ECMLPKDD
PRAGUE
Sep 2013

Pairwise Meta-Rules for Better Meta-Learning-based Algorithm Ranking

Quan Sun and Bernhard Pfahringer, Department of Computer Science, The University of Waikato, New Zealand

{qs12, bernhard}@cs.waikato.ac.nz

THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

Meta-Learning Introduction

- Meta-learning is a rich field, usually explained as “learning to learn”. In this paper, the term is used in the sense of “meta-learning for algorithm ranking or recommendation”.
- When choosing an algorithm for a given dataset, simply optimising all known learning algorithms is usually not feasible under strict time constraints.
- Meta-learning tries to support and automate this process. Meta-learning generates meta-knowledge mapping the properties of a dataset, captured by meta-features, to the relative performances of the available algorithms.

A meta-learning system

The basic steps of building a meta-learning system:

1. collect a set of datasets
2. define some meta-features of each dataset, e.g., the #. of instances, the #. of numeric or categorical features... Existing meta-learning systems are mainly based on three types of meta-features: statistical, information-theoretic and landmarking-based meta-features, or **SIL** for short.
3. estimate the predictive performance of the available algorithms (eg, CV), for every dataset in the dataset collection

ART Forests

```
Input:
T (number of ART to use)
D (training data);
u (number of features to use, default $log_2 M+1$)
C (splitting and stopping criterions, details are
    given in the paper)

$ART_{ensemble} \leftarrow \emptyset$
for $i = 1$ to $T$
  $D_i \leftarrow getBootstrapSample(D)$
  $ART_i \leftarrow$ ART$(D_i, u, C)$
  $ART_{ensemble} \leftarrow ART_{ensemble} \cup ART_i$
end for
return $ART_{ensemble}$
```

Meta-Dataset

For algorithm ranking/recommendation, our goal is to predict the relative performance between algorithms. Thus, the (raw) meta-dataset can be transformed to represent the rankings of the algorithms.

$$M = \begin{array}{c|cccccc} & f_1 & f_2 & f_3 & \text{C4.5} & \text{SVM} & \text{k-NN} \\ d_1 & 100 & 0.52 & -1.0 & 0.85 & 0.86 & 0.77 \\ d_2 & 300 & 0.45 & 2.0 & 0.55 & 0.52 & 0.70 \\ d_3 & 450 & 0.77 & 1.5 & 0.71 & 0.83 & 0.69 \end{array}$$

$$M^* = transform(M) \Longrightarrow$$

$$\begin{array}{c|cccccc} & f_1 & f_2 & f_3 & \text{C4.5} & \text{SVM} & \text{k-NN} \\ d_1 & 100 & 0.52 & -1.0 & 2 & 1 & 3 \\ d_2 & 300 & 0.45 & 2.0 & 2 & 3 & 1 \\ d_3 & 450 & 0.77 & 1.5 & 2 & 1 & 3 \end{array}$$

ART's Splitting Criterion

In the ART algorithm, we use the median value of a meta-feature's range as the binary split point to split the data D, the current partition, into two sub-partitions D^+ and D^-. The best split point is determined to be the one that maximises the R^2 statistic:

$$R^2 = 1 - \frac{\sum_{l=1}^{L}\sum_{i=1}^{n^{(l)}} d_{Spearman}(y^{(li)}, \hat{z}^{(l)})}{\sum_{l=1}^{L}\sum_{i=1}^{n^{(l)}} d_{Spearman}(y^{(li)}, \hat{z}^{(D)})}, \quad (1)$$

where L is the number of partitions, and $n^{(l)}$ is the number of examples in partition l. R^2 is originally designed to measure the proportion of the spread explained by the differences between the two partitions. In the paper, we showed that R^2 can be computed efficiently:

$$R^2 = 1 - \frac{n^{(D^+)}(h - 2||\bar{y}^{(D^+)}||^2) + n^{(D^-)}(h - 2||\bar{y}^{(D^-)}||^2)}{n^{(D)}(h - 2||\bar{y}^{(D)}||^2)}, \quad (2)$$

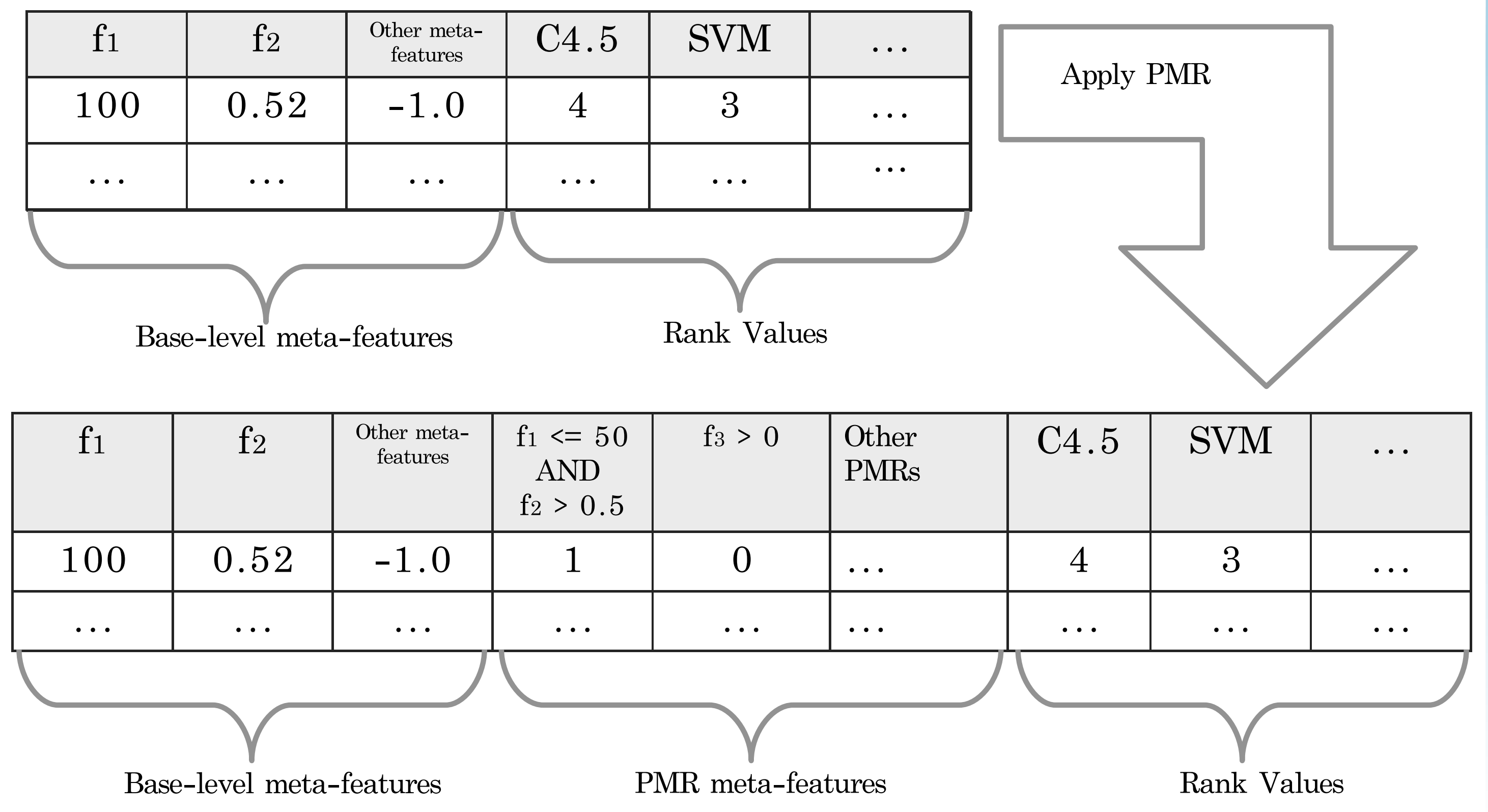
where $h = \frac{m(m+1)(2m+1)}{3}$.

Additional materials

The ART Forests software can be downloaded at:
`http://www.cs.waikato.ac.nz/~qs12/ml/meta/`

Pairwise Meta-Rules (PMR)

1. Construct a binary classification dataset for each algorithm pair. Each binary dataset (i, j pair, $i < j$) has two class labels. In total, there are $\frac{m\times(m-1)}{2}$ (m is the #. of target algorithms) binary classification datasets.
2. Build a RIPPER rule model for each of the $\frac{m\times(m-1)}{2}$ binary datasets.
3. Add meta-rules in each RIPPER model as new meta-features to the original meta-feature space.



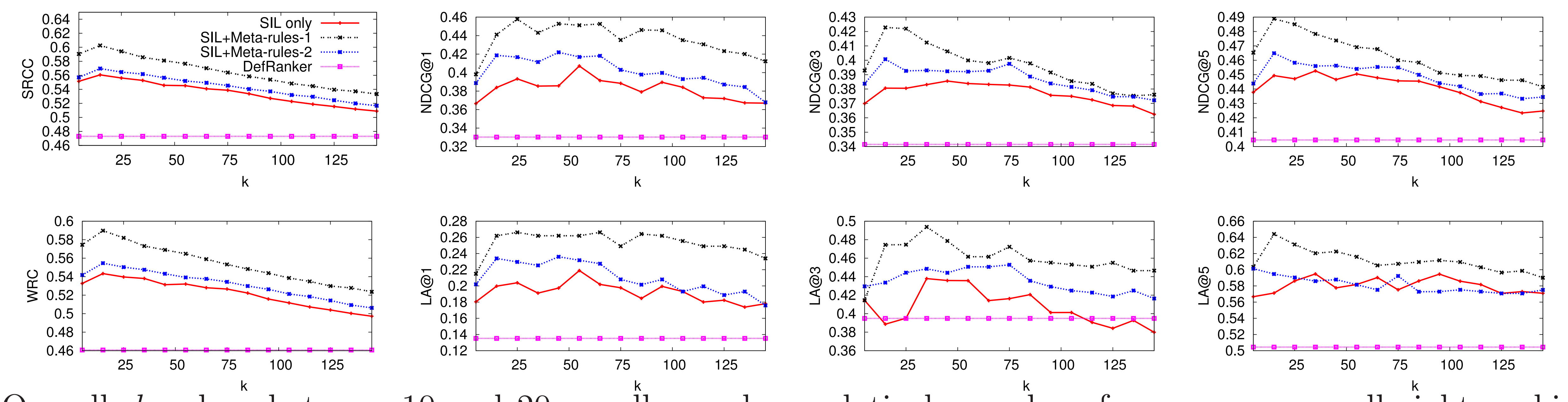
f1	f2	Other meta-features	C4.5	SVM	...
100	0.52	-1.0	4	3	...
...	...	...	...	...	...

f1	f2	Other meta-features	f1 <= 50 AND f2 > 0.5	f3 > 0	Other PMRs	C4.5	SVM	...
100	0.52	-1.0	1	0	...	4	3	...
...	...	...	...	...	...	...	...	...

Experiments and Results

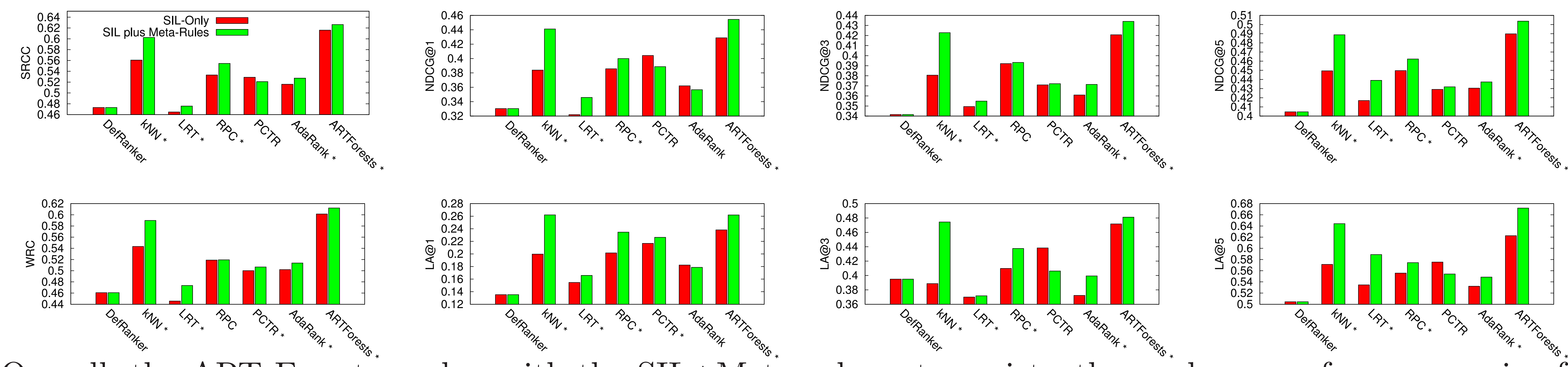
In this paper, meta-learning is used to rank 20 supervised machine learning algorithms over 466 datasets. When generating algorithm rankings from the 466 datasets, for each of the 20 algorithms, we manually specify parameters and their respective value ranges for PSO to optimise. AUC is used as the target metric. We run the 20 algorithms, with PSO-based parameter optimisation, on 466 binary classification datasets and use 10-fold cross-validation based AUC scores for ranking generation.

Comparison of meta-feature sets based on k-NN performance curves We compared 3 meta-feature sets, including two PMR-based variants.



Overall, k values between 10 and 20 usually produce relatively good performance across all eight ranking metrics. Regarding the choice of meta-feature sets, the SIL+Meta-rules-1 set outperforms the SIL-only and the SIL+Meta-rules-2 meta-feature sets.

Comparison of ranking performances of multiple rankers 7 meta-learners are used in experiments:



Overall, the ART Forests ranker with the SIL+Meta-rules set consistently produces performance gains for all different metrics, and is placed as the best ranker for 7 out of 8 metrics.