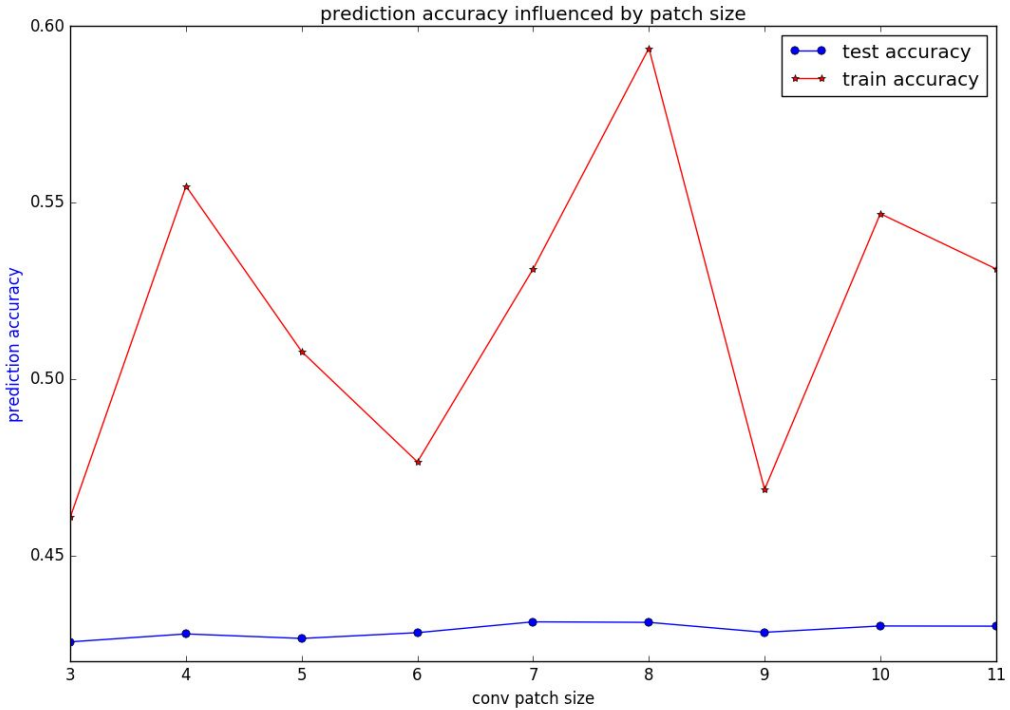
- Pooling reduces overfitting, by a little bit
- No difference in test accuracy
- Picking 2x2 max pooling because it reduces training time (and model complexity)



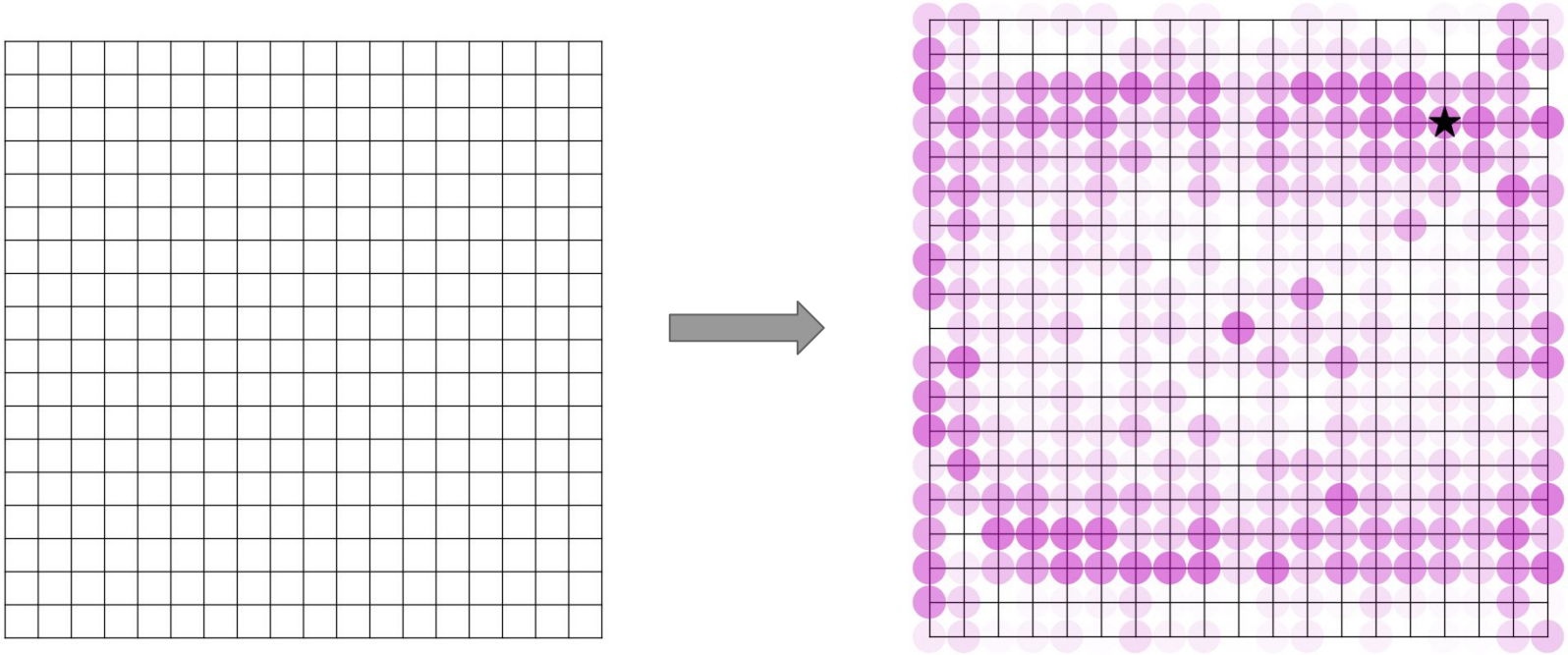
Neural Net Models 2: What's the right convolutional patch size?

Take away

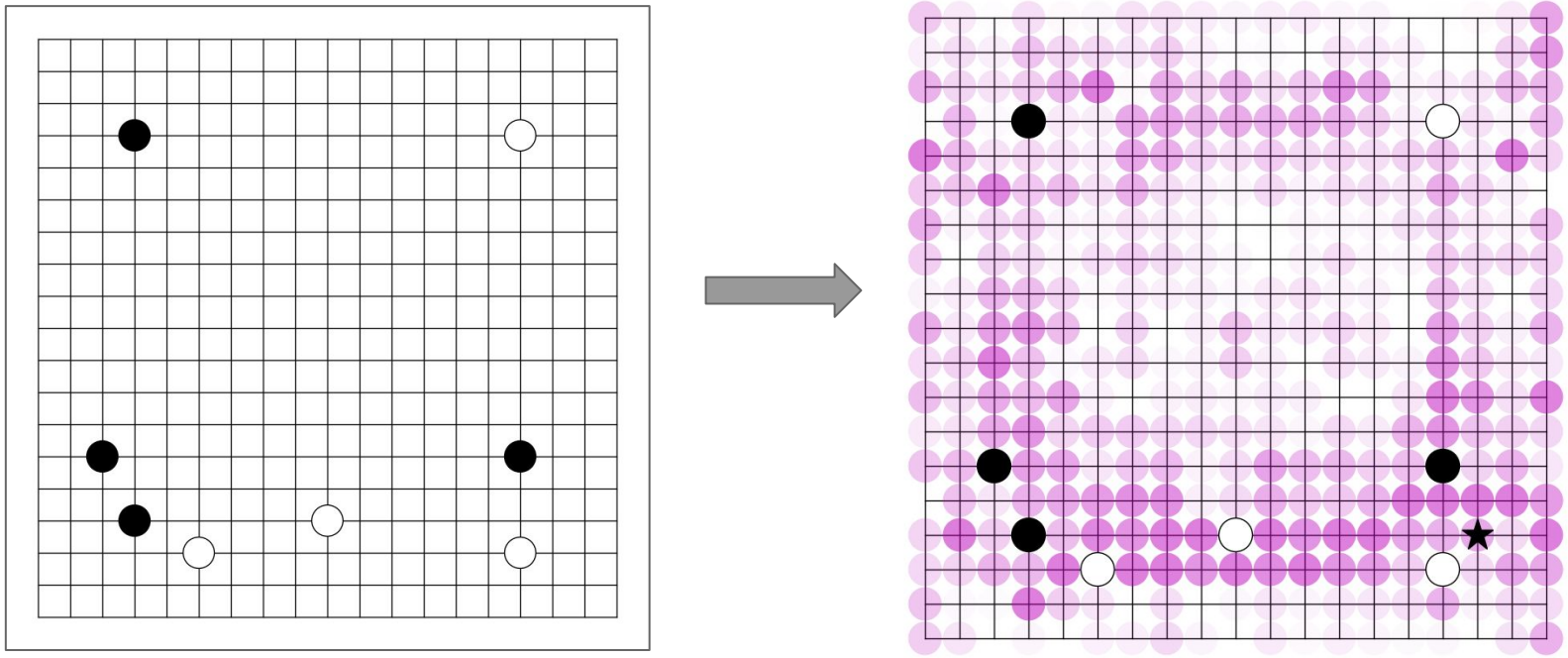
- Patch size of 7 or 8 is best
- Matches gameplay intuition



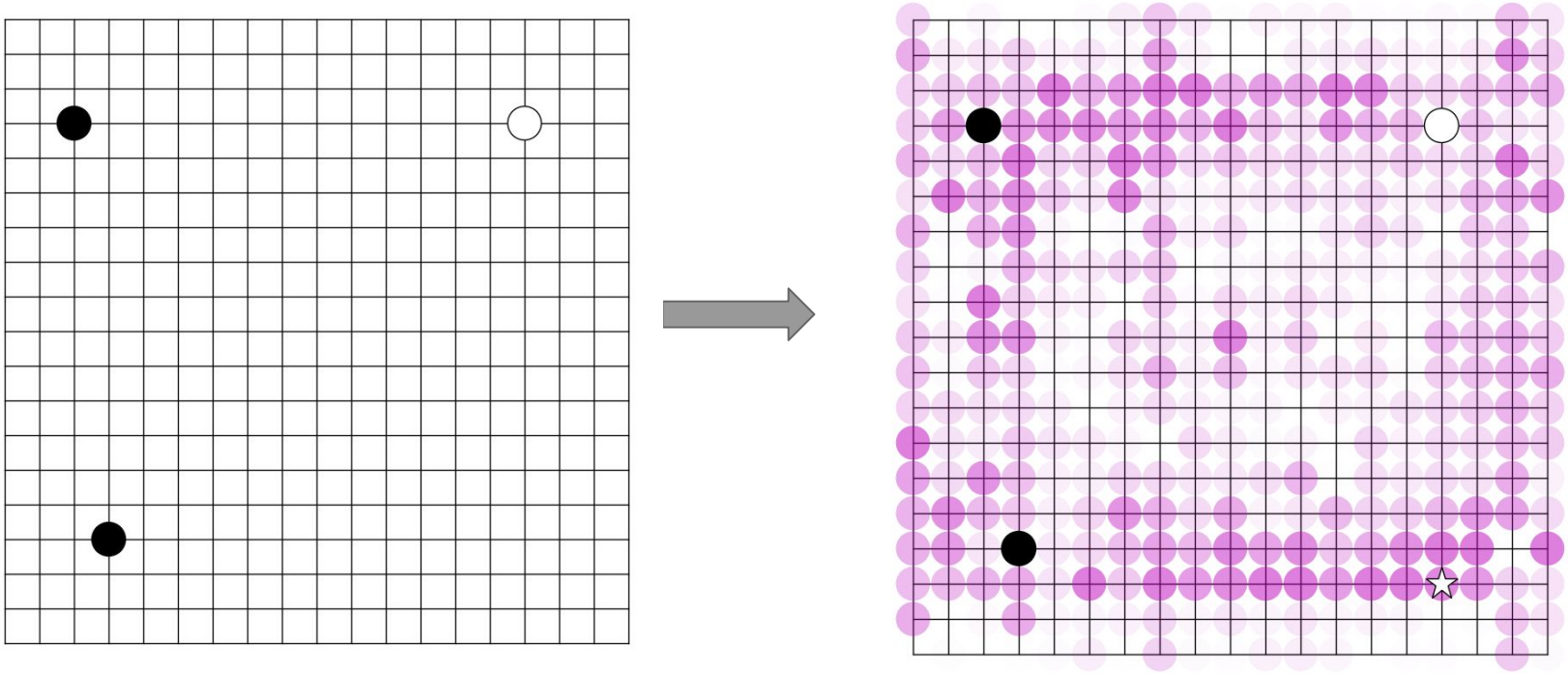
Visualize the Predictions in Game Play (o)



Visualize the Predictions in Game Play (1)



Visualize the Predictions in Game Play (3)



Tools/Resources used

- Tensorflow (GPU version)
- Hyades (the supercomputing cluster in UCSC)

References:

- Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." Nature 529.7587 (2016): 484-489.
- Clark, Christopher, and Amos Storkey. "Teaching deep convolutional neural networks to play go." arXiv preprint arXiv:1412.3409 (2014).

Thanks

- David Parks for Inspiration, encouragement and discussions
- Keshav Mathur for GPU, cuda, batch loading etc help
- Dimitris Achlioptas for the class