Algorithm Recommendation as Collaborative Filtering

Michèle Sebag & Mustafa Misir & Philippe Caillou

TAO, CNRS − INRIA − Université Paris-Sud

AutoML Wshop, ICML 2015

Control layer in algorithmic platforms

Goal

deliver peak performance on any/most problem instances

A general issue

- In constraint programming — Rice 76
- In stochastic optimization — Grefenstette 87
- In machine learning (meta-learning) — Bradzil 93

Scope: Selection and Calibration

- Offline control
  Portfolio algorithm selection, optimal hyper-parameter setting
- Online control
  adjusting hyper-parameters during the run

Control layer in algorithmic platforms

Goal

deliver peak performance on any/most problem instances

A general issue

- In constraint programming — Rice 76
- In stochastic optimization — Grefenstette 87
- In machine learning (meta-learning) — Bradzil 93

Scope: Selection and Calibration

- Offline control
  Portfolio algorithm selection, optimal hyper-parameter setting
- Online control
  adjusting hyper-parameters during the run

Control

An optimization problem

Given a problem instance, find

$$\theta^* = \text{arg opt } \{ \text{ Performance } (\theta, \text{ pb instance}) \ \}$$

with θ: algorithm and hyper-parameters thereof

Learn objective function "Performance"

- Learn it (surrogate optimisation)

  Hutter et al. 11; Thornton et al. 13

- Learn a monotonous transformation thereof

  Bardenet et al. 13; this talk

See also Reversible learning McLaurin et al. 15

Control: A meta-learning problem

Procedure

- Gather problem instances (benchmark suite)
- Design descriptive features for pb instances
- Run algorithms on pb instances
- Build meta-training set:

$$\mathcal{E} = \{(\text{desc. of } i\text{-th pb instance, perf. of } j\text{-th algo})\}$$

- Learn $\hat{h}$ from $\mathcal{E}$
- Decision making (predict, optimize)

Some advances in CP and SAT

- CPHydra — O'Mahony et al. 08
  case-based reasoning; kNN
- Satzilla — Xu et al. 08
  learn $\widehat{\text{runtime}}$(inst,alg); select argmin $\widehat{\text{runtime}}$
- ParamILS — Hutter et al. 09
  learn $\widehat{\text{perf}}$(hyper-param); optimize $\widehat{\text{perf}}$
- Programming by optimization — Holger Hoos, 12
  http://www.prog-by-opt.net/

100 Features — Hutter et al. 06, 07

Static features

Problem definition: density, tightness

Variable size and degree (min, max, average, variance

Constraint degree and cost category (exp, cubic, quadratic,

lin. cheap, lin. expensive)

Dynamic features

Heuristic criteria(variable): wdeg, domdeg, impact: min, max, average

Constraint weight (wdeg): min, max, average

Constraint filtering: min, max,

average of number of times called by propagation

ML control, the bottleneck

$$\mathcal{E} = \{(\text{desc. of } i\text{-th pb instance, perf. of } j\text{-th algo})\}$$

Bottleneck: design good cheap descriptive features

Tentative interpretation

- SAT: "high level" problem instance
- ML: a problem instance is a dataset $\equiv$ distribution.
  Learning distribution parameters is expensive

Some advances in ML

- Matchbox — Stern et al. 10
  Collaborative filtering + Bayesian learning
- SCOT — Bardenet et al. 13
  $\widehat{\text{perf}}$(hyper-param); optimize $\widehat{\text{perf}}$
  where $\widehat{\text{perf}}$ is learned using learning-to-rank.
- AutoWeka — Thornton et al. 13
  SMAC (Sequential Model-based Algorithm Configuration)
  applied on the top of Weka.

Overview

Context

ALORS: Algorithm Recommender System

Empirical evaluation
- Collaborative filtering performance
- Cold start performance

Visualizing the problem/alg landscape

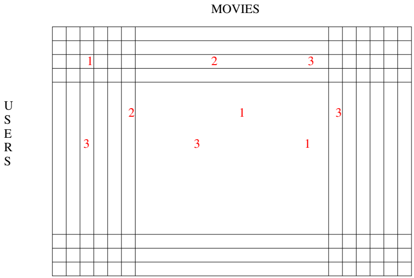
Main idea

Stern et al. 10

Netflix challenge

Recommender systems

- Set of users, set of products
- Users like/dislike a few products
- A sparse matrix



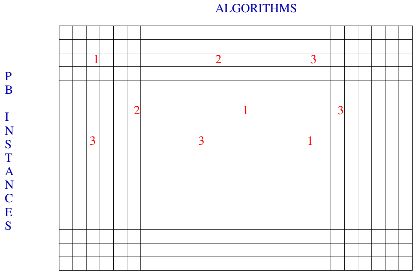
Main idea

Stern et al. 10

Recommender systems

Netflix challenge

- Set of users, set of products
- Users like/dislike a few products
- A sparse matrix



Algorithm selection

- Set of problem instances, set of algorithms
- Pb instance likes "better"

  algorithms that behave "better" on instance

Differences

- Meta-Learning is not (yet) a Big Data problem
  (500.000 users, 180.000 movies in Netflix)
- The main issue is: dealing with a brand new problem instance:
  cold start

Milestones

Acquire data — Sparse matrix

- Run a few alg. on problem instances

Collaborative filtering — Fill the matrix

- Content-based
- Model-based

Cold start

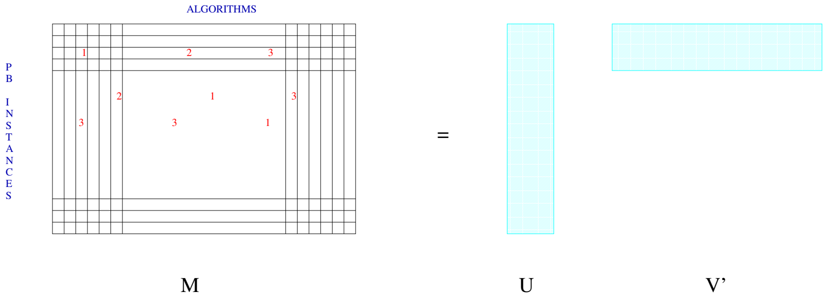
- Handle a brand new pb instance

Collaborative filtering

Matrix decomposition

- U: $P \times k$, k = nb latent factors
- V: $A \times k$,
- s.t. $\mathcal{M} \approx UV'$

$$m_{i,j} \approx \langle u_i, v_j \rangle$$

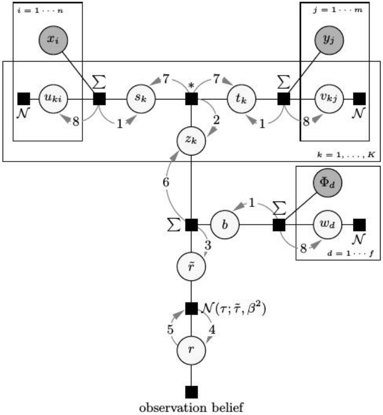


Bayesian Collaborative filtering

Matchbox

Sterner et al. 10

- Define priors on U and V independent Gaussian
- Finite number of perf. levels (1, 2, 3)
- Learn thresholds from $\langle u, v \rangle$ to perf. level
- Latent features = linear combinations of initial features



Matchbox, 2

Specificities

- Include a bias $r_{i,j} \approx \langle u_i, v_j \rangle + b_i$
- Include a threshold-based rank decoding

$$m_{i,j} \approx f(r_{i,j},\ \text{thresholds})$$

Motivations

- Non stationary phenomenons
- Fast approximation possible, using a single propagation

ALORS: Algorithm Recommender System

Standard SVD: Find U (n_i, k), V (n_a, k)

$$\arg\min Loss(\mathcal{M}, UV')$$

Loss

- RMSE — root mean square error
- MAE — mean absolute error
- Rank loss — Cofirank, Weimer et al. 07

CofiRank

Criterion NDCG

$$DCG(\pi, k) = \sum_{i=1}^{k} \frac{2^{\pi(i)} - 1}{log(i+2)}$$

$$NDCG(\pi, k) = \frac{DCG(\pi, k)}{DCG(\pi^*, k)}$$

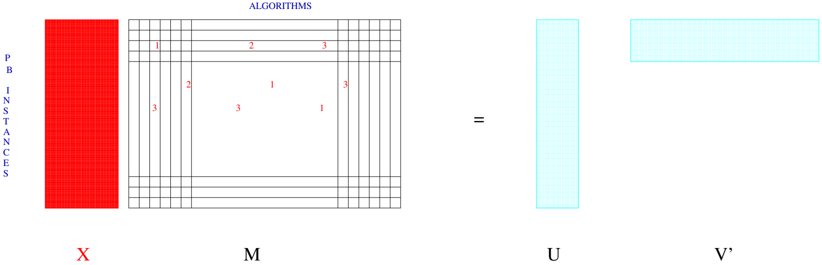
Non convex !

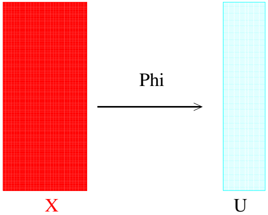
- Use a linear convex upper bound
- Alternate minimization (opt. U with fixed V; then opt. V with fixed U)

Cold start in ALORS: the cornerstone of meta-learning

Assuming descriptive features X

- Use matrix decomposition to build latent features U
- Learn $U \approx \phi(X)$





Overview

Experimental setting

Goals of experiments

- Comparison with Matchbox
- Sensitivity study wrt $\mathcal{M}$ sparsity
- Performance of cold-start
- Inspecting latent features

Domains

Satisfiability benchmark	SAT 2011
Constraint programming challenge	CP 2008
Black-box optimization benchmark	BBOB 2012
Machine learning	Joaquin Vanschoren

Experimental setting

- Sparsity in 10% - 90% (at least 1 non-missing performance on each line)
- Cold start: 10-fold CV

Comparison with Matchbox

On OpenML, SAT 2011 - 2012, CSP 2008

no significant differences

Varying the 1st rank threshold in Matchbox (5%, 10%, 33%):

no significant differences

An artificial problem

200 pb instances $\times$ 30 algorithms

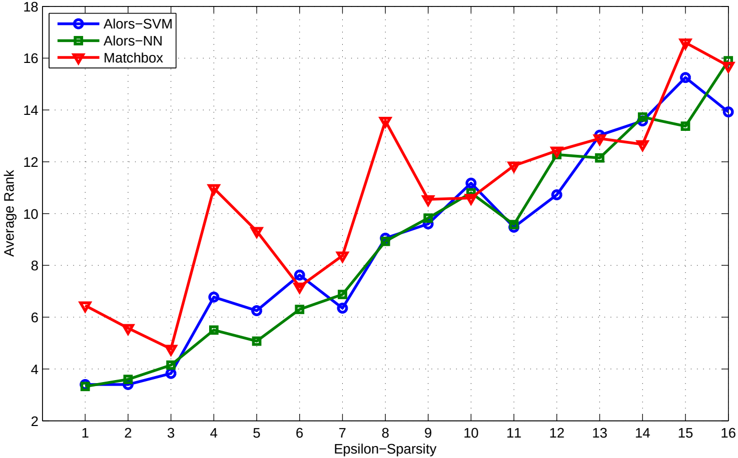
$x_i, y_j \sim U[-10, 10]^{10}$

$$m_{i,j} = d(x_i, y_j) + \mathcal{N}(0, \epsilon)$$

Where $d(x_i, y_j)$ is the Euclidean distance over three coordinates of x_i and y_j

Comparison with Matchbox, 2

Average rank of recommended system

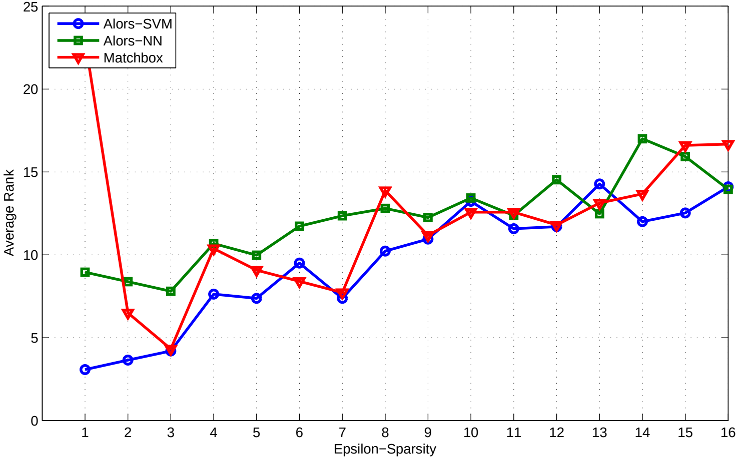


1st axis: 4 * noise + sparsity

Comparison with Matchbox, 3

A more fair comparison, providing Matchbox with features $x^{(\ell)}$ and feature products $x^{(\ell)} \times x^{(k)}$

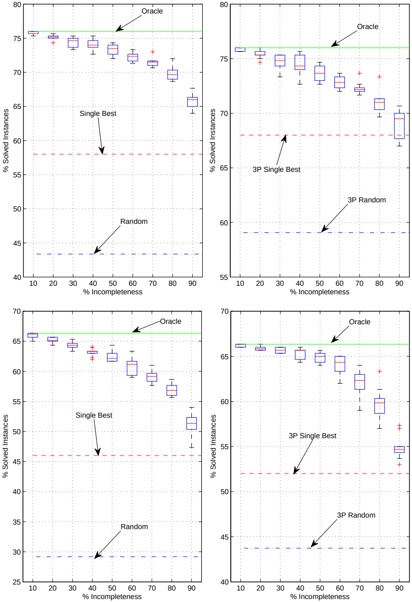
Average rank of recommended system



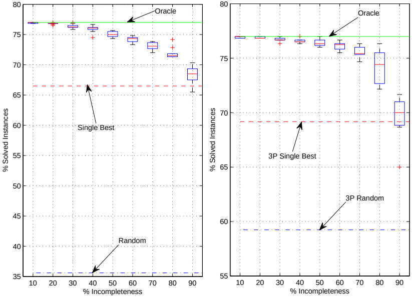
Collaborative filtering performance

- kNN-ALORS $>>$ CF-ALORS for low sparsity
- Then CF-ALORS catches up
- Low sensitivity to # latent factors $k \leq 10$

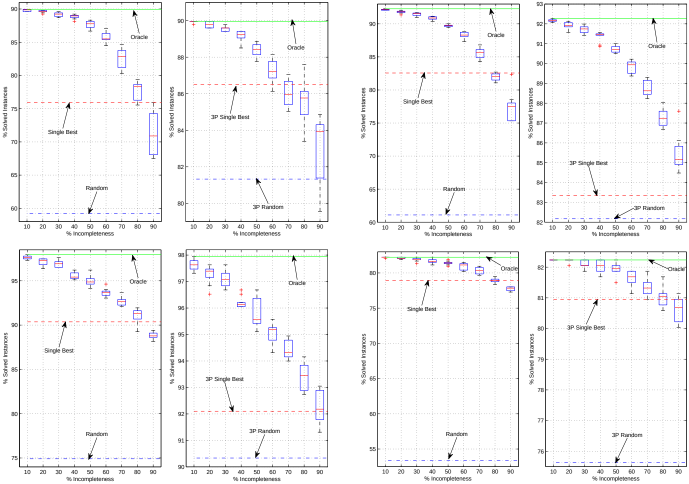
On SAT 2011


Oracle
Single Best
Random
% Solved Instances
% Incompleteness
3P Single Best
3P Random

On SAT 2011, followed

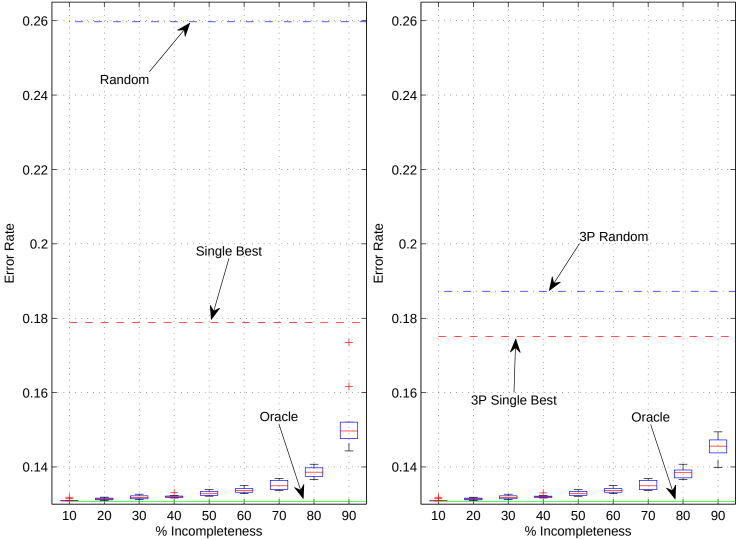


On CSP 2008



% Solved Instances
% Incompleteness
Oracle
Single Best
Random
3P Single Best
3P Random

On OpenML



3P-single best very moderately improves on the single best: the single best already is an ensemble learner (bagging).

Overview

Context

ALORS: Algorithm Recommender System

Empirical evaluation

- Collaborative filtering performance
- Cold start performance

Visualizing the problem/alg landscape

Cold start performance

On SAT

Method	Phase 1		
	APP	CRF	RND
Oracle	22.8 ± 2.5	19.9 ± 2.1	46.2 ± 3.7
SingleBest	$\mathbf{17.4 \pm 3.0}$	13.8 ± 1.9	39.9 ± 3.2
Random	13.0 ± 2.2	8.8 ± 1.5	21.4 ± 1.8
model-CF + SVM-CS	17.2 ± 2.5	$\mathbf{15.1 \pm 3.1}$	$\mathbf{43.0 \pm 3.9}$
memory-CF + SVM-CS	$\mathbf{17.7 \pm 2.6}$	$\mathbf{15.2 \pm 2.6}$	42.2 ± 4.3
model-CF + NN-CS	16.9 ± 2.9	$\mathbf{14.8 \pm 3.1}$	$\mathbf{42.9 \pm 3.7}$
memory-CF + NN-CS	17.2 ± 2.8	14.6 ± 2.8	42.2 ± 3.7
3P-SingleBest	$\mathbf{20.4 \pm 2.3}$	15.6 ± 2.0	41.5 ± 3.3
3P-Random	17.7 ± 2.2	13.1 ± 1.6	35.5 ± 2.5
3P-(model-CF + SVM-CS)	19.6 ± 2.4	$\mathbf{17.0 \pm 2.8}$	$\mathbf{45.0 \pm 3.8}$
3P-(memory-CF + SVM-CS)	19.6 ± 2.4	$\mathbf{17.0 \pm 2.4}$	$\mathbf{44.9 \pm 3.6}$
3P-(model-CF + NN-CS)	19.4 ± 2.5	16.6 ± 2.5	44.6 ± 3.6
3P-(memory-CF + NN-CS)	19.3 ± 2.6	$\mathbf{16.8 \pm 2.5}$	44.7 ± 3.5

On SAT, followed

Method	Phase 2		
	APP	CRF	RND
Oracle	25.3 ± 2.1	22.9 ± 2.5	49.2 ± 2.4
SingleBest	$\mathbf{21.5 \pm 3.6}$	16.3 ± 2.5	40.8 ± 2.4
Random	19.3 ± 2.9	12.0 ± 1.2	33.5 ± 2.3
model-CF + SVM-CS	$\mathbf{21.5 \pm 3.1}$	18.1 ± 3.0	$\mathbf{44.5 \pm 4.2}$
memory-CF + SVM-CS	21.3 ± 3.2	$\mathbf{18.5 \pm 2.7}$	44.0 ± 4.4
model-CF + NN-CS	$\mathbf{21.5 \pm 3.1}$	17.8 ± 2.6	$\mathbf{44.8 \pm 4.2}$
memory-CF + NN-CS	$\mathbf{21.5 \pm 3.3}$	$\mathbf{18.3 \pm 2.6}$	$\mathbf{44.5 \pm 4.4}$
3P-SingleBest	$\mathbf{23.5 \pm 3.0}$	18.8 ± 2.5	43.5 ± 2.3
3P-Random	22.9 ± 2.1	17.2 ± 1.4	45.1 ± 2.2
3P-(model-CF + SVM-CS)	23.4 ± 2.6	$\mathbf{20.8 \pm 3.4}$	47.1 ± 2.9
3P-(memory-CF + SVM-CS)	$\mathbf{23.5 \pm 2.6}$	$\mathbf{20.8 \pm 2.6}$	46.9 ± 3.2
3P-(model-CF + NN-CS)	$\mathbf{23.5 \pm 2.5}$	20.5 ± 2.3	$\mathbf{47.5 \pm 2.9}$
3P-(memory-CF + NN-CS)	$\mathbf{23.8 \pm 2.6}$	$\mathbf{20.7 \pm 2.4}$	$\mathbf{47.3 \pm 3.0}$

Cold start performance on CSP

Method	GLOBAL	k-ARY-INT	2-ARY-EXT	N-ARY-EXT
Oracle	49.3 ± 3.2	130.2 ± 3.2	62.0 ± 1.3	44.9 ± 2.3
Random	32.4 ± 3.2	86.2 ± 4.9	47.4 ± 3.2	29.2 ± 2.4
SingleBest	41.6 ± 5.2	$\mathbf{116.5 \pm 5.8}$	$\mathbf{57.2 \pm 2.3}$	43.1 ± 2.8
model-CF + SVM-CS	39.5 ± 5.1	111.5 ± 7.4	56.2 ± 2.9	42.2 ± 2.0
memory-CF + SVM-CS	43.6 ± 4.8	$\mathbf{115.3 \pm 6.9}$	$\mathbf{57.1 \pm 2.9}$	$\mathbf{43.4 \pm 2.4}$
model-CF + NN-CS	39.4 ± 5.1	110.8 ± 6.9	56.1 ± 2.9	42.1 ± 2.1
memory-CF + NN-CS	$\mathbf{44.1 \pm 4.4}$	115.0 ± 6.4	$\mathbf{57.4 \pm 2.9}$	$\mathbf{43.4 \pm 2.6}$
3P-Random	44.6 ± 3.2	115.9 ± 4.4	57.2 ± 2.3	41.3 ± 2.0
3P-SingleBest	$\mathbf{47.4 \pm 3.8}$	117.6 ± 5.6	$\mathbf{58.3 \pm 2.5}$	$\mathbf{44.2 \pm 2.3}$
3P-(model-CF + SVM-CS)	44.0 ± 4.0	119.6 ± 5.8	57.4 ± 2.4	43.8 ± 2.2
3P-(memory-CF + SVM-CS)	$\mathbf{47.0 \pm 3.6}$	$\mathbf{122.2 \pm 5.8}$	$\mathbf{58.3 \pm 2.2}$	$\mathbf{44.2 \pm 2.2}$
3P-(model-CF + NN-CS)	43.9 ± 4.0	119.1 ± 5.7	57.3 ± 2.5	43.8 ± 2.2
3P-(memory-CF + NN-CS)	$\mathbf{46.9 \pm 3.5}$	$\mathbf{121.9 \pm 5.4}$	$\mathbf{58.6 \pm 2.3}$	$\mathbf{44.2 \pm 2.2}$

Cold start performance on OpenML

Method	Avg Error Rate
Oracle	0.121 ± 0.000
SingleBest	$\mathbf{0.170 \pm 0.000}$
Random	0.253 ± 0.000
memory-CF + SVM-CS	0.180 ± 0.008
memory-CF + NN-CS	0.184 ± 0.008
3P-SingleBest	0.166 ± 0.000
3P-Random	0.179 ± 0.000
3P-(memory-CF + SVM-CS)	$\mathbf{0.160 \pm 0.004}$
3P-(memory-CF + NN-CS)	0.163 ± 0.003

Not much margin of improvement: single best close to oracle.

Overview

Context

ALORS: Algorithm Recommender System

Empirical evaluation
- Collaborative filtering performance
- Cold start performance

Visualizing the problem/alg landscape

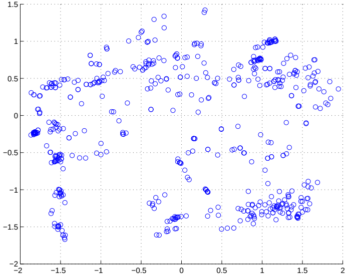
Where we learn something about the field

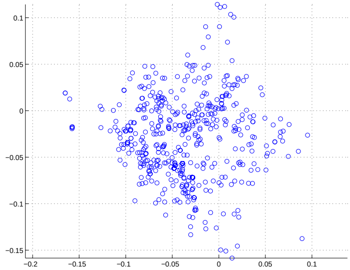
Each pb instance: a vector in $\mathbb{R}^d$; mapped onto $\mathbb{R}^2$ using Multi-dimensional scaling.

Left: initial features

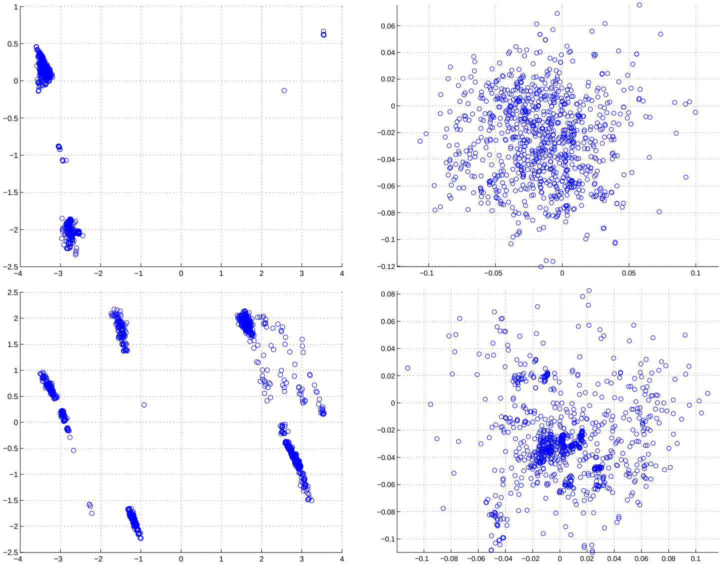
Right: Latent features

On SAT

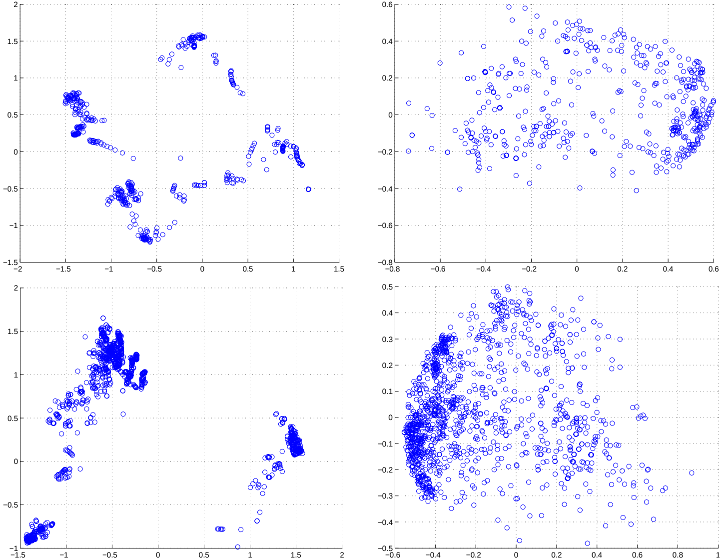




On SAT, followd

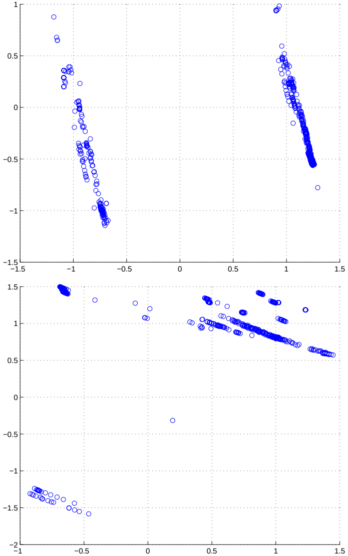


On CSP

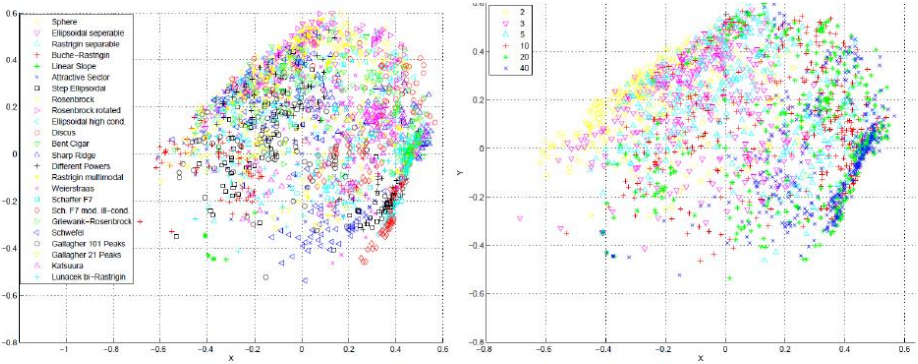


2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−1.5
−1
−0.5
0
0.5
1
1.5
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
2
1.5
1
0.5
0
−0.5
−1
−1.5
−1
−0.5
0
0.5
1
1.5
2
0.5
0.4
0.3
0.2
0.1
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1

On CSP, followed

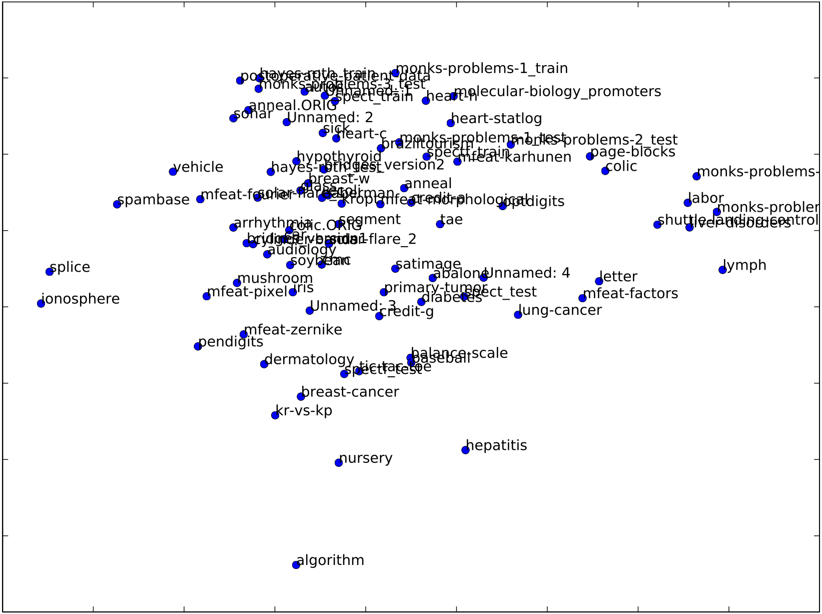


On Black-Box Optimization functions



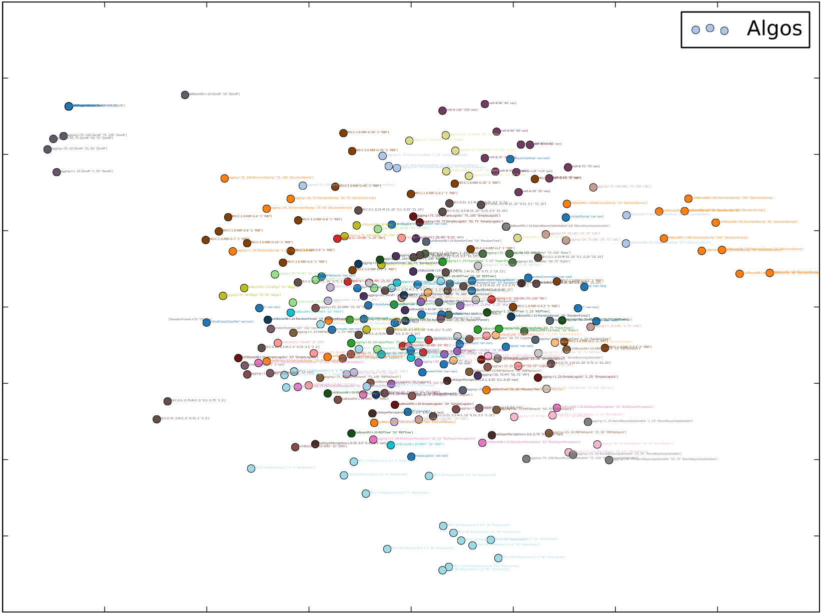
On OpenML

Datasets



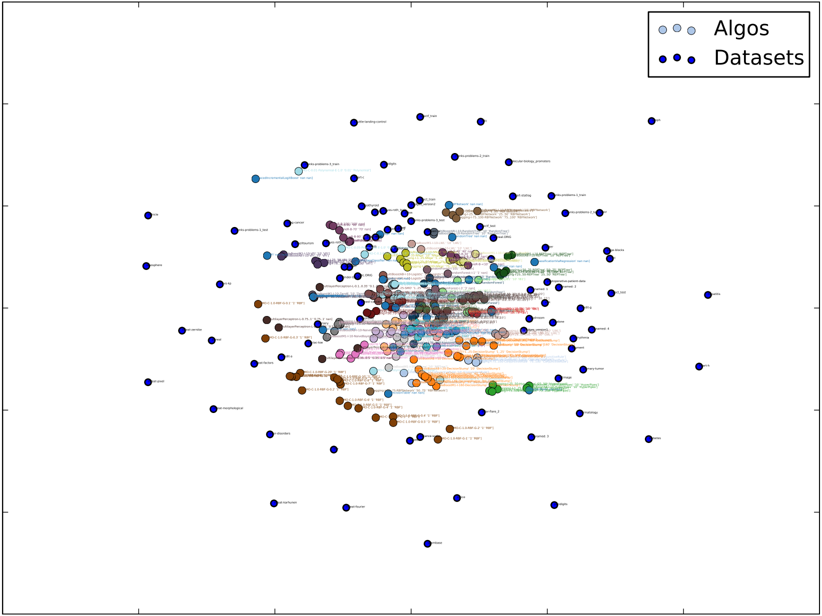
On OpenML, 2

Algorithms



On OpenML, 3

Algorithms and Datasets



Conclusion

- Algorithm recommender system works
- Cold start requires initial features
  - These can be poorly informative (BBOB)
  - Current ML features are not informative enough
- Provides educated (latent) features

Short and mid-term perspectives

Use latent features in order to

- Assess a benchmark suite (diversity);
- Assess a validation procedure (coverage of the benchmark suite used to validate a new algorithm)
- Assess novelty of an algorithm

Learn descriptive features

- using clusters based on latent features

Longer-term perspectives

- Intrinsic description of alg / problems
- Certification of portfolios
- Understand what makes it hard (new cues for parameterized complexity)

A typology of problems and algorithms