Bagging Ensemble Selection

Quan Sun and Bernhard Pfahringer

The University of Waikato, New Zealand

December 8, 2011

Outline

Ensemble learning

Let x be an instance and $m_i, i = 1...k$, a set of base classifiers that output probability distributions $m_i(x, c_j)$ for each class label $c_j, j = 1...n$. The output of the final classifier ensemble $y(x)$ for x can be expressed as:

$$y(x) = \arg\max_{c_j} \sum_{i=1}^{k} w_i m_i(x, c_j), \tag{1}$$

where w_i is the weight of base classifier m_i.

Ensemble learning strategies can be seen as methods for calculating optimal weights for each base classifier in terms of a classification goal.

Popular ensemble learning algorithms

- Stacking/Blending (Wolpert 1992)
- AdaBoost (Freund and Schapire 1995)
- Bagging for supervised learning (Breiman 1996)
- RandomForest (Ho 1995; Breiman 2001)
- **EnsembleSelection** (Rich Caruana, et al, "Ensemble Selection from Libraries of Models", ICML '04)

The EnsembleSelection (ES) algorithm

EnsembleSelection(A, L, F, E, r**)**

A are the base classifiers for building the model library
L is the model library
F is the training set
E is the ensemble
r is the hillclimb set ratio

1. $T \leftarrow RandomSample(F, r)$ // T is the build set
2. $H \leftarrow F - T$ // H is the hillclimb set
3. $E \leftarrow$ empty ensemble
4. $L \leftarrow BuildModelLibrary(A, T)$ // build model library on T
5. Assign weight to base model; weights can be calculated by using forward stepwise selection guided by performance on the hillclimb set H (Caruana2004)

Return $E \leftarrow$ subset of base classifiers in L with greater than zero weight

---

Rich Caruana, et al, "Ensemble Selection from Libraries of Models" (ICML '04)

Overfitting and the hillclimb ratio

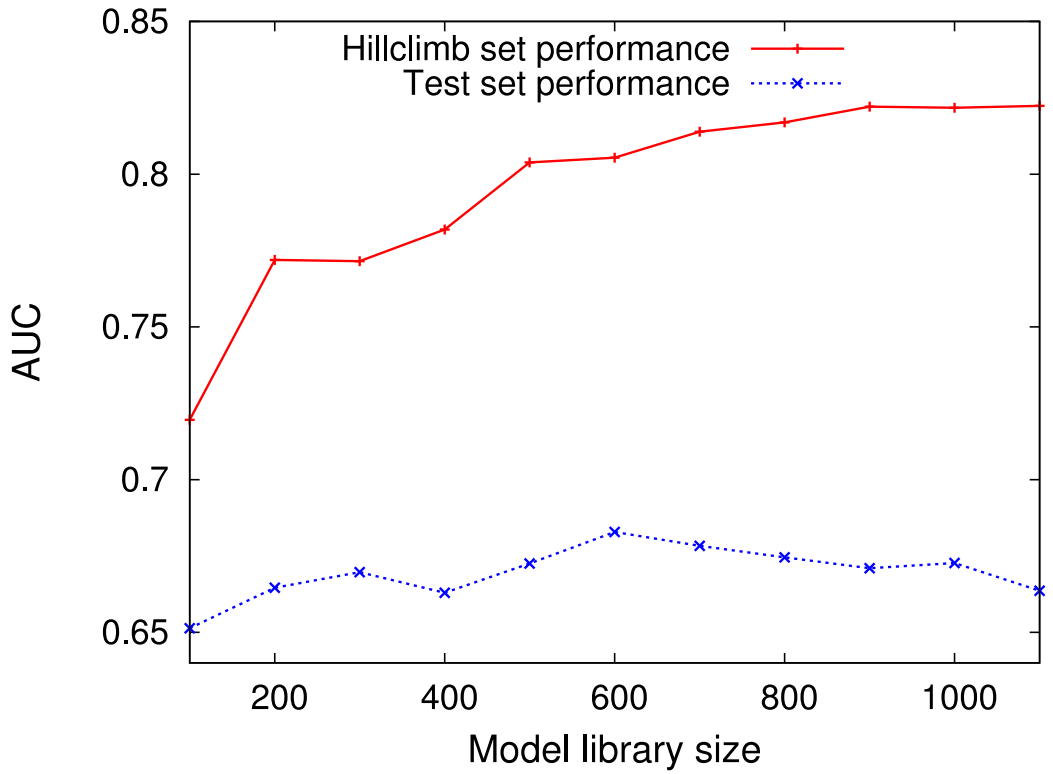
ES is superior to many other well-known ensemble algorithms (Caruana2004).

ES has been highlighted in winning solutions of:

Netflix 07, KDD Cup 09, UCSD/FICO contest 10, Kaggle - The "predict grant applications" competition 11

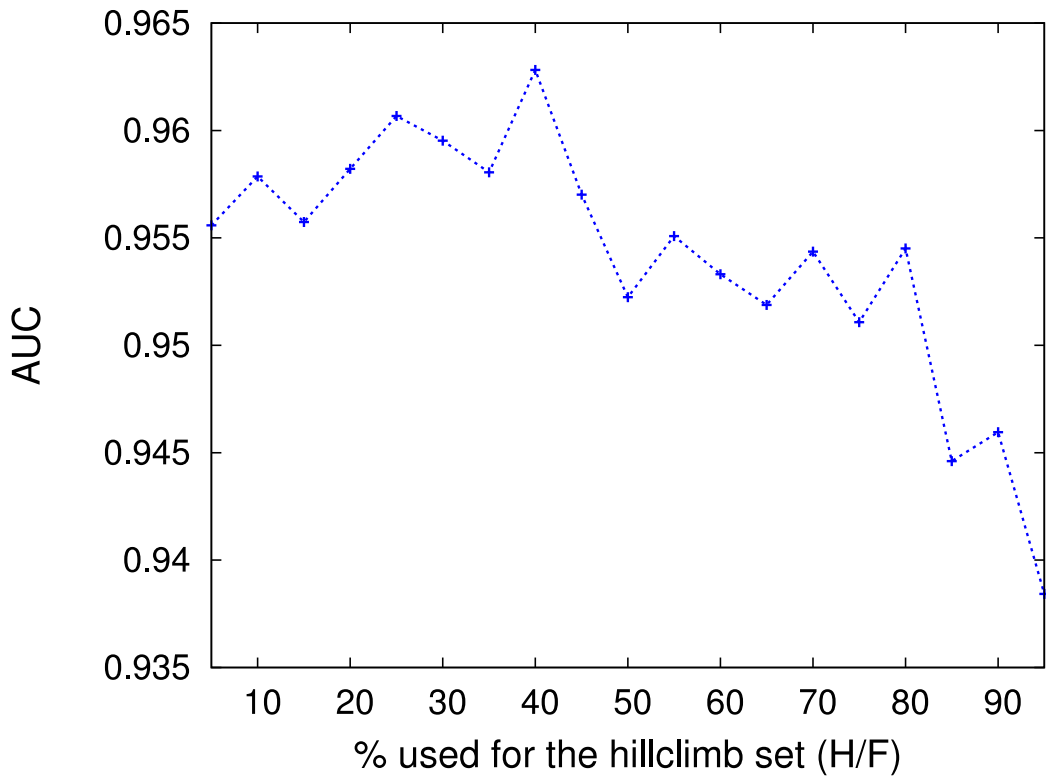
However, sometimes ES overfits the hillclimbing set, reducing the performance of the final ensemble (example on next slide)

Hillclimb set overfitting (KDD09 customer churn data)



Hillclimb and test set learning curves of ES. The red curve is the hillclimb set performance and the blue curve is the test set performance.

How much data should be used for the hillclimb set? (Waveform-5000 data)



Given training set F, the hillclimb set H is a subset of F. Hillclimb set ratio $r = H/F$. For this dataset, $r \approx 0.4$ is optimal.

Our idea - ES ensemble

- Bagging (Breiman1996) is based on the instability of base classifiers, which can be exploited to improve the predictive performance of such unstable base classifiers. ES is unstable, so theoretically bagging ES may produce a more robust ensemble.
- Use the out-of-bag sample for hillclimbing (no longer need to estimate the optimal hillclimb set ratio)

We propose three bagging ES algorithms

The **BaggingES-Simple** algorithm is the straightforward application of bagging to ensemble selection, with ES being the base classifier. (still need to set the hillclimb ratio)

The **BaggingES-OOB** algorithm uses the full bootstrap sample for model generation, and the respective out-of-bag sample as the hillclimb set for model selection. The hillclimb set is expected to have about $1/e \approx 36.8\%$ unique examples. (no hillclimb set ratio any more).

The **BaggingES-OOB-EX** algorithm is an extreme case of BaggingES-OOB, where in each bagging iteration only the single best classifier (in terms of performance on the hillclimb set) is selected. Therefore, if the number of bagging iterations is M, then the final ensemble size will be exactly M as well.

The BaggingES-OOB algorithm

Inputs:

S is the training set

E is the Ensemble Selection classifier

T is the number of bootstrap samples

Basic BaggingES-OOB procedure:

for $i \leftarrow 1$ to T {

$S_b \leftarrow$ bootstrap sample from S

$S_{oob} \leftarrow$ out of bag sample

train base classifiers in E on S_b

$E_i \leftarrow$ do ensemble selection based on base classifiers' performance on S_{oob}

}

Datasets basic characteristics

10 real-world datasets, converted to binary problems by keeping only the two largest classes each. A subset of 10k instances is selected for experiments.

Data set with release year	#Insts	Atts:Classes	Class distribution (#Insts)
Adult 96	48,842	14:2	23% vs 77% (10,000)
Chess 94	28,056	6:18	48% vs 52% (8,747)
Connect-4 95	67,557	42:3	26% vs 74% (10,000)
Covtype 98	581,012	54:7	43% vs 57% (10,000)
KDD09 CustomerChurn 09	50,000	190:2	8% vs 92% (10,000)
LocalizationPersonActivity 10	164,860	8:11	37% vs 63% (10,000)
MAGICGamma Telescope 07	19,020	11:2	35% vs 65% (10,000)
MiniBooNE Particle 10	130,065	50:2	28% vs 72% (10,000)
Poker Hand 07	1,025,010	11:10	45% vs 55% (10,000)
UCSD FICO Contest 10	130,475	334:2	9% vs 91% (10,000)
Original data sets			Final binary data sets

BaggingES vs. ES/ES++ Experimental setup

Algorithms in comparison

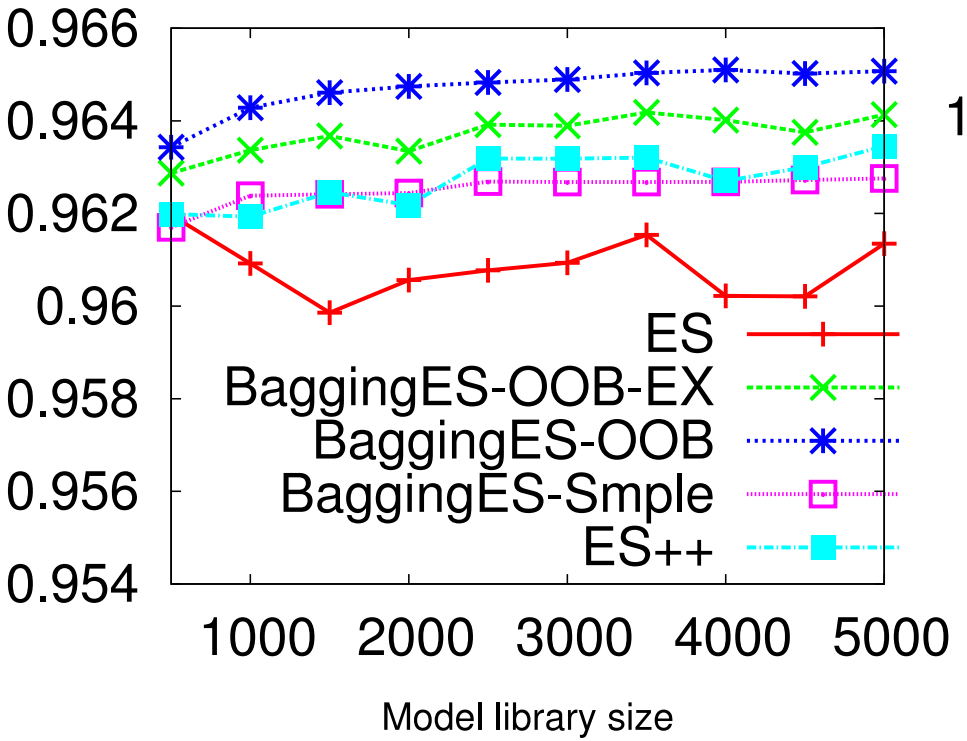
- BaggingES-Simple, BaggingES-OOB, BaggingES-OOB-EX
- ES and ES++ (improved version of ES, [Caruana2004])
- RandomTree is used as the base classifier

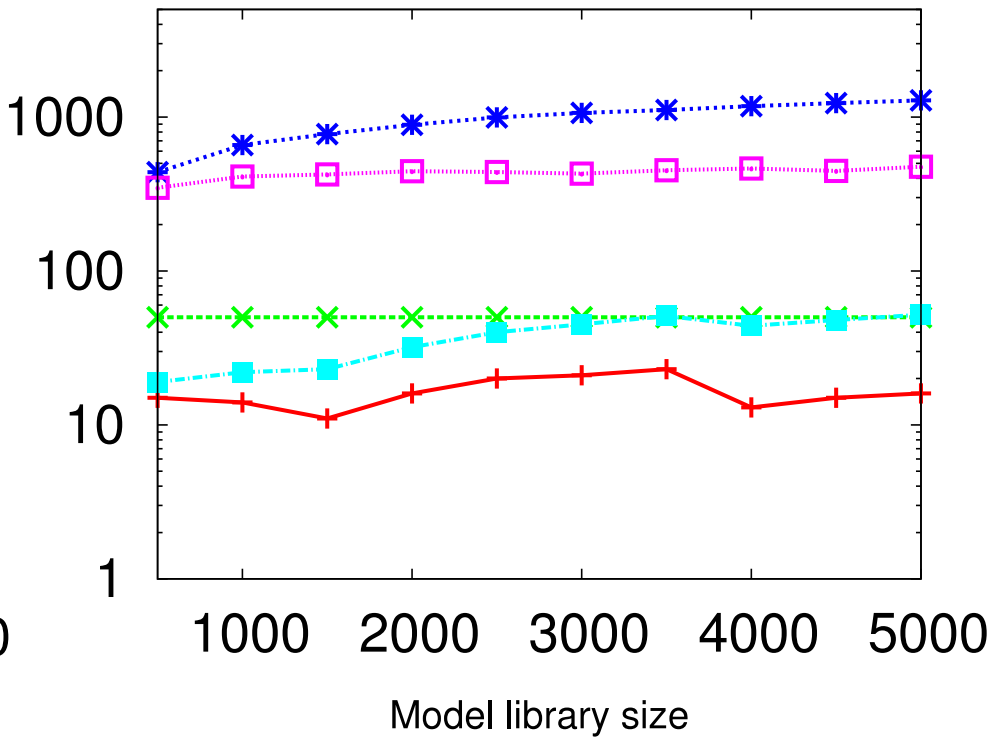
Bagging iterations for the three BaggingES algorithms is set to 50. For each dataset, we run 10 experiments per algorithm, increasing the size of the model library per bag by 10 for each successive experiment: from 10 to 20, then to 30 and so on until 100 for the tenth experiment.

AUC is calculated based on 5 runs of 66% vs. 34% training/testing evaluation per experiment.

BaggingES vs. ES/ES++ (same num. of base classifiers)

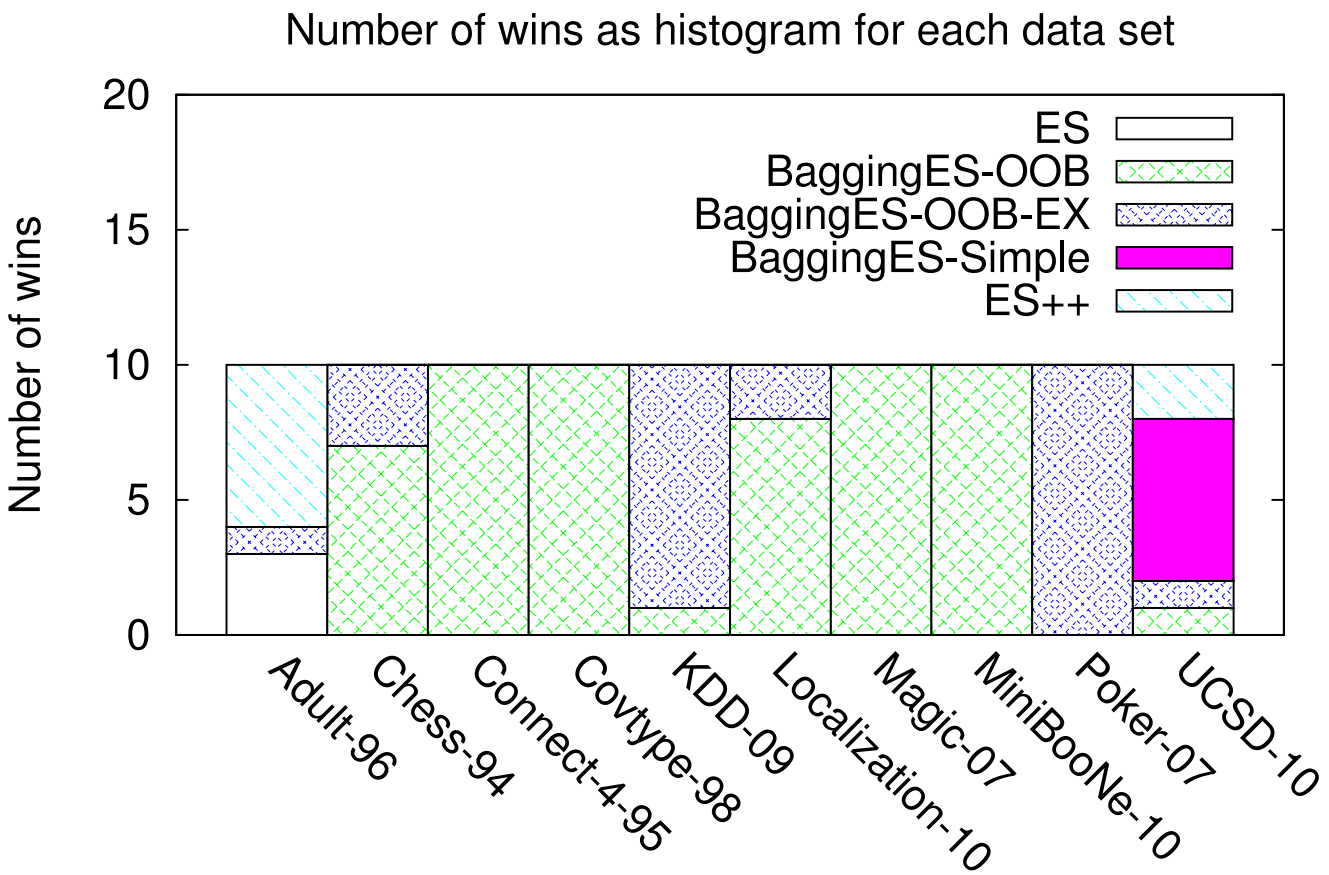
The MiniBooNE particle identification dataset, distinguish electron neutrinos (signal) from muon neutrinos (background)





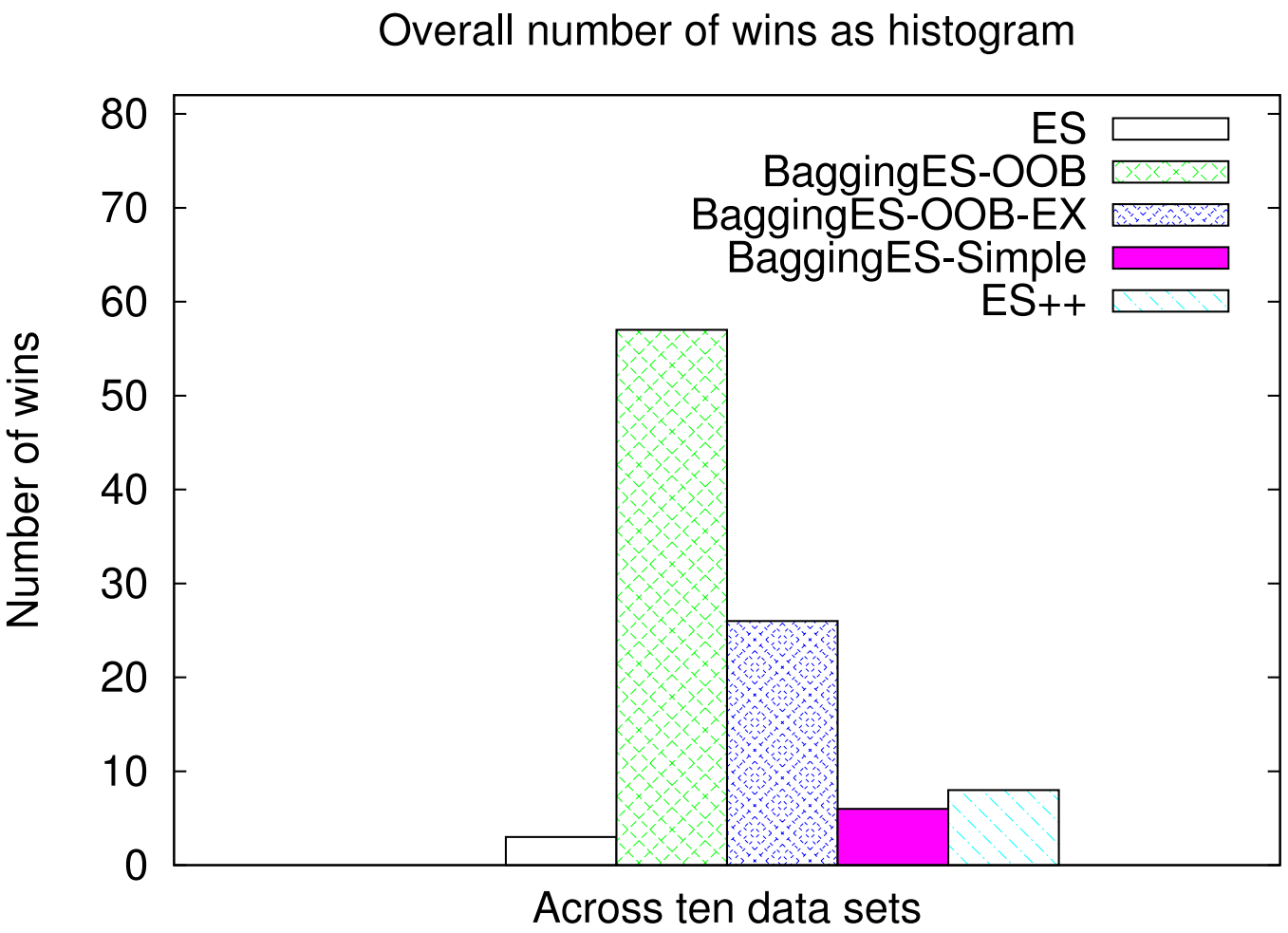
Left panel, *y*-axis is the AUC value; Right panel, *y*-axis is the final ensemble size in logarithmic scale

BaggingES vs. ES/ES++



10 datasets, 10 different model library sizes per experiment. In total 500 individual experiments

BaggingES vs. ES/ES++



10 datasets, 10 different model library sizes per experiment. In total 500 individual experiments

BaggingES vs. Other ensemble algorithms
Experimental setup

- Compared to Voting, Stacking (linear regression as the meta-level classifier), AdaBoost.M1 and RandomForest.
- Max. number of base classifiers (RandomTree) is allowed to use is 5,000 (please note that AdaBoost.M1 may stop early).
- Num. of bagging iterations for BaggingES-OOB is set to 50; 100 base classifiers per bag (in total 5,000 base classifiers).

Results: BaggingES vs. Other ensemble algorithms (same num. of base classifiers)

"●" BES-OOB is significantly better, "○" BES-OOB is significantly worse, significance level 0.05; BES-OOB is significantly better on 82% of all 50 experiments (loss: 2, tie: 7, wins: 41);

Dataset	BES-OOB	Voting	Stacking	AdaBoost.M1	RandomForest
Adult-96	0.905	0.902 ●	0.892 ●	0.783 ●	0.902 ●
Chess-94	0.875	0.859 ●	0.841 ●	0.971 ○	0.862 ●
Connect-04	0.918	0.911 ●	0.897 ●	0.905 ●	0.912 ●
Covtype-98	0.884	0.882 ●	0.875 ●	0.878 ●	0.882 ●
KDD-09	0.678	0.678 -	0.656 ●	0.580 ●	0.675 -
Localiz-10	0.966	0.957 ●	0.940 ●	0.938 ●	0.960 ●
Magic-07	0.920	0.916 ●	0.910 ●	0.868 ●	0.919 ●
MiniB-10	0.964	0.963 ●	0.959 ●	0.928 ●	0.963 ●
Poker-07	0.697	0.660 ●	0.620 ●	0.740 ○	0.674 ●
UCSD-10	0.649	0.648 -	0.612 ●	0.632 ●	0.646 -
(win/tie/loss)		(0/2/8)	(0/0/10)	(2/0/8)	(0/2/8)

Conclusions & Future work

Conclusions

- BaggingES-based ensemble strategies, particularly the BaggingES-OOB algorithm outperforms the original ES algorithm
- On the 10 datasets tested in this paper, BaggingES-OOB's predictive performance is competitive (in many cases, superior) to other popular ensemble algorithms
- If fast prediction is required, then the BaggingES-OOB-EX algorithm is recommended (because the user can set the size of the final ensemble)

Future work

- BaggingES for regression
- Experiment with more diverse model library

Thank You

Questions?

Appendix - Some additional materials

In the BaggingES vs. ES/ES++ experiment, the hillclimb set ratio for ES/ES++ is set to 30%

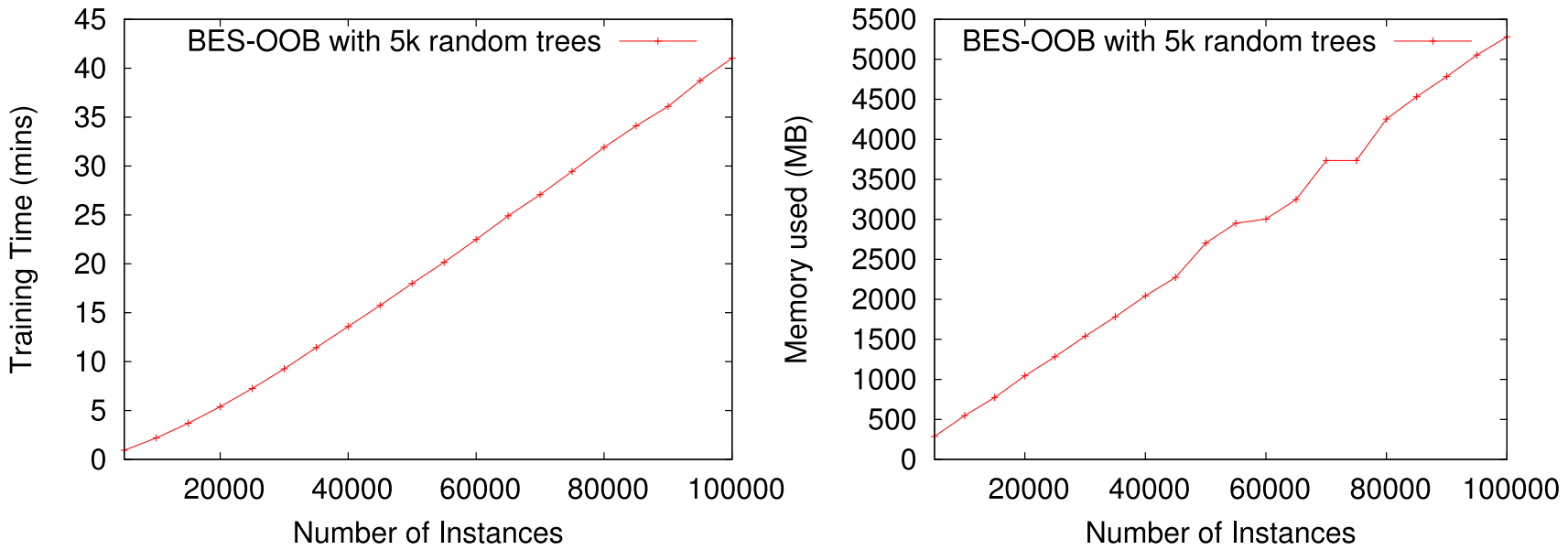


Figure: Training time and memory cost for building a BES-OOB model with 5k random trees (50 bags, 100 trees/bag, minInstancesAtLeaf = 50) on a data set with 100 numeric attributes and 2 classes. Exp is done on an i7 PC (6 cores are used when building each model).

Appendix - BES-OOB for Regression