Evolving Non-linear Stacking Ensembles for Prediction of Go Player Attributes

Josef Moudřík[1] Roman Neruda[2]

[1]Charles University in Prague
Faculty of Mathematics and Physics
J.Moudrik@gmail.com

[2]Institute of Computer Science
Academy of Sciences of the Czech Republic
roman@cs.cas.cz

IEEE SSCI 2015

- "Oldest" game in the world,
- perfect information, deterministic rules,
- board size of 19×19 intersections,
- two players — Black and White,
- goal (roughly): enclose more territory.

- **AI is hard:**
- high branching factor,
- no clear evaluation function.

Introduction: Motivation and Goals

- Large collections of Go game records available online.
- Traditionally (computer-wise) used for:
  - Opening dictionaries,
  - learning domain-aware heuristics, e.g. [Coulom, 2007],
  - to train predictive models, e.g. CNN [Clark and Storkey, 2014].
- **Our Goal:** Predict player attributes (such as strength & style) from a set of games.
- **Previous work:** [Moudrik et al., 2015]

  Player's Games $\xrightarrow{\text{feature extraction}}$ Dataset
- **This work:**

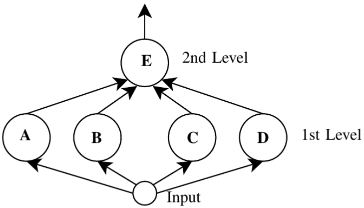
  Dataset $\xrightarrow{\text{learning}}$ Predictive Model

- **Learning:** Given model M, find parameters Θ that maximize accuracy (on some data).
- **Ensemble:** The model M is composed of multiple sub-models m_i (and params Θ_i), with a strategy for training and combining the results.

  Some Examples: bagging, boosting, stacking, dropout.
- **Why should ensembles be interesting?** Efficient combination can mitigate individual model's weaknesses and combine their strenghts.
- **In Practise,** ensembles often improve accuracy and robustness of models.

Methodology: Stacking

- Two-level hierarchical model
- Diverse 1st level models (**A** – **D**)
- 2nd level model (**E**) aggregates outputs from 1st level
- **Training strategy:**
  Internal Cross-validation
- **E** should learn how to optimally combine **A–D** predictions.



- So far, only linear models have been used as 2nd level model.[1]
- **An Engineering Question:** When solving a task, how to choose the best combination of **E** and **A–D** learners?

[1] To our best knowledge.

- Let us have a set of **Base Learners** BL.
- **Individual encoding:** $(I, Folds, \vec{v})$
  $I \in [1..|BL|], Folds \in [2..6], \vec{v} \in 2^{|BL|}$
- **Mutation 1:** changes I or $Folds$ to any correct value.
- **Mutation 2:** flips one random bit in $\vec{v}$.
- **Crossover:** random crossover of $\vec{v}_{mother}$ and $\vec{v}_{father}$.[2]
- **Fitness:** $1/RMSE$ of the resulting stacking ensemble.

[2] In each generation, two parents form two offspring, one gets $(I, Folds)$ from father, one from mother; $\vec{v}$'s are given by the crossover.

Methodology: Base Learners

And their parameterisations

Base learner	**Settings**
Mean regression	—
PLS regression	$l \in \{2, \ldots, 10\}$
k-nearest nb.	$k \in \{10, 20, \ldots, 60\}$, $\alpha \in \{10, 20\}$, $\delta \in \{\text{Manhattan}, \text{Euclidean}\}$, all combinations.
Random Forests	$N \in \{5, 10, 25, 50, 100, 200\}$
Neural network	$max \in \{50, 100, 200, 500\}$ iterations $\epsilon \in \{0.001, 0.005\}$, 1 layer with number of neurons $\in \{10, 20\}$, all combinations. Symmetric sigmoid activation function.
Bagged Neural Networks	For ensemble sizes of $\in \{20, 40, 60\}$, each Neural network (from right above) was tested.

Experiments: Go — Strength

Dataset and setup

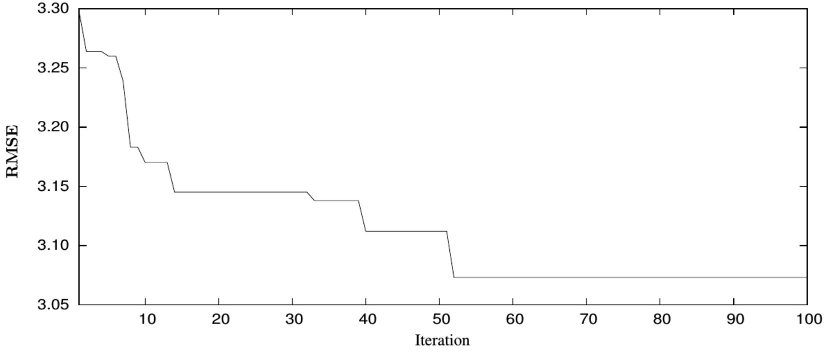
- Precisely defined in [Moudrik et al., 2015].
- Data from 100 000 games from KGS [Shubert, 2013] were divided by 26 ranks in Go 20 kyu – 1 kyu, 1 dan – 6 dan.
- 26×120 pairs $(x, y), |x| = 1040, y \in [1 \ldots 26]$.
- Population was initialized by best hand-tuned learner.

GA Parameter	**Value**
Population size S	16
Elite size E	1
Max number of iterations	100
Probability of **Mutation 1**	0.2
Probability of **Mutation 2**	0.5
Fitness function	$1/RMSE$, see[3]

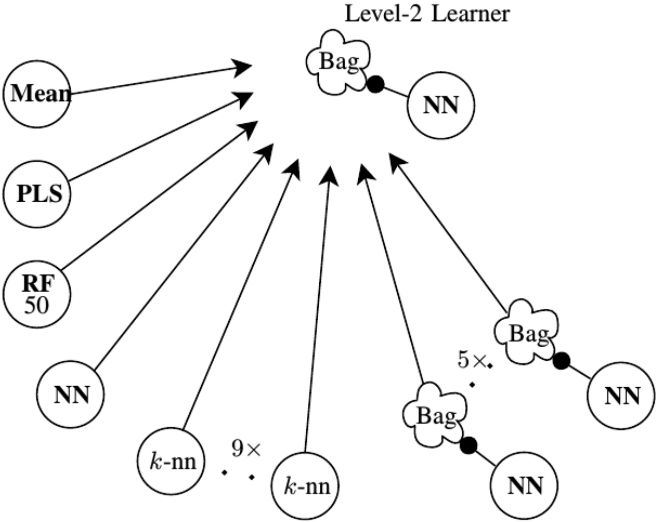
[3] Computed using 5-fold CV on a sub-sampled dataset.

Experiments: Go — Strength

Results, fitness evolution and comparison



Learner	**RMSE**	**Mean cmp**
Mean regression	7.507	1.00
Random Forrest	3.869	1.94
PLS	3.176	2.36
Bagged NN	2.66	2.82
Hand-tuned learner	2.635	2.85
Best GA stacking ensemble	2.607	2.88



Level-2 Learner
Bag
NN
Mean
PLS
RF
50
NN
k-nn
9×
k-nn
Bag
NN
5×
Bag
NN

Experiments: Go — Strength

Results, best individual

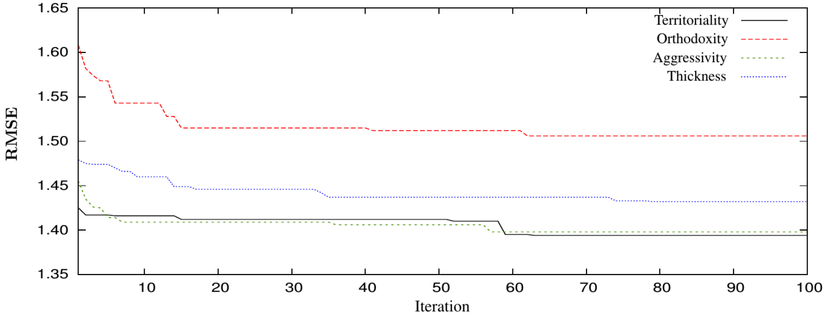
Ensemble I.	**Settings**
Stacking	6 folds, level 2 learner: Bagged ($20\times$) NN: $\epsilon = 0.005$, $max = 500$ iter., 1 layer , 10 neurons.
Base I.	**Settings**
Mean regression	—
PLS regression	$l = 3$
Random Forests	$N = 50$
Neural network	$\epsilon = 0.001$, $max = 200$ iter., 1 layer, 20 neurons.
k-nn	$k = 20$, $\alpha = 20$, $\delta =$ Euclidean.
k-nn	$k = 40$, $\alpha = 10$, $\delta =$ Manhattan, Euclidean.
k-nn	$k = 40$, $\alpha = 20$, $\delta =$ Euclidean.
k-nn	$k = 50$, $\alpha = 10$, $\delta =$ Manhattan.
k-nn	$k = 50$, $\alpha = 20$, $\delta =$ Manhattan, Euclidean.
k-nn	$k = 60$, $\alpha = 10$, $\delta =$ Euclidean.
k-nn	$k = 60$, $\alpha = 20$, $\delta =$ Euclidean.
Bagged NN	$20 \times$ NN: $\epsilon = 0.001$, $max = 100$, 1 layer, 10 neur.
Bagged NN	$40 \times$ NN: $\epsilon = 0.005$, $max = 100$, 1 layer, 10 neur.
Bagged NN	$40 \times$ NN: $\epsilon = 0.001$, $max = 500$, 1 layer, 20 neur.
Bagged NN	$20 \times$ NN: $\epsilon = 0.005$, $max = 200$, 1 layer, 20 neur.
Bagged NN	$40 \times$ NN: $\epsilon = 0.005$, $max = 500$, 1 layer, 20 neur.

Experiments: Go — Style
Dataset and setup

- Precisely defined in [Moudrik et al., 2015].
- Professional games from the GoGoD Database [Hall and Fairbairn, 2011].
- 24 profesionals, each assessed on 4 scales by playing style.
- 24×12 pairs $(x, y), |x| = 640, y \in [1 \ldots 10]$.
- Population was initialized by best hand-tuned learner.

Parameter	**Value**
Population size S	10
Elite size E	1
Number of iterations Max	100
Probability of **Mutation 1**	0.2
Probability of **Mutation 2**	0.5
Ensemble size limit	5

Style	**1**	**10**
Territoriality	Moyo	Territory
Orthodoxity	Classic	Novel
Aggressivity	Calm	Fighting
Thickness	Safe	Shinogi



	RMSE	
Learner	Territoriality	Orthodoxity
Mean regression	2.403	2.421
Hand tuned learner	1.434	1.636
The best GA learner	1.394	1.506
Learner	Aggressivity	Thickness
Mean regression	2.179	1.682
Hand tuned learner	1.423	1.484
The best GA learner	1.398	1.432

Conclusion

- We shown an algorithm for **evolving non-linear stacking ensembles**.
- Algorithm forms complex diverse ensembles of learners, which
- give **substantial improvements** for **prediction of Go player attributes**.
- One disadvantage is that computing fitness takes quite some time (nested CV) — parallelize!
- Feature extraction and the prediction model $\rightarrow$ Online Learning Tool: `http://gostyle.j2m.cz`

References I

Clark, C. and Storkey, A. (2014). Teaching deep convolutional neural networks to play go. *arXiv preprint arXiv:1412.3409*.

Coulom, R. (2007). Computing Elo Ratings of Move Patterns in the Game of Go. In van den Herik, H. J., Mark Winands, Jos Uiterwijk, and Maarten Schadd, editors, *Computer Games Workshop*, Amsterdam Pays-Bas.

Hall, T. M. and Fairbairn, J. (winter 2011). Games of Go on Disk — GoGoD Encyclopaedia and Database.

Moudrik, J., Baudis, P., and Neruda, R. (2015). Evaluating go game records for prediction of player attributes. In *Computational Intelligence and Games (CIG), 2015 IEEE Conference on*, pages 162–168.

Shubert, W. (2013). KGS — kiseido go server.

Appendix, Best Hand Tuned Ensemble

Ensemble learner	**Settings**
Stacking	4 folds, level 2 learner: NN, $\epsilon = 0.005$, $max = 100$ iter., 1 layer, 10 neurons.
Base learners	**Settings**
Mean regression	—
PLS regression	$l = 3$
k-NN	$k = 50$, $\alpha = 20$, $\delta =$ Manhattan.
Random Forests	$N = 50$
Bagged NN	$20 \times$ NN: $\epsilon = 0.001$, $max = 100$ iter., 1 layer, 10 neurons.