Wrapper-Filter Feature Selection Algorithm Using a Memetic Framework

[Z. Zhu, Y. Ong, and M. Dash]

Zombie + PSO

Filipe Mutz

Schedule

- Memetic Framework
- PSO Zombie
- Comparative Results

Memetic Algorithm

Memetic = Evolutionary + Local Search

“Memetic Algorithms are **population-based** metaheuristics (...) inspired by Darwin’s principles of natural evolution and Dawkins’ notion of a meme defined as a unit of cultural evolution that is capable of **local refinement**.”

(Zhu et al., 2007)

Memetic Algorithm

Procedure

1 **Begin**
2 **Initialize:** Randomly generate an initial population;
3 **While** (Stopping conditions are not satisfied)
4 Evaluate all the population;
5 **For** each subset chosen to undergo the local improvement process
6 Perform local search and replace it with locally improved solution in the spirit of Lamarckian learning;
7 **End For**
8 Perform evolutionary operators based on selection, crossover, and mutation;
9 **End While**
10 **End**

Local Search Types

Run one of the following steps for some iterations:

- Binary: Flip a bit according to a criteria. Keep the new bit pattern if it improved the solution.
- Continuous: Add a gaussian noise to a selected dimension of the particle. Keep the change if it improved the solution.

Problem Representation

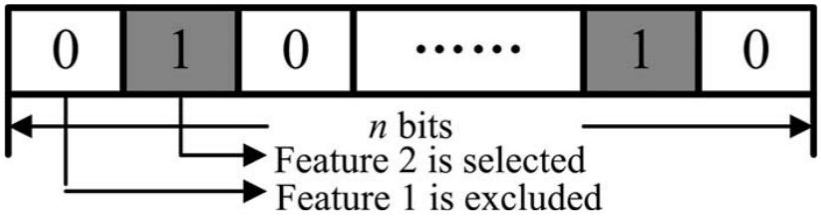
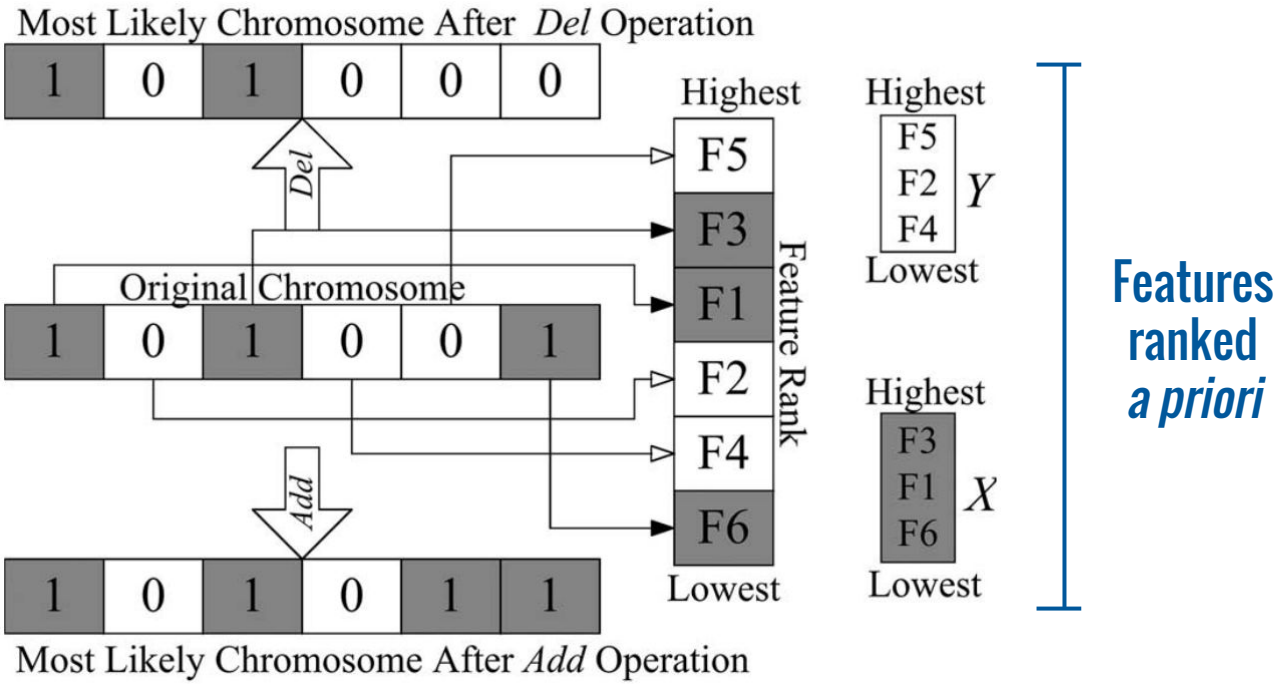


Fig. 2. Representation of chromosome as a binary bit string.

The objective function is defined by the classification accuracy or feature set reduction with a given max. loss

Local Search Step



Local Search Parameters

Intensity of LS is quantified by:

L : Local search length. Maximum number of ADD/DEL operations in each LS.

W : Local search interval. Specifies the number of elite chromosomes that undergo the improvement procedure in each generation.

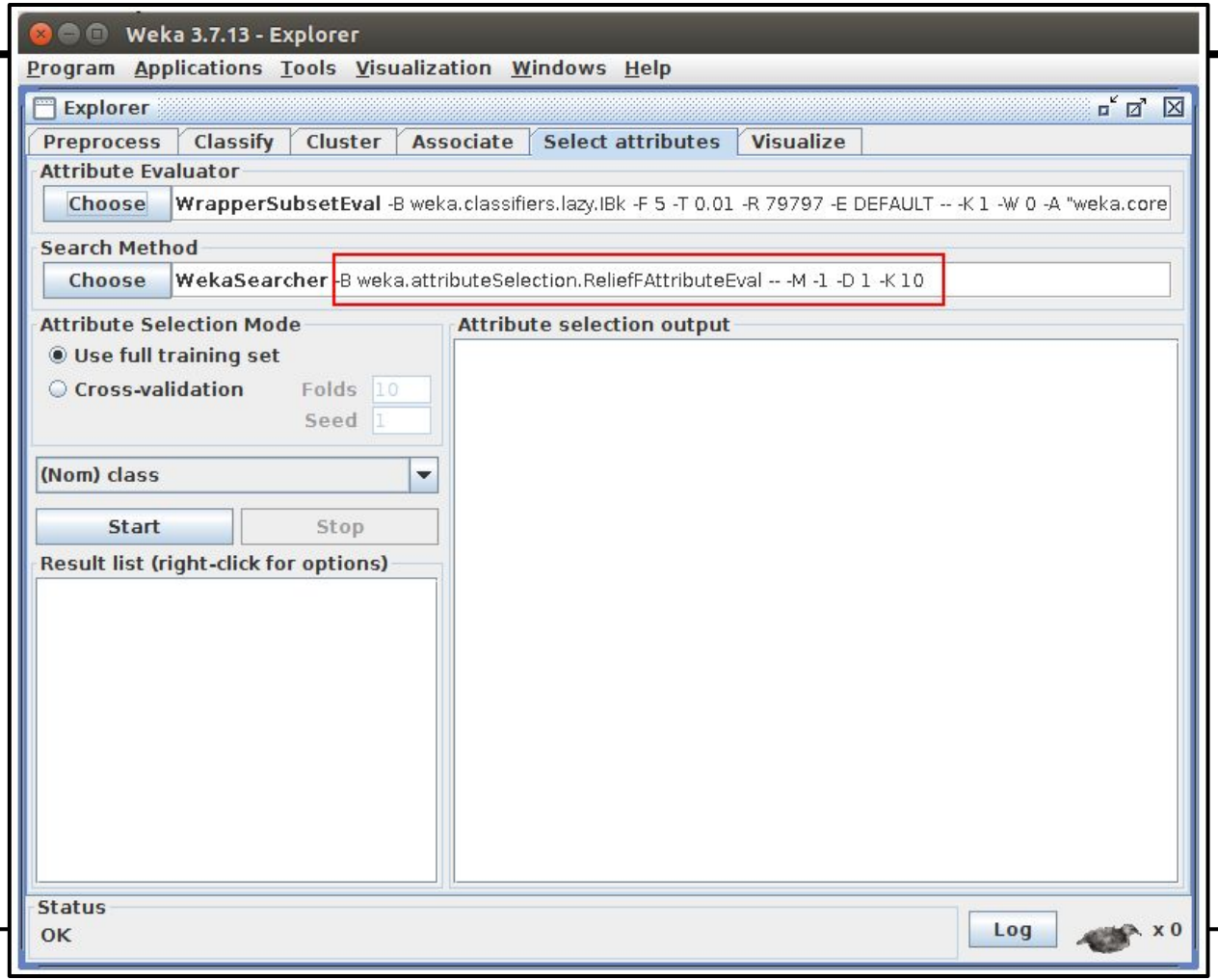
Local Search Strategies

[Impl. in WEKA]

Rankers	Stop Conditions
ReliefF (-) if different in KNN of the same class, and (+) if different in KNN of other classes.	Improvement First Stop as soon as the first improvement is found.
Gain Ratio Compare the general data entropy and the entropy given a subset of features.	Greedy Test all the L^2 possible combinations of ADD/DEL operations and select the best.
Chi-Square Uses the Chi^2 metric to discretize continuous variables. At the end, if a feature has a single possible value, then it is not relevant.	Sequential ADD the most relevant feature and DEL the least significant feature.

Implementation

- weka.attributeSelection
  - AddDelIMutation.java [extends Mutation]
    - Object execute(Object object);
  - ObjectiveFunction.java [extends Problem]
    - void evaluate(Solution sol);
  - Optimizer.java
    - int[] run(SubsetEvaluator featureSetEvaluator, ASEvaluation featureRanker, Instances data);
  - WekaSearcher.java [extends ASSearch implements OptionHandler]
    - int[] search(ASEvaluation ASEval, Instances data);
    - Enumeration<Option> listOptions();
    - void setOptions(String[] options);
    - void resetOptions();
    - String[] getOptions();


Weka 3.7.13 - Explorer
Program Applications Tools Visualization Windows Help
Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Attribute Evaluator
Choose WrapperSubsetEval -B weka.classifiers.lazy.IBk -F 5 -T 0.01 -R 79797 -E DEFAULT -- -K 1 -W 0 -A "weka.core
Search Method
Choose WekaSearcher -B weka.attributeSelection.ReliefFAttributeEval -- -M -1 -D 1 -K 10
Attribute Selection Mode
Use full training set
Cross-validation
Folds 10
Seed 1
Attribute selection output
(Nom) class
Start
Stop
Result list (right-click for options)
Status
OK
Log
x 0

PSO + Zombie

PSO	PSO + Zombie
P_{best}	Humans naturally go to P_{best} with LS
G_{best}	The most promising region is stored now
P_{best} to G_{best} random motion	With a given probability (0.6), humans get closer to promising areas (a bit is made equals)
-	With a given probability (0.1), zombies chase the closest human instead of making random moves (relevant to end-game)

Implementation

- algorithms
  - BinaryZombieSearch.java [+ PSO]
    - BitSet run();
  - SpecifierInterface.java
- weka.attributeSelection
  - FeatureSelectionSpecifier.java [extends Specifier]
    - double calcSubsetMerit(BitSet selection);
    - int numAttributes();
    - int numDimensions();
  - Optimizer.java
  - WekaSearcher.java [extends ASSearch implements OptionHandler]

Results

Base	GA + LS [Best Result]	Zombie Swarm [Best Result]
German Credit	0.733 [6]	**0.7415 [3]**
Ionosphere	0.94301 [8]	**0.9487 [6]**
Supermarket	**0.5246 [82]**	0.452777 [78]
Sonar	0.92692 [27]	**0.93269 [27]**

Questions?