A Ranking Based Co-Optimization Parameter Tuning Framework

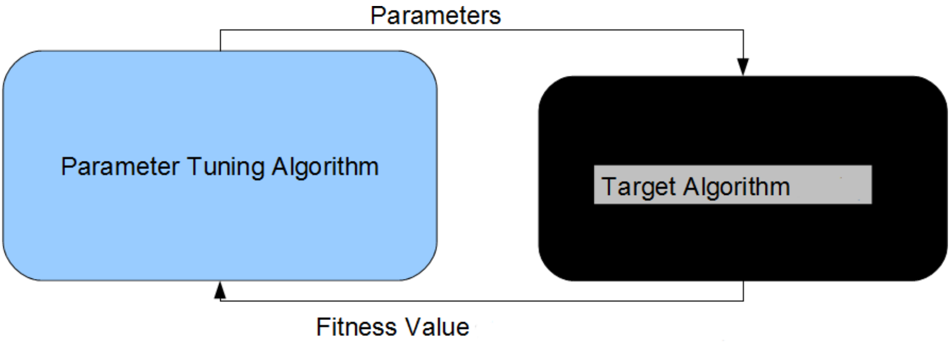
Rémi Bardenet and Mátyás Brendel

CNRS LAL, France

Siminole Meeting, 26th January 2011

Outline

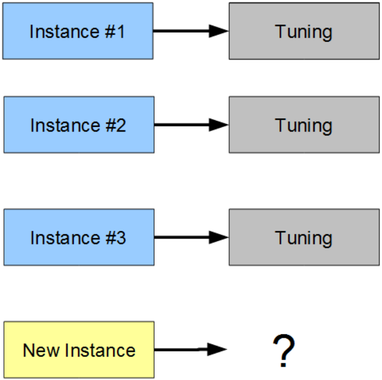
The Classical (Static) Parameter Tuning Scenario



Target Algorithm

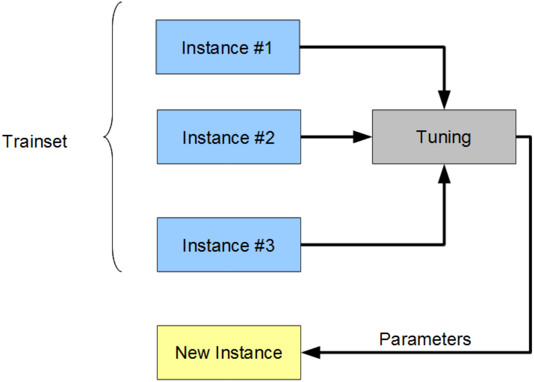
Target algorithm can be optimization or classification with generalization error to optimize.

Per Instance Static Tuning



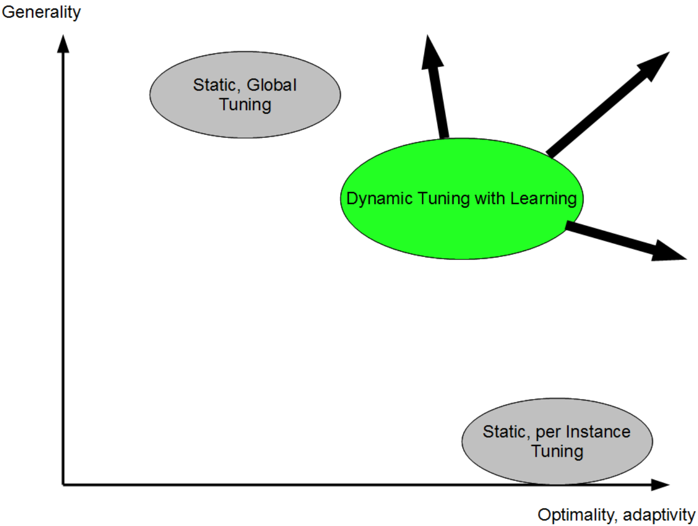
- Immense computational cost for many instances
- Does not aim at generalization to new instances

Global Static Tuning



- Huge information loss
- Global parameter is mediocre: clearly suboptimal for almost every instances

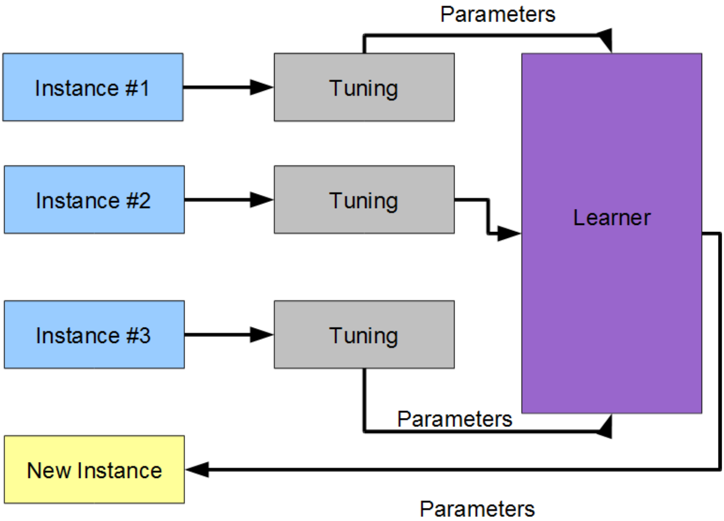
Optimality versus Generality



Interregnum

Remi presents results with a GP surrogate method parameter tuning applied to Deep Belief Networks.

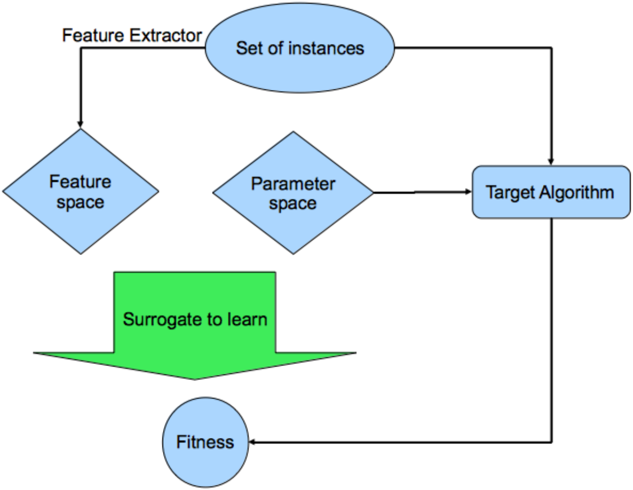
An Incomplete Sketch of the Novum



The Role of features

- Something is missing to telling apart problem-instances
- Some input is missing about the new problem-instance, we could work with.
- Solution: we need features to describe problem-instances.

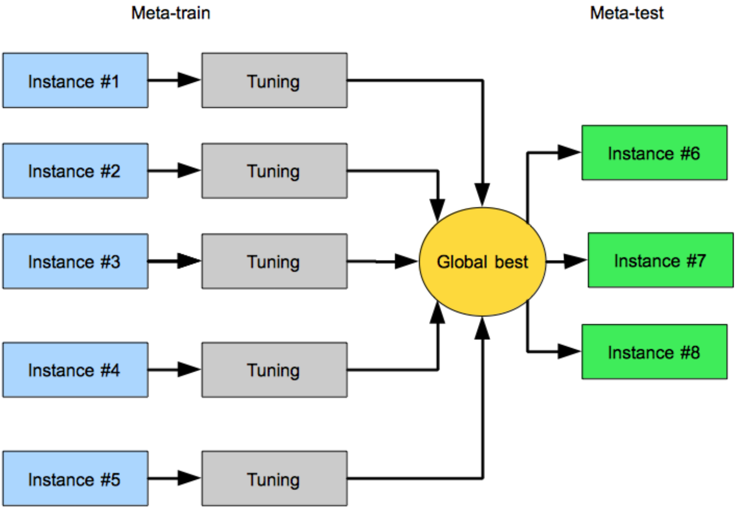
Learning a Feature+Parameters to Fitness Model



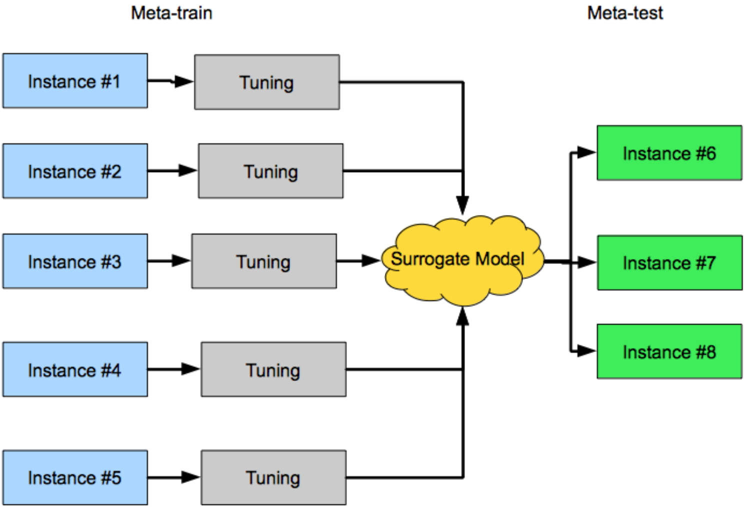
Description of 4 experiments

- D) Silly Weka user: Take the single parameter-set best on meta-train and use it for test.
- C) Experienced Weka User Last Minute Submission: train model on meta-train, use it for test.
- B) Beginner graduate: starting from parameters of D, tune meta-test problems separately, without features, by surrogate optimization.
- A) Experienced Graduate: starting form model C tune problems by a common surrogate with features.

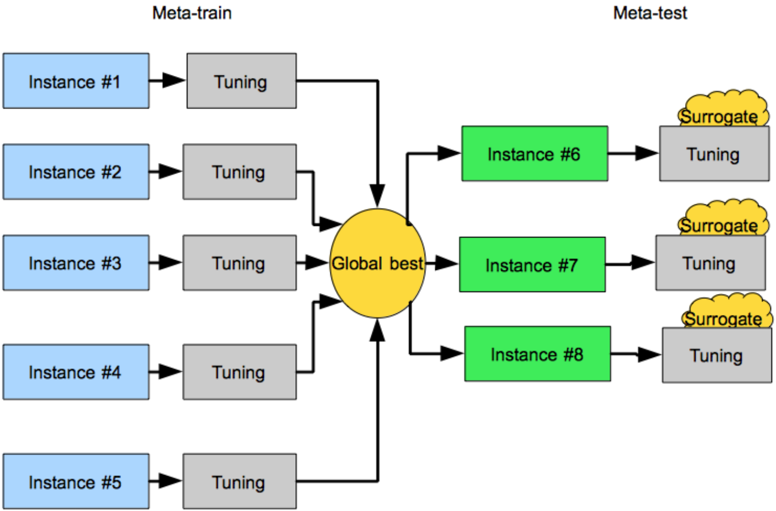
D) Silly Weka User



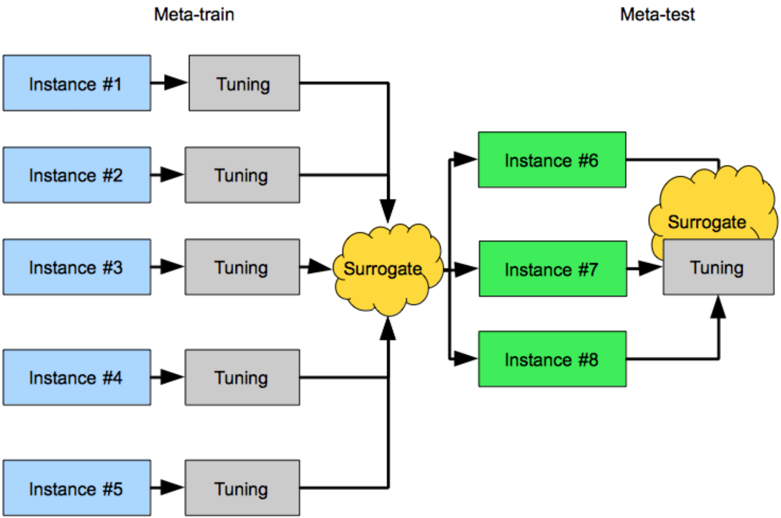
C) Last Minute Experienced



Beginner Graduate



Experienced Graduate



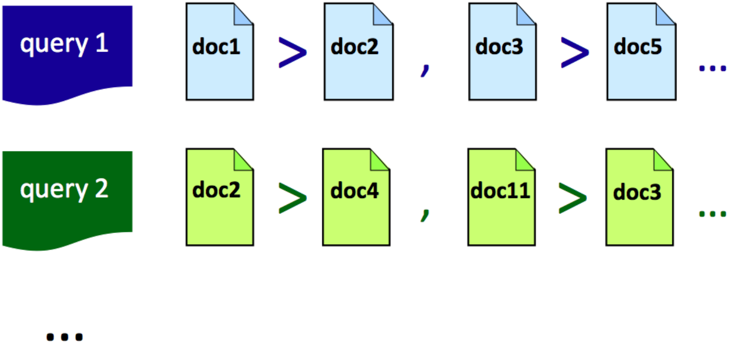
What is at stake with the 4 experiments?

- "Last minute experienced" shall be better than "silly Weka", testing whether the features and the surrogate is good of any value.
- In experiment B) ("beginner graduate") we will compare surrogate optimization to random search. This is just a sanity check. Not a new result.
- Comparing "beginner graduate" to "experienced graduate" tests again if features and surrogate is of any value. Moreover, it tests if co-optimization is better than independent optimizations.
- Nota bene: C) and D) are totally not comparable to A) and B), because in A and B we carry out many evaluations on meta-test.

General, top-level Validation Method

Meta-train and meta-test is done by 5-fold cross-validation splits of problems (not data!) of the overall repository.

Subset Ranking



Reference

Figure taken from Agarwal [2010]

SVMRank

- Problem formulation comes from subset ranking from query based information retrieval.
- This is suitable for us if queries are interpreted to correspond to problem-instances.

Problem instances

- UCI Machine Learning Repository, we could make use of 27 problems.
- Kent Ridge Bio-medical Dataset, we will be able to use 16 problems.
- Multi-class protein fold recognition data set, we will be able to use 6 problems.

Parameters of Multiboost

Name	Values
Number of leaves	1 2 5 10 20 50 100
Number of products	2 3 4 5 7 10 15 20 30
Number of Iterations	2 5 10 20 50 100 200 500 1000 2000 5000 10000

Table: Multiboost parameters that are controlled by tuning. Actually the parameters are transformed to a logarithmic scale and normalized. Number of leaves or products has to be tuned depending on the type of the basic learner in multiboost.

Precomputed data

A grid is created as the Cartesian product of these values. The precomputed values of this grid is used for training, for testing, but the same grid is also used when the expected improvement is optimized.

Features

Name	minimum	mean	maximum
log number of attributes	0.7	1.65	2.38
number of classes	2	26	6
log instances per attributes	1	3.9	7.05
PCA reduction rate	0.14	0.63	0.86

Table: Minimum, mean and maximum values of the features in the UCI domain. Features are normalized before training.

Some results

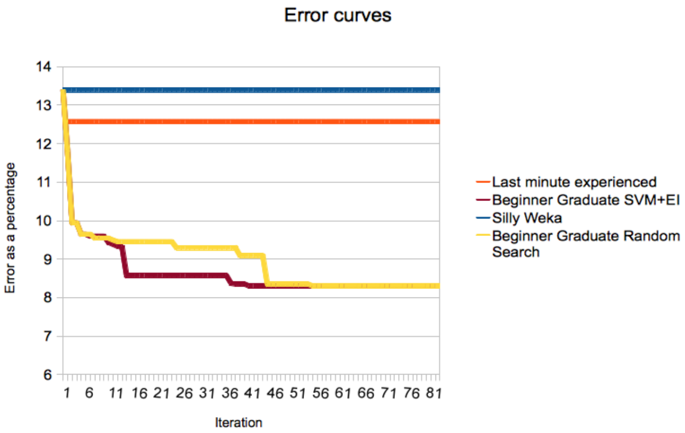
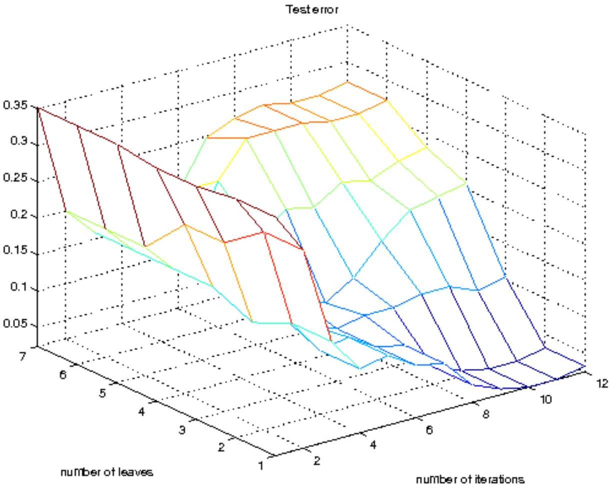


Figure: The figure is done only with a 80-20 split, not very reliable yet.

Regression Values of the Parameters

(Loading images/mesh.avi)

Real fitness values of the parameters



Sigma Values of the Parameters

(Loading images/sigmamesh.avi)

▸ Sigma values as the function of the parameter values for problem "balance-scale"

Non-conclusional conclusions of the results

- "Last minute experienced" is better than "silly Weka". So there is something to look for.
- In "beginner graduate", SVM+EI is better than random search. That is expected, we have seen this in other papers.
- No other conclusions, we are far from ready.

The Scope of our Method

- Our parameter tuning method is a framework
- It can be applied to any optimization or classification algorithm to be tuned, which has different instances (problems) to solve and some features can be extracted
- Surrogate can be RankSVM, RankGP, or any ranking method. In some cases SVM or GP, or any other nonlinear regression method.

Related Fields

- Multi-task learning, but we have features
- Portfolio-learning in AI Planning (e.g.Roberts et al. [2007]), we have parameters instead of portfolio, they use all kinds of Weka models

Computational limits

- Currently meta train-set consists of 7 (values of N) x12 (values of T) x22 (problems) points, i.e. 1848 lines.
- Computing a surrogate of this size takes some time. For SVM it is few dozens of minutes, but with GP is it much longer. With GP we are on the limits.
- The precomputed values take a few weak using several multi-core servers.
- A framework, where parameter values are free is possible with a scheduling algorithm, similar to Brendel and Schoenauer [2011]

Limits of the Current Data Repository

- 30-50 problems, 4 features, 2 parameters
- More problems would be more interesting. More features would be more fruitful.
- But we are on the limits with this number of problems.

Ongoing and Future Work

- Use all repositories
- Finnish multi-problem experiment. (A)
- Test RankGP
- Test other target classifiers, like SVM, and multiboost with productlearner.
- Get more problems
- This enables to extend features
- And to tune more parameters

Thank You

Questions?

Machine Learning Formalism

- n problem-instances: i=1,2,..n, each described by d=5 features
- One solver (multiboost), with d'=2 parameters
- $f_i \in \mathbb{R}^{d'}$ = features, describing instance i
- $p_{i,j} \in \mathbb{R}^{d}$ = parameter values of instance i applied in iteration j
- $x_{i,j} \in \mathbb{R}^{d+d'}$ = input (parameters +features) values
- $y_{i,j} \in \mathbb{R}$ = desired output, label, obtained as the error measure of multiboost applied to problem i with parameters $p_{i,j}$
- Meta-train-set: $\{(x_{i,j}, y_{i,j}) : i = 1, 2 \ldots n, j = 1, 2, \ldots\}$
- Meta-test-set: Similar to meta-train, but with different problems

SVMRank Formulaton Outline

- We are looking for weights w such that.
- If $x_{i,j} \prec x_{i,k}$ then $wx_{i,j} > wx_{i,k}$ shall hold
- Reformulate with slack variables: $wx_{i,j} - wx_{i,k} > 1 - \xi_{i,j,k}$
- Non-linear SVMRank: transformation to the kernel space is added. We use RBF, with parameters C and gamma to tune by grid-search with a cross-validation on meta train. Yes, again a tuning problem.

Implementation

Implementation taken from Joachims [2002]

Sh. Agarwal. Ranking methods in machine learning. In *SIAM Cnference on Data Mining (SDM10)*, 2010.

M. Roberts, A. Howe, and L. Flom. Learned models of performance for many planners. In *Working Notes of ICAPS 2007*, 2007.

M. Brendel and M. Schoenauer. Instance-based parameter tuning for evolutionary ai planning. In *Proceedings of the 20th Genetic and Evolutionary Computation Conference*, 2011.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.