Hierarchical Meta-Rules for Scalable Meta-Learning

Quan Sun and Bernhard Pfahringer

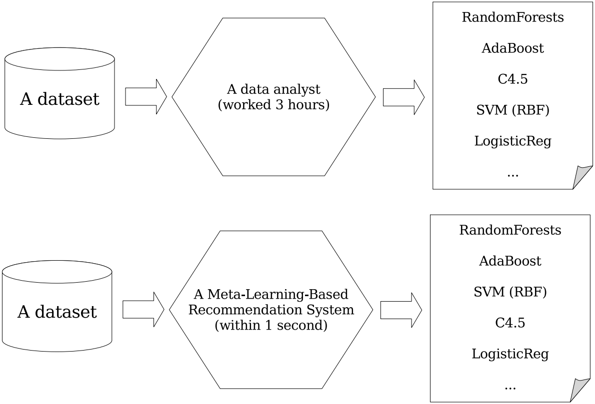
Waikato University
Hamilton, New Zealand

03 Dec 2014

Meta-learning

- Meta-learning is usually explained as "learning to learn".
- In this paper, the term is used in the sense of "**meta-learning for algorithm ranking or recommendation**".

A Successful Meta-learning System



- Meta-learning tries to support and automate algorithm selection, by generating meta-knowledge mapping the properties of a dataset to the relative performances of algorithms.

Recent Meta-learning Applications

OpenML.org — sharing experimental results, learn from data and experiments ...

DataRobot.com — automatically generate thousands of models and select the most accurate ones . . .

The Meta-learning Task

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

Meta-Dataset

Thus, for each dataset we get a list of available algorithms with their performance estimates. Given the above information, we can construct a meta-dataset, which is a $n \times v$ matrix, where $v = u + m$. Here, v is the sum of the number of meta-features u and the number of algorithms m, and n is the number of datasets. Below is an example dataset, where $n = 3$, $u = 3$ and $m = 5$.

$$M = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \end{array} \begin{array}{c} \begin{array}{cccccccc} f_1 & f_2 & f_3 & \text{C4.5} & \text{LG} & \text{k-NN} & \text{RF} & \text{GBT} \end{array} \\ \begin{pmatrix} 100 & 0.52 & -1.0 & 0.85 & 0.86 & 0.77 & 0.93 & 0.92 \\ 300 & 0.45 & 2.0 & 0.55 & 0.52 & 0.70 & 0.85 & 0.81 \\ 450 & 0.77 & 1.5 & 0.71 & 0.83 & 0.69 & 0.74 & 0.78 \end{pmatrix} \end{array}$$

- For algorithm ranking, 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.

Meta-Dataset cont'd

$$M = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \end{array} \begin{array}{c} \begin{array}{cccccccc} f_1 & f_2 & f_3 & \text{C4.5} & \text{LG} & \text{k-NN} & \text{RF} & \text{GBT} \end{array} \\ \begin{pmatrix} 100 & 0.52 & -1.0 & 0.85 & 0.86 & 0.77 & 0.93 & 0.92 \\ 300 & 0.45 & 2.0 & 0.55 & 0.52 & 0.70 & 0.85 & 0.81 \\ 450 & 0.77 & 1.5 & 0.71 & 0.83 & 0.69 & 0.74 & 0.78 \end{pmatrix} \end{array}$$

Thus, the original meta-dataset can be transformed to represent the ranks:

$$\Gamma = \mathit{transform}(M) \equiv$$

$$\begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \end{array} \begin{array}{c} \begin{array}{cccccccc} f_1 & f_2 & f_3 & \text{C4.5} & \text{LG} & \text{k-NN} & \text{RF} & \text{GBT} \end{array} \\ \begin{pmatrix} 100 & 0.52 & -1.0 & 4 & 3 & 5 & 1 & 2 \\ 300 & 0.45 & 2.0 & 4 & 5 & 3 & 1 & 2 \\ 450 & 0.77 & 1.5 & 4 & 1 & 5 & 3 & 2 \end{pmatrix} \end{array}$$

Meta-learning Approaches

- The k-Nearest Neighbors approach
- The pairwise classification approach
- The learning to rank approach
- The label ranking approach
- The single/multi-target regression approach

Pairwise Meta-Rules (PMR)

- PMR uses a rule learner to learn pairwise rules between targets first, and then use these rules as new meta-features.
- Explicitly adding the logical pairwise information between each pair of the target algorithms to the meta-feature space might improve a meta-learner's predictive accuracy.

Pairwise Meta-Rules: Step 1

Construct a binary classification dataset for each algorithm pair. Each binary dataset (i, j pair, $i < j$) has two class labels:

$$A^{(ij)} = \begin{array}{cc} & \begin{array}{ccccc} f_1 & f_2 & \cdots & f_u & \text{class label} \end{array} \\ \begin{array}{c} d_1 \\ d_2 \\ \vdots \\ d_n \end{array} & \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,u} & l_1 = \begin{cases} \texttt{Yes} & \text{if Algorithm } i \text{ is better;} \\ \texttt{No} & \text{otherwise.} \end{cases} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,u} & l_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,u} & l_n \end{pmatrix} \end{array}$$

In total, there are $\frac{m \times (m-1)}{2}$ (m is the #. of target algorithms) binary classification datasets.

Pairwise Meta-Rules: Step 2

- Build a RIPPER rule model for each of the $\frac{m \times (m-1)}{2}$ binary datasets.
- Add meta-rules in each RIPPER model as new meta-features to the original feature space

These rules are called the Pairwise Meta-Rules (PMR). Following is an example rule model (set) for two algorithms `SVM` and `C4.5`:

```
If Num.Features > 100 AND Num.Instances < 3000 Then SVM is better
If Num.NumericFeatures > 80%  Then SVM is better
Otherwise C4.5 is better.
```

Pairwise Meta-Rules (PMR)

f_1	f_2	Other meta-features	C4.5	SVM	...
100	0.52	-1.0	4	3	...
...	...	...	...	...	...

Base-level meta-features

Rank Values

Apply PMR

f_1	f_2	Other meta-features	$f_1 <= 50$ AND $f_2 > 0.5$	$f_3 > 0$	Other PMRs	C4.5	SVM	...
100	0.52	-1.0	1	0	...	4	3	...
...	...	...	...	...	...	...	...	...

Base-level meta-features

PMR meta-features

Rank Values

Pairwise Meta-Rules Training Complexity

Given m target objects (e.g., algorithms), the training complexity of the PMR method with respect to m is quadratic: $\binom{m}{2} = m \times (m-1)/2$. This is usually not a problem when m is moderate, such as when ranking 20 different learning algorithms.

However, for problems with a much larger m, such as the meta-learning-based parameter ranking problem, where m can be 100+, the PMR method is less efficient.

Pairwise Meta-Rules Training Complexity

In this paper, we propose a novel method named Hierarchical Meta-Rules (HMR), which is based on the theory of orthogonal contrasts.

The proposed HMR method has a linear training complexity with respect to m, providing a way of dealing with a large number of objects that the PMR method cannot handle efficiently.

Hierarchical Meta-Rules

Assume we have three algorithms C4.5, Logistic Regression (LG) and Random Forests (RF) to rank,

in the Pairwise Meta-Rules (PMR) method, we build m x (m - 1) / 2 = 3 x (3 - 1) / 2 = 3 rule models: {C4.5 vs. LG}, {C4.5 vs. RF}, {LG vs. RF};

In the Hierarchical Meta-Rules (HMR), we build only m - 1 = 3 - 1 = 2 rule models, say only for {C4.5 vs. LG}, {C4.5 vs. RF}, given we *thought* building a model for {LG vs. RF} is not necessary for this case.

Hierarchical Meta-Rules (HMR)

"...*given we thought building a model for $\{LG$ vs. $RF\}$ is not necessary for this case*..."

well, the thought is based on:

applying the orthogonal contrasts (OC) theory to replace pairwise comparison (in PMR) with group-wise comparison (in HMR), so unnecessary pairwise comparisons are eliminated without sacrificing too much predictive performance.

Hierarchical Meta-Rules (HMR)

The *groups* in group-wise comparisons are found by using an *appropriate* clustering algorithm that is consistent with orthogonal contrasts (OC) definitions and propositions (def. 1,2,3,4 and prop. 1, 2 in Page. 7 - 8).

Experiments

4 meta-learning datasets (sec. 3.1)

Dataset	#.Instances	#.Features	#.Targets
algo20	466	80	20
rf70	466	80	70
lg100	466	80	100
smo110	466	80	110

Experiments

2 clustering algorithms (sec. 3.2)

- Hierarchical Clustering with complete linkage (HC)
- Bisecting k-means (BKM)

Experiments

2 meta-learning algorithms (sec. 3.3)

- Approximate Ranking Tree Forests (ARTF)
- k-nearest neighbors for meta-learning (k-NN)

Experiments

Dataset	Base Features (BF)	BF plus HMR-HC	BF plus HMR-BKM	BF plus PMR
algo20	0.601±0.018	0.599±0.018	0.599±0.018	0.608±0.016 •
rf70	0.558±0.030	0.569±0.032 •	0.571±0.030 •	0.584±0.033 •
lg100	0.666±0.016	0.674±0.017 •	0.673±0.017 •	0.675±0.017 •
smo110	0.218±0.017	0.217±0.017	0.218±0.020	0.219±0.016 •

(a) Meta-learner: ARTF

Dataset	Base Features (BF)	BF plus HMR-HC	BF plus HMR-BKM	BF plus PMR
algo20	0.552±0.019	0.559±0.018 •	0.559±0.019 •	0.592±0.017 •
rf70	0.500±0.033	0.532±0.036 •	0.539±0.033 •	0.557±0.035 •
lg100	0.653±0.016	0.665±0.017 •	0.664±0.016 •	0.667±0.018 •
smo110	0.174±0.015	0.171±0.014 ○	0.174±0.015	0.189±0.015 •

(b) Meta-learner: *k*-NN

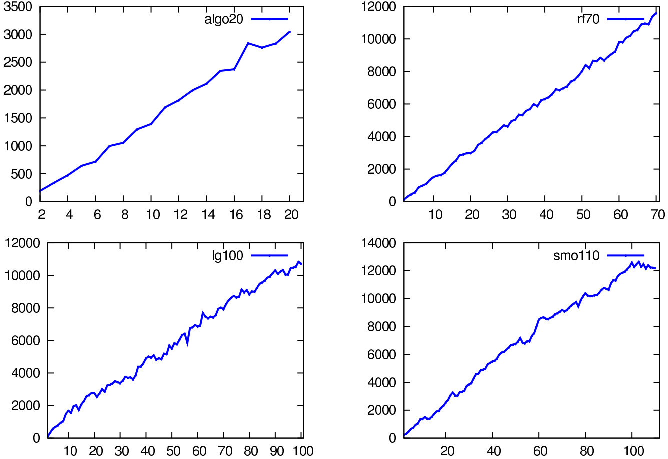
Ranking performances of two meta-learners that use or without using meta-rule methods. • or ○ means the predictive performance of a meta-learner using the respective meta-rule method is significantly better or worse than that without using the predictive meta-rule method.

Experiments

Dataset	HMR-HC	HMR-BKM	PMR
algo20	2	2	24
rf70	13	13	854
lg100	18	17	1343
smo110	22	22	2070

Runtime (in seconds) of different meta-rule construction methods; Values are the average of 30 runs.

Experiments



Meta-rule construction runtime of the HMR-HC method on four datasets. Values in the figures represent the mean of 10 runs. X-axis represents the number of targets; Y-axis represents the meta-rule construction time in milliseconds.

Conclusions

- Speedup Pairwise Meta-Rules (PMR) with Hierarchical Meta-Rules (HMR) — quadratic to linear training complexity
- k-NN meta-learner plus HMR-based methods worked better
- HMR is an early application of the OC theory (the idea may be applied to other problems, e.g. label ranking, multi-label classification)

Thank you :-)