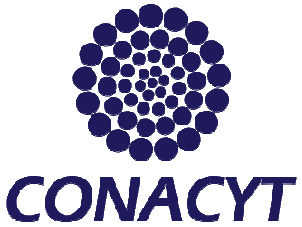
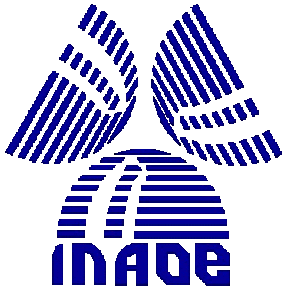




Machine Learning through Genetic Programming

Hugo Jair Escalante

hugojair@inaoep.mx
http://ccc.inaoep.mx/~hugojair/

Laboratorio de
Tecnologías del Lenguaje
Ciencias Computacionales, INAOE

Tonantzintla, Puebla, 2017

Contents

Machine Learning through Genetic Programming

EVOLUTIONARY COMPUTATION

Evolutionary Computing

- EC Is the collective name for a range of **problem-solving** techniques based on principles of biological evolution, such as **natural selection** and **genetic inheritance**. These techniques are being increasingly widely applied to a variety of problems, ranging from practical applications in industry and commerce to leading-edge scientific research.

A.E. Eiben and J.E. Smith, **Introduction to Evolutionary Computing,** Springer, Natural Computing Series 1st edition, 2003, ISBN: 3-540-40184-9, Corr. 2nd printing, 2007, ISBN: 978-3-540-40184-1

Evolutionary Computing

- Trial and error problem solving approach:
  - While not_satisfied_with_solution
    1. Generate candidate solution(s) for the problem at hand
    2. Evaluate the quality of the candidate solution (s)
  - Return best_solution_found

EC techniques generate new solutions according to (rough) analogies with biological evolution principles

The Main Evolutionary Computing Metaphor

A given environment is filled with a population of individuals that strive for survival and reproduction

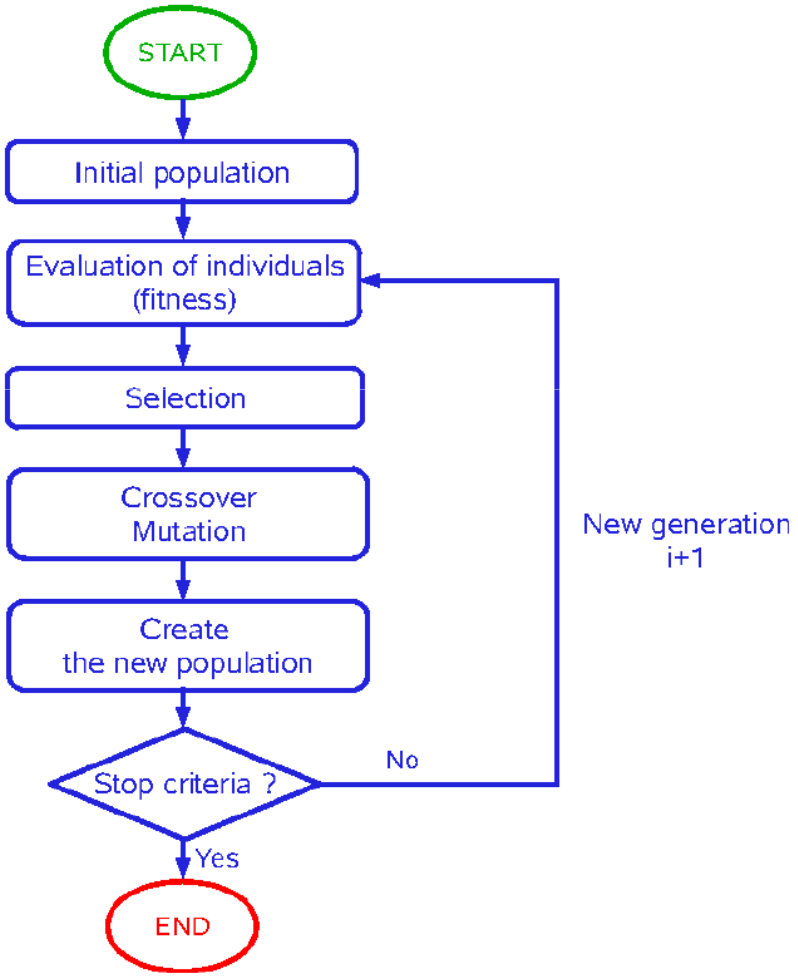
A problem is to be solved via stochastic trial-and-error, using a starting point a set of candidate solutions

EVOLUTION		**PROBLEM SOLVING**
Environment	⟷	Problem (search space)
Individual	⟷	Candidate Solution
Fitness	⟷	Quality of the solution

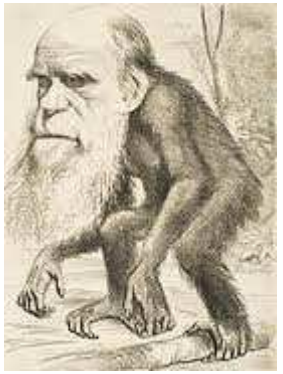
Fitness → chances for survival and reproduction

Quality → chance for seeding new solutions

Generic EA

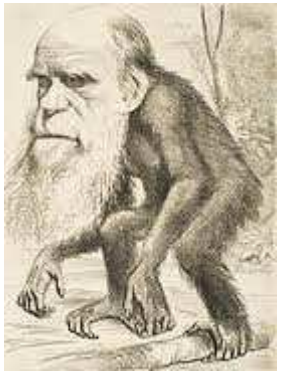

START
Initial population
Evaluation of individuals (fitness)
Selection
Crossover
Mutation
Create the new population
Stop criteria ?
No
Yes
END
New generation i+1

Darwinian Evolution 1: Survival of the fittest



- All environments have finite resources
  (i.e., can only support a limited number of individuals)
- Lifeforms have basic instinct/ lifecycles geared towards reproduction
- Therefore some kind of selection is inevitable
- Those individuals that compete for the resources most effectively have increased chance of reproduction

Darwinian Evolution 2: Diversity drives change



- Phenotypic traits:
  - Behaviour / physical differences that affect response to environment
  - Partly determined by inheritance, partly by factors during development
  - Unique to each individual, partly as a result of random changes
- If phenotypic traits:
  - Lead to higher chances of reproduction
  - Can be inherited

  then they will tend to increase in subsequent generations,
- leading to new combinations of traits …

Darwinian Evolution: Summary

- Population consists of diverse set of individuals

- Combinations of traits that are better adapted tend to increase representation in population

  Individuals are “units of selection”

- Variations occur through random changes yielding constant source of diversity, coupled with selection means that:

  Population is the “unit of evolution”

- Note the absence of “guiding force”

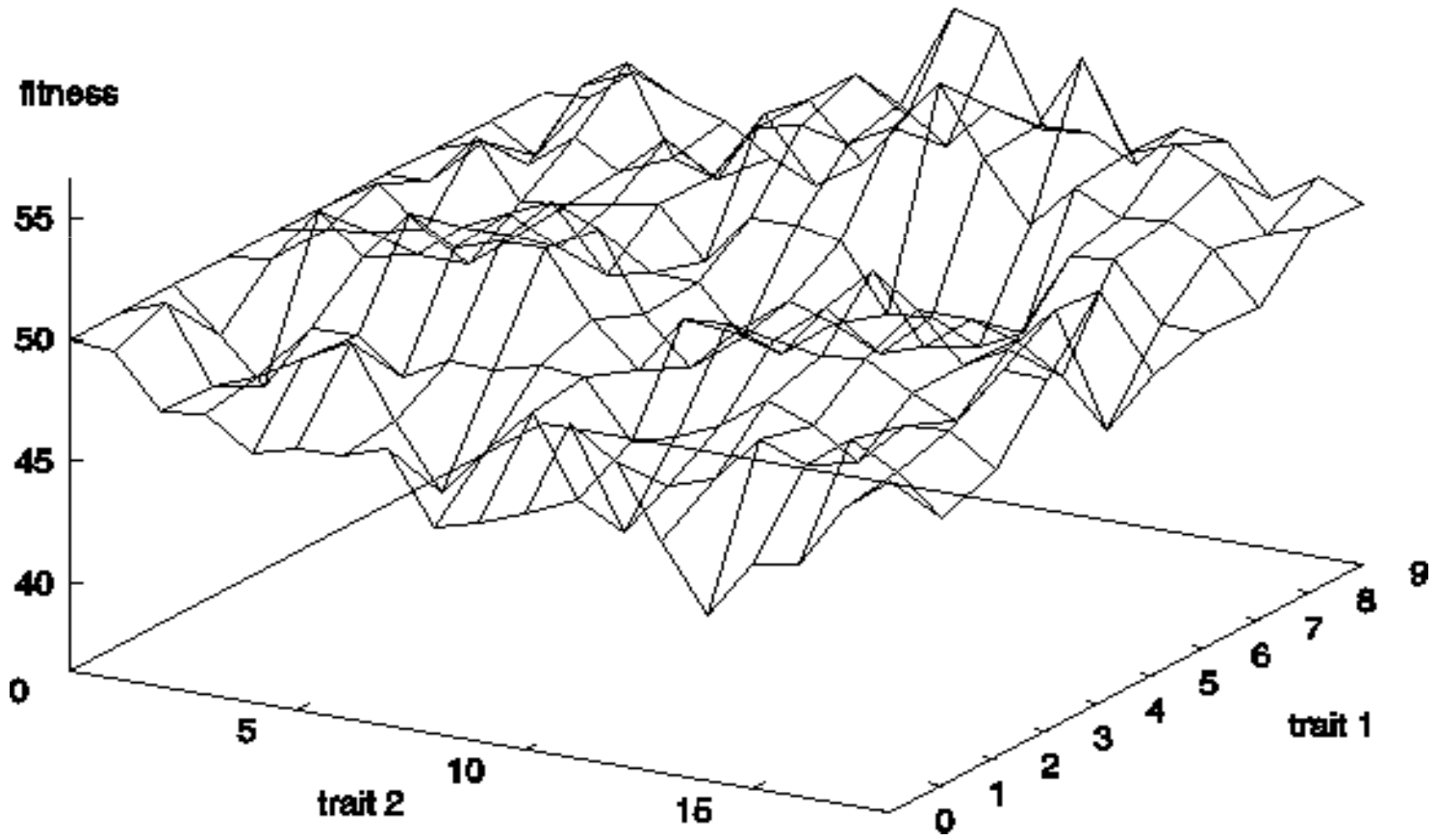
Adaptive landscape metaphor (Wright, 1932)

- Can envisage population with *n* traits as existing in a *n+1*-dimensional space (landscape) with height corresponding to fitness

- Each different individual (phenotype) represents a single point on the landscape

- Population is therefore a “cloud” of points, moving on the landscape over time as it evolves - adaptation

Example with two traits

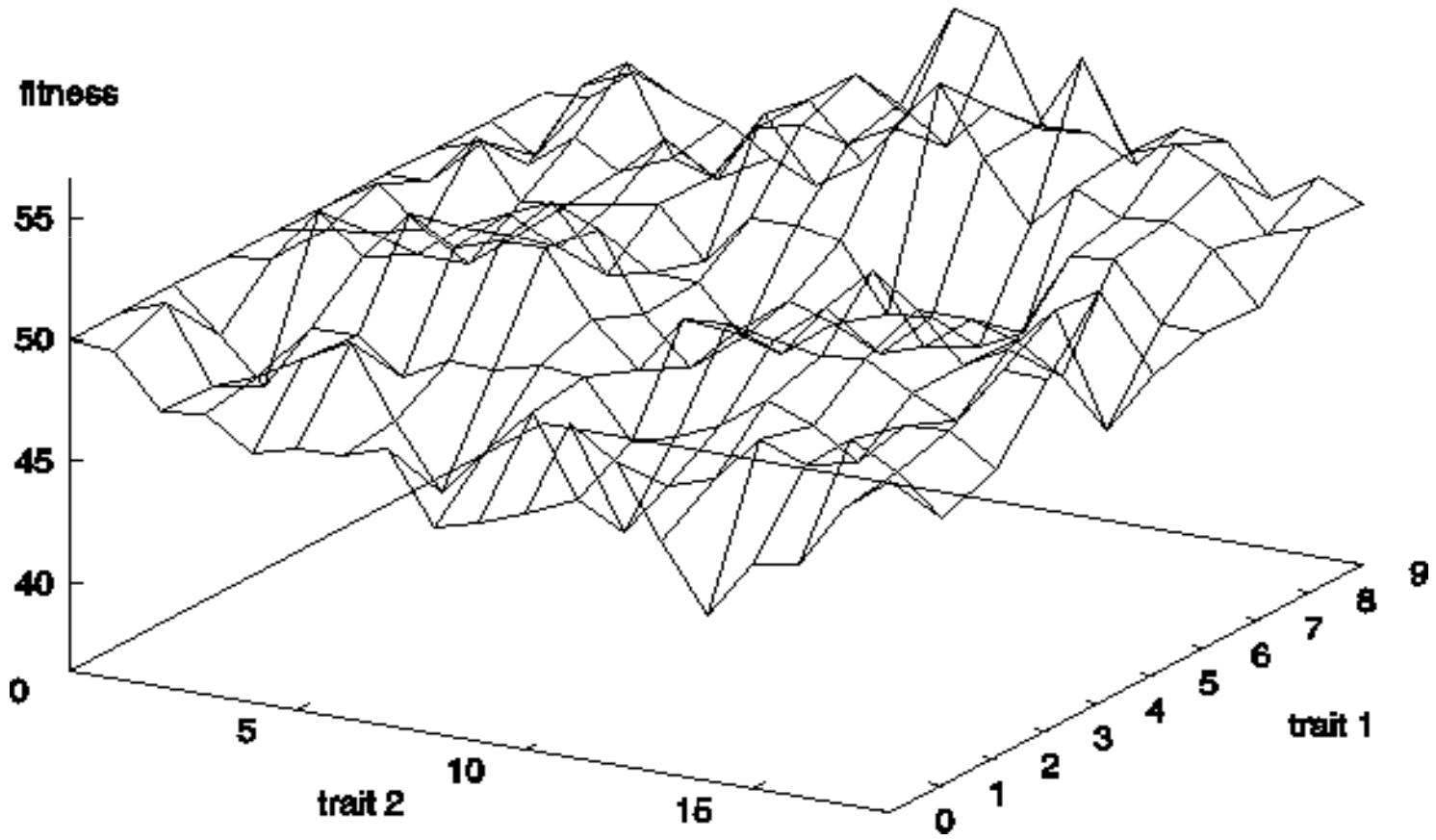


Adaptive landscape metaphor (cont'd)

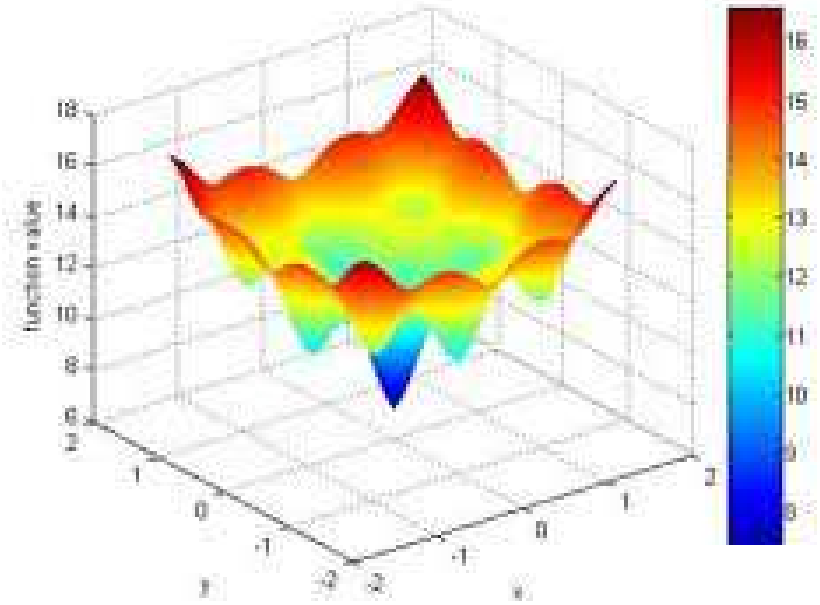
- Selection "pushes" population up the landscape

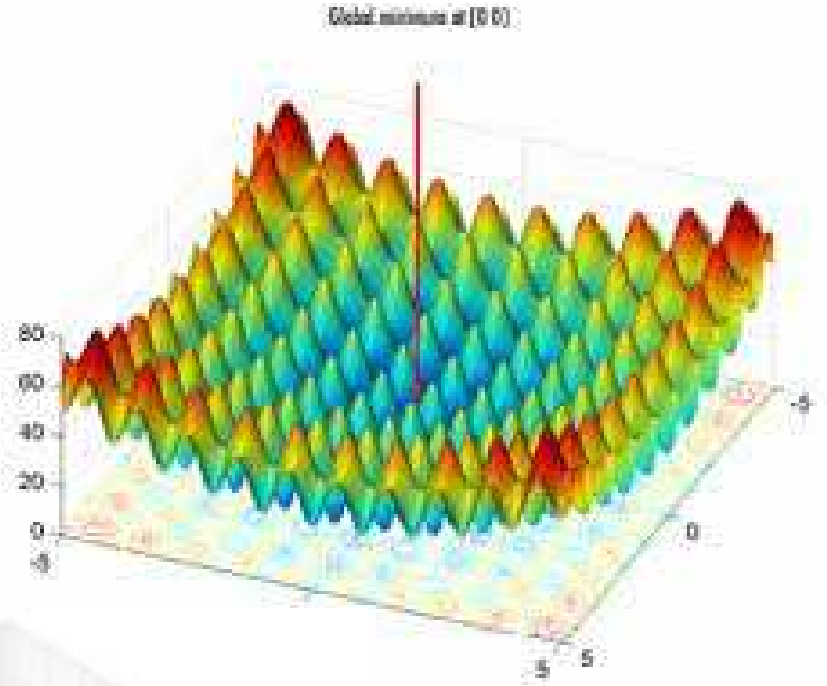
- Genetic drift:
  - random variations in feature distribution
    (+ or -) arising from sampling error
  - can cause the population "melt down" hills, thus crossing valleys and leaving local optima

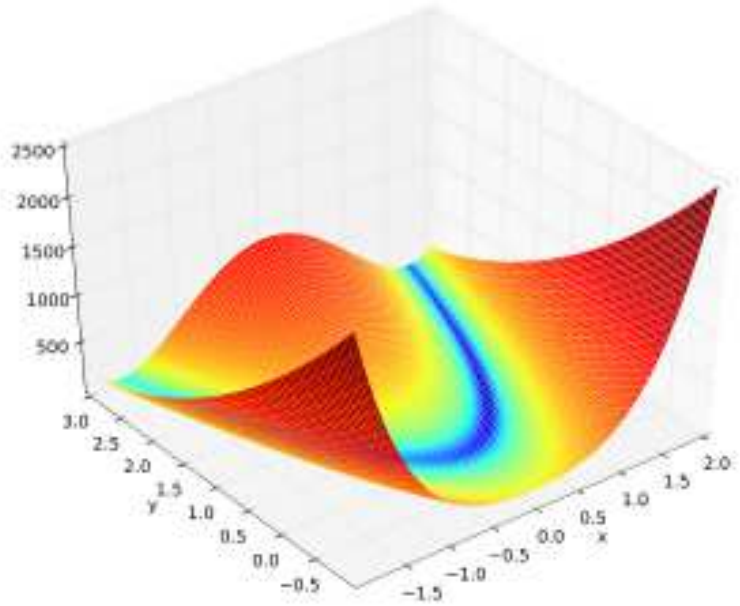
Example with two traits



Link to optimization







EAs in a nutshell

- A population of individuals exists in an environment with limited resources
- ***Competition*** for those resources causes selection of those ***fitter*** individuals that are better adapted to the environment
- These individuals act as seeds for the generation of new individuals through recombination and mutation

EAs in a nutshell

- The new individuals have their fitness evaluated and compete (possibly also with parents) for survival.

- Over time ***Natural selection*** causes a rise in the fitness of the population

EAs in a nutshell

- EAs fall into the category of “generate and test” algorithms
- They are stochastic, population-based algorithms
- Variation operators (recombination and mutation) create the necessary diversity and thereby facilitate novelty
- Selection reduces diversity and acts as a force pushing quality

Brief History: the ancestors

- 1948, Turing:
  proposes "genetical or evolutionary search"
- 1962, Bremermann
  optimization through evolution and recombination
- 1964, Rechenberg
  introduces evolution strategies
- 1965, L. Fogel, Owens and Walsh
  introduce evolutionary programming
- 1975, Holland
  introduces genetic algorithms
- 1992, Koza
  introduces genetic programming

Evolutionary computing

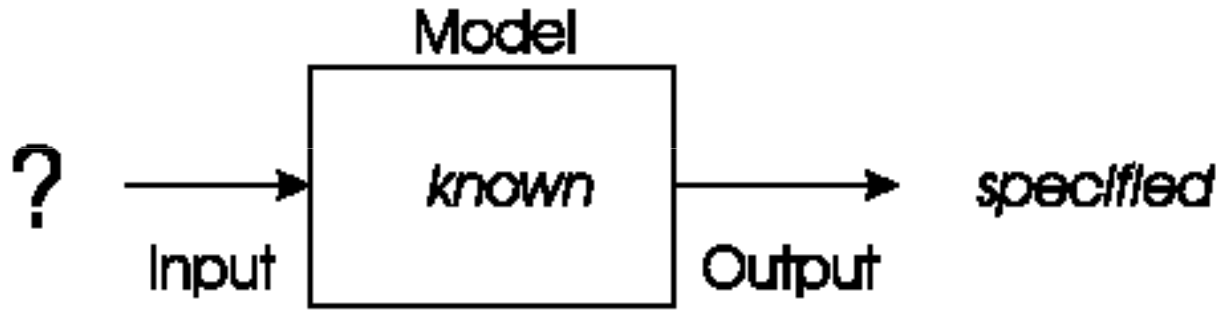
Adequate problems for EC

- (Hard) combinatorial optimization
- Multi-objective optimization
- Constrained optimization
- Tough (differentiable) objective functions
- Multimodal objective functions
- …

Domains/applications requiring fast response
NP- problems
Requiring (almost) accurate solutions

Problem type 1 : Optimization

- We have a model of our system and seek inputs that give us a specified goal



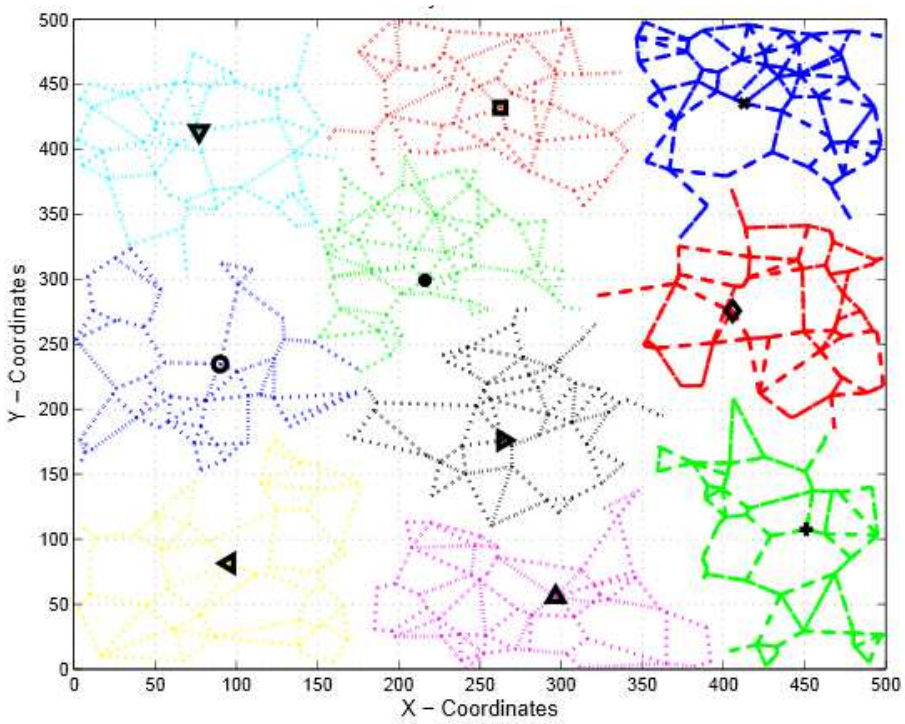
- e.g.
  - time tables for university, call center, or hospital
  - design specifications, etc etc

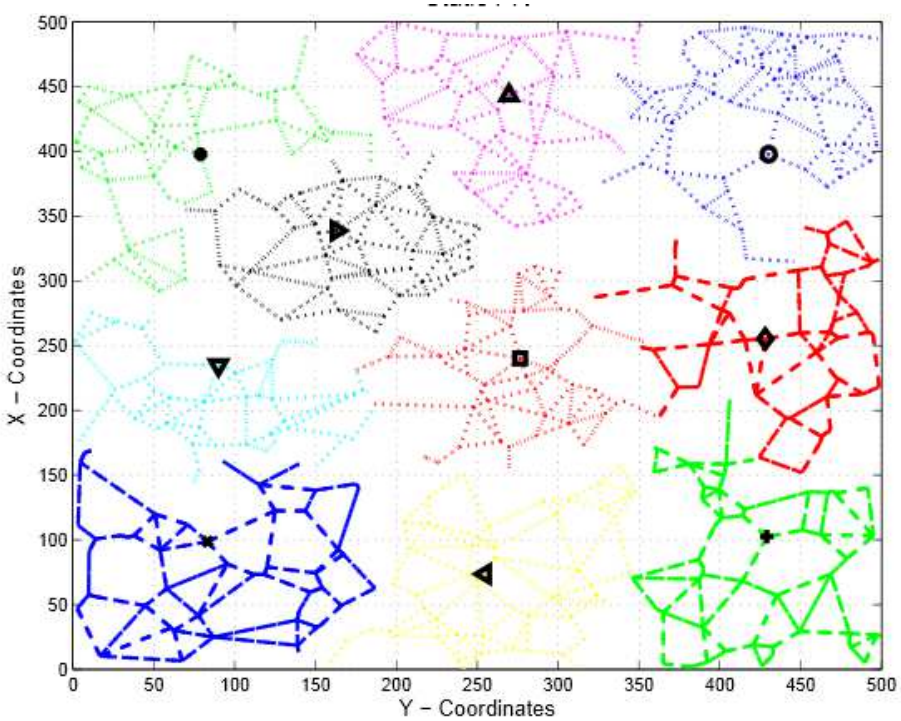
Problem type 1 : Optimization

$$\min_{X \in \Pi} \quad f(X) = \max_{k \in K} \max_{i \in X_k} \left\{ d_{c(k),i} \right\}$$

$$\text{subject to} \quad \frac{w^a(X_k)}{\mu^a} \in [1 - \tau^a, 1 + \tau^a] \qquad k \in K,\ a \in A$$

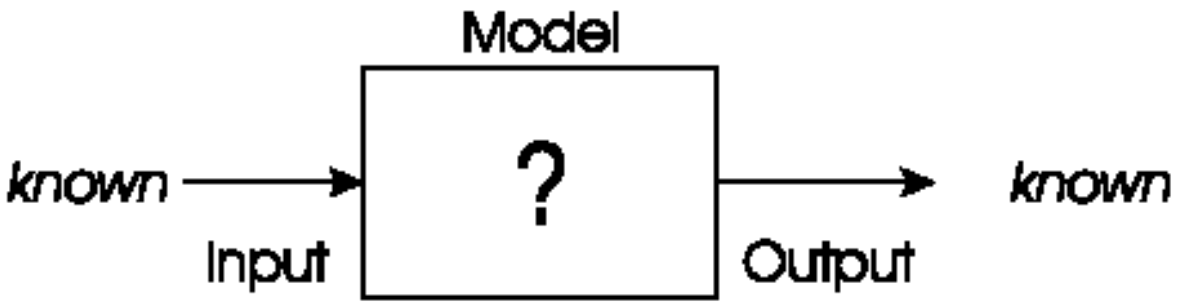
$$G_k = G(V_k, E(V_k)) \text{ is connected} \qquad k \in K$$





Problem types 2: Modelling

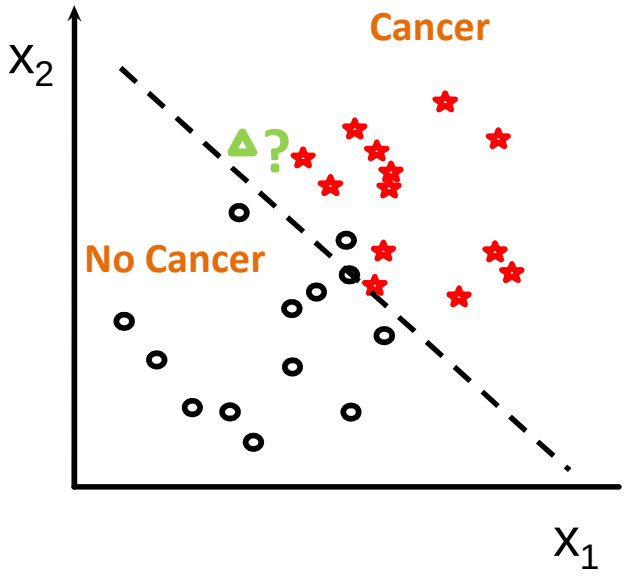
- We have corresponding sets of inputs & outputs and seek model that delivers correct output for every known input



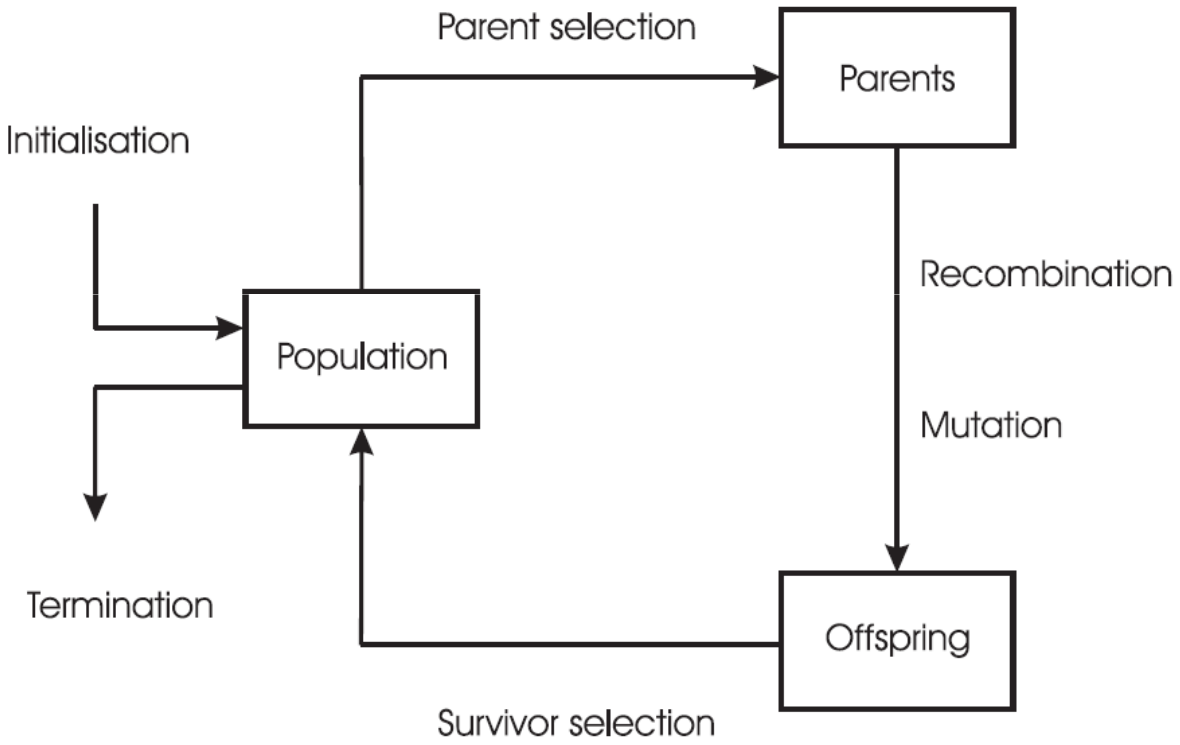
- Evolutionary machine learning
- Genetics-based machine learning

Pattern classification

- To learn a model able to make predictions regarding a variable of interest, using a set of other variables. E.g., classifying:
  - *Emails as spam vs safe-email*
  - *Tomographies as tumor vs no-tumor*
  - *Image faces as the identity of somebody*
  - *Scanned handwritten text as digits*



General Scheme of EAs



Pseudo-code for typical EA

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```

What are the different types of EAs

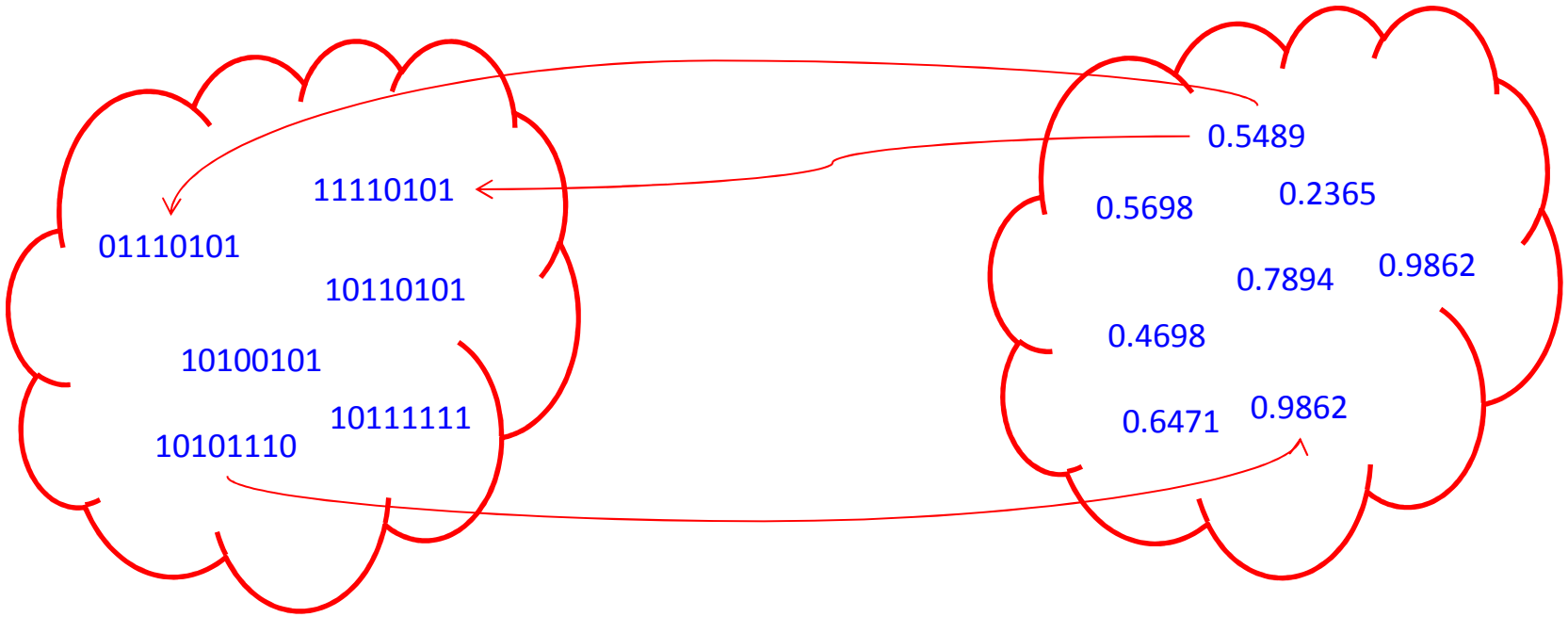
- Historically different flavours of EAs have been associated with different representations
  - Binary strings : Genetic Algorithms
  - Real-valued vectors : Evolution Strategies
  - Finite state Machines: Evolutionary Programming
  - Trees: Genetic Programming

What are the different types of EAs

- These differences are largely irrelevant, best strategy
  - choose representation to suit problem
  - choose variation operators to suit representation
- Selection operators only use fitness and so are independent of representation

Representations

- Candidate solutions (individuals) exist in *phenotype* space – solutions space
- They are encoded in chromosomes, which exist in *genotype* space – search space
  - Encoding : phenotype=> genotype (not necessarily one to one)
  - Decoding : genotype=> phenotype (must be one to one)



Representations

- Chromosomes contain genes, which are in (usually fixed) positions called loci (sing. locus) and have a value (allele)

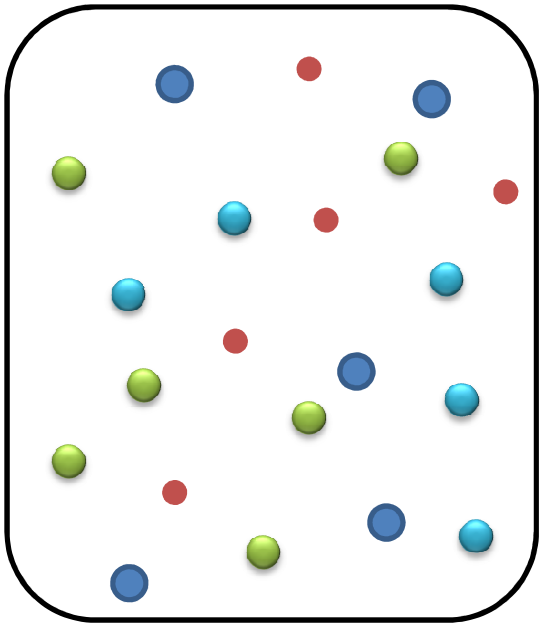
In order to find the global optimum, every feasible solution must be represented in genotype space

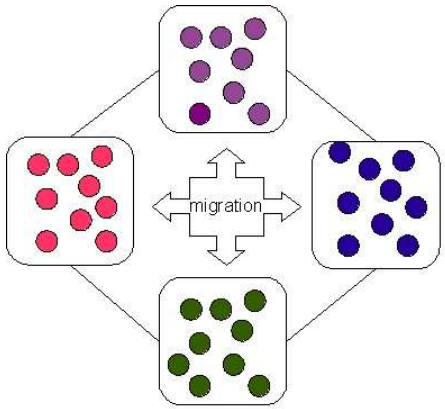
Evaluation (Fitness) Function

- Represents the requirements that the population should adapt to
- a.k.a. *quality* function or *objective* function
- Assigns a single real-valued fitness to each phenotype which forms the basis for selection
  - So the more discrimination (different values) the better
- Typically we talk about fitness being maximised
  - Some problems may be best posed as minimisation problems, but conversion is trivial

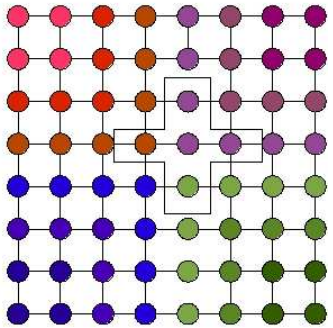
Population

- Holds (representations of) possible solutions
- Usually has a fixed size and is a *multiset* of genotypes
  - Some sophisticated EAs also assert a spatial structure on the population e.g., a grid.
  - Panmictic or centralized population – global interaction
  - Decentralized population – local interaction





Distributed PGA



Cellular GA

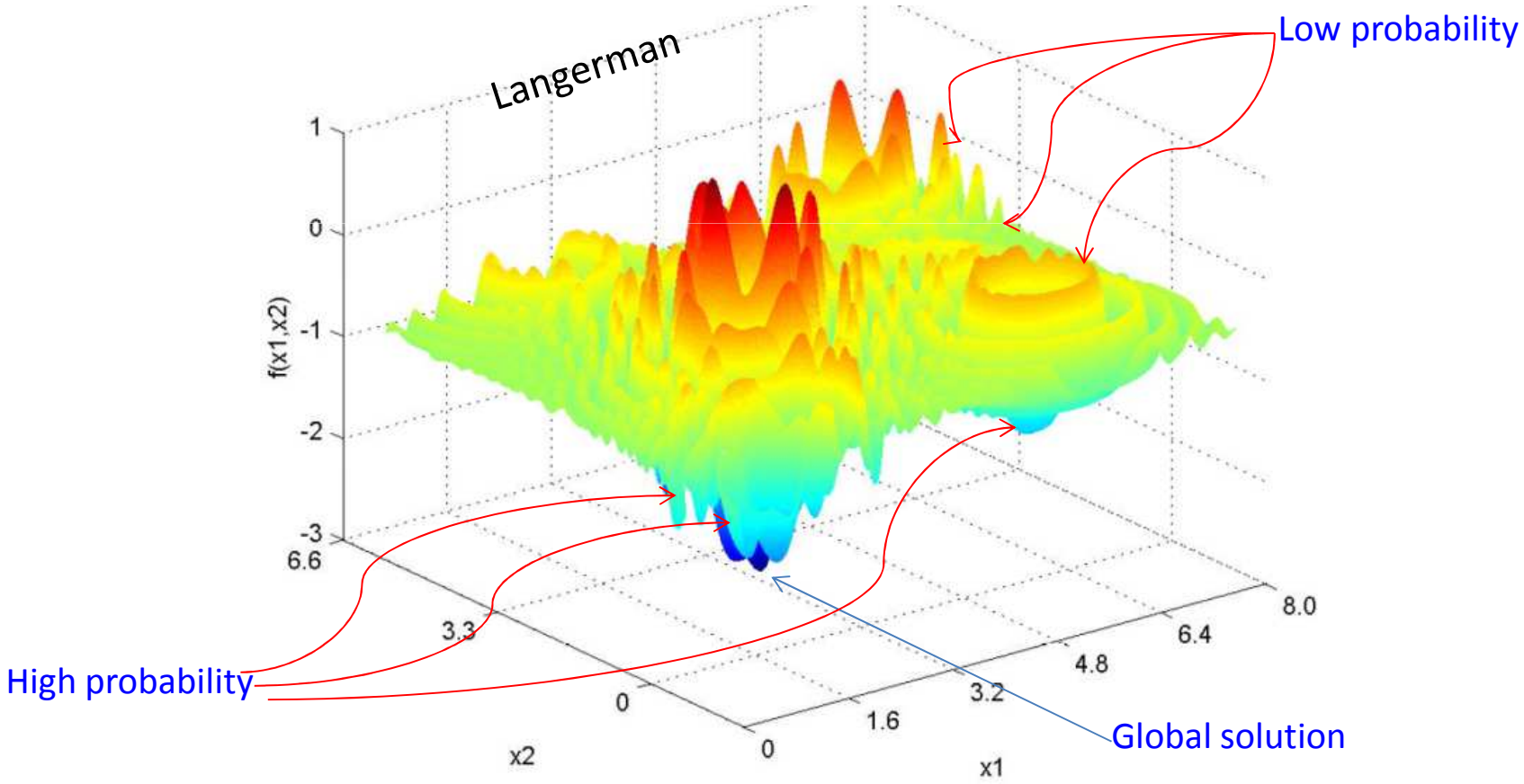
Population

- Selection operators usually take whole population into account i.e., reproductive probabilities are *relative* to *current* generation

- Diversity of a population refers to the number of different fitnesses / phenotypes / genotypes present (note not the same thing)
  - Entropy

Parent Selection Mechanism

- Assigns variable probabilities of individuals acting as parents depending on their fitnesses



Parent Selection Mechanism

- Usually probabilistic
  - high quality solutions more likely to become parents than low quality
  - but not guaranteed
  - even worst in current population usually has non-zero probability of becoming a parent
- This *stochastic* nature can aid escape from local optima

Variation Operators

- Role is to generate new candidate solutions
- Usually divided into two types according to their arity (number of inputs):
  - Arity 1 : mutation operators
  - Arity >1 : Recombination operators
  - Arity = 2 typically called crossover

- There has been much debate about relative importance of recombination and mutation
  - Nowadays most EAs use both
  - Choice of particular variation operators is representation dependant

Mutation

- Acts on one genotype and delivers another
- Element of randomness is essential and differentiates it from other unary heuristic operators
- Importance ascribed depends on representation and dialect:
  - Binary GAs – background operator responsible for preserving and introducing diversity
  - EP for FSM's/ continuous variables – only search operator
  - GP – hardly used
- May guarantee connectedness of search space and hence convergence proofs

Recombination

- Merges information from parents into offspring
- Choice of what information to merge is stochastic
- Most offspring may be worse, or the same as the parents
- Hope is that some are better by combining elements of genotypes that lead to good traits
- Principle has been used for millennia by breeders of plants and livestock

Survivor Selection

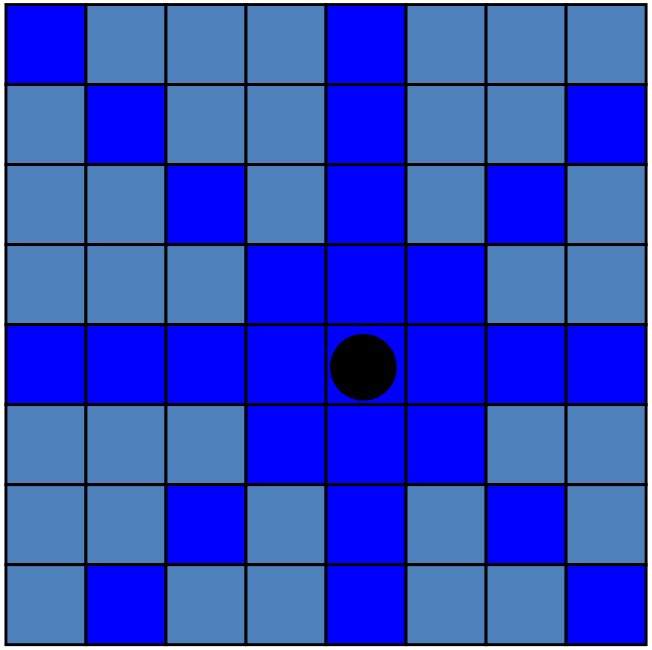
- a.k.a. ***Replacement***
- Most EAs use fixed population size so need a way of going from (parents + offspring) to next generation
- Often deterministic
  - Fitness based : e.g., rank parents+offspring and take best
  - Age based: make as many offspring as parents and delete all parents
- Sometimes do combination (elitism)

Initialization / Termination

- Initialization usually done at random,
  - Need to ensure even spread and mixture of possible allele values
  - Can include existing solutions, or use problem-specific heuristics, to "seed" the population

- Termination condition checked every generation
  - Reaching some (known/hoped for) fitness
  - Reaching some maximum allowed number of generations
  - Reaching some minimum level of diversity
  - Reaching some specified number of generations without fitness improvement

Example: the 8 queens problem

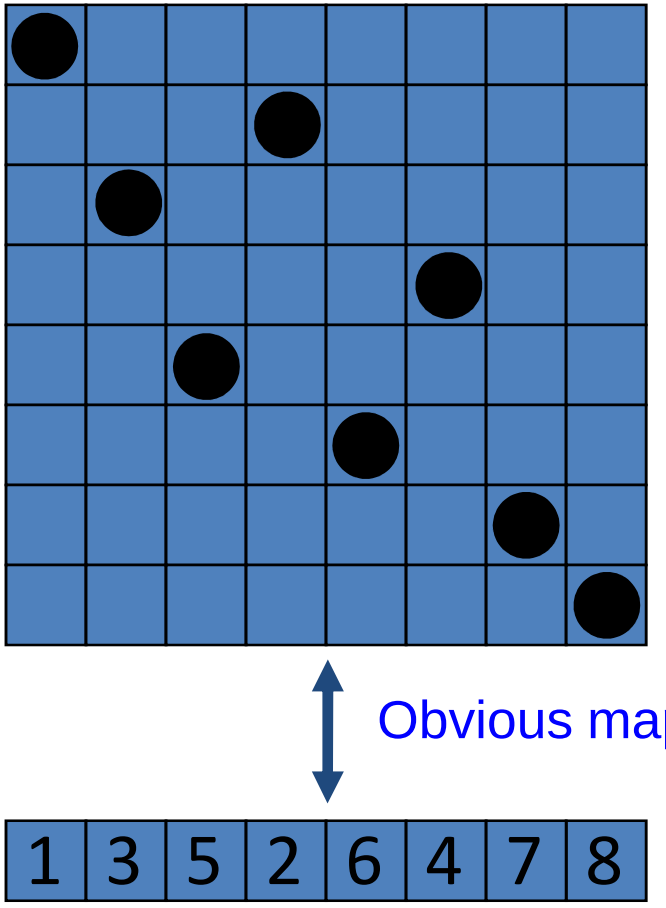


Place 8 queens on an 8x8 chessboard in such a way that they cannot check each other

The 8 queens problem: representation

Phenotype:
a board configuration

Genotype:
a permutation of
the numbers 1 - 8



8 Queens Problem: Fitness evaluation

- Penalty of one queen:
  the number of queens she can check.

- Penalty of a configuration:
  the sum of the penalties of all queens.

- Note: penalty is to be minimized

The 8 queens problem: Mutation

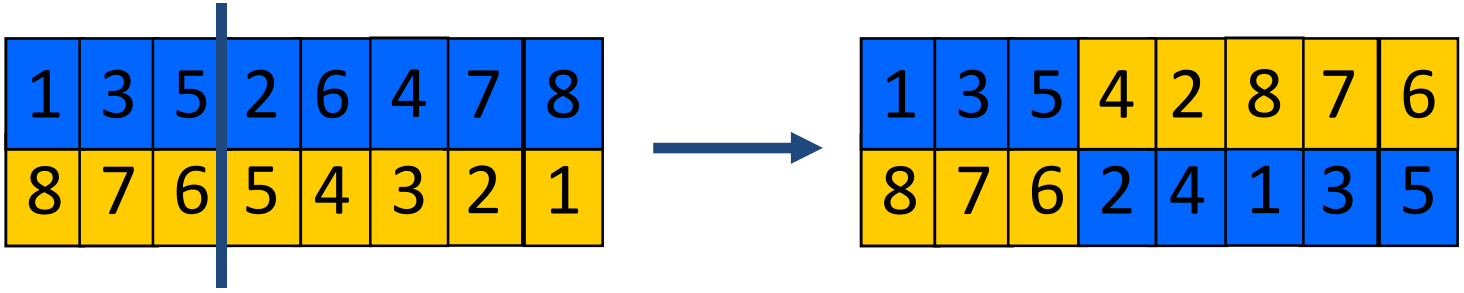
Small variation in one permutation, e.g.:

- swapping values of two randomly chosen positions,

The 8 queens problem: Recombination

Combining two permutations into two new permutations:

- choose random crossover point
- copy first parts into children
- create second part by inserting values from other parent:
  - in the order they appear there
  - beginning after crossover point
  - skipping values already in child



The 8 queens problem: Selection

- Parent selection:
  - Pick 5 parents and take best two to undergo crossover
- Survivor selection (replacement)
  - When inserting a new child into the population, choose an existing member to replace by:
    - sorting the whole population by decreasing fitness
    - enumerating this list from high to low
    - replacing the first with a fitness lower than the given child

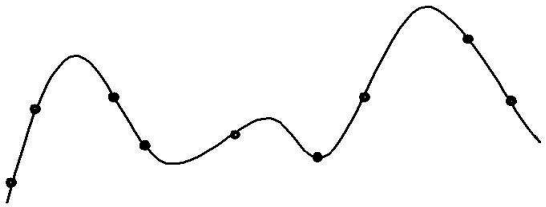
8 Queens Problem: summary

Representation	Permutations
Recombination	"Cut-and-crossfill" crossover
Recombination probability	100%
Mutation	Swap
Mutation probability	80%
Parent selection	Best 2 out of random 5
Survival selection	Replace worst
Population size	100
Number of Offspring	2
Initialisation	Random
Termination condition	Solution or 10,000 fitness evaluation

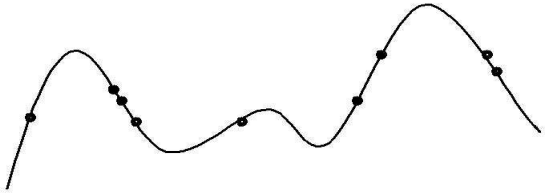
Note that it is ***only one possible***
set of choices of operators and parameters

Typical behaviour of an EA

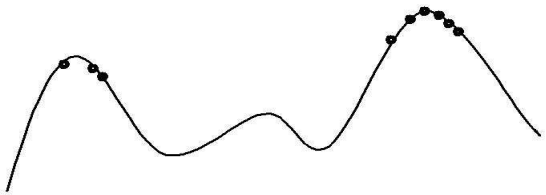
Phases in optimizing on a 1-dimensional fitness landscape



Early phase:
quasi-random population distribution

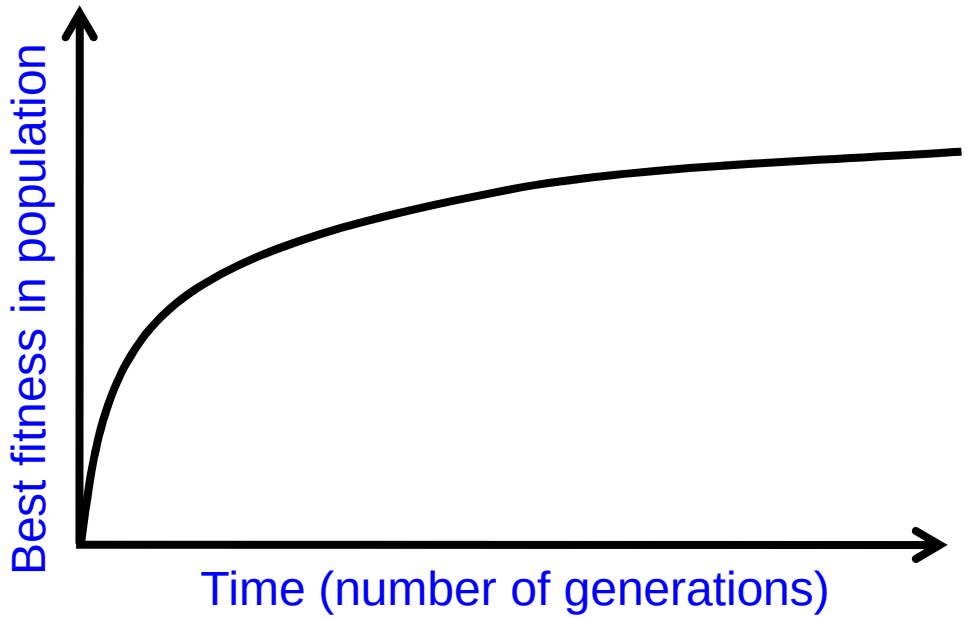


Mid-phase:
population arranged around/on hills



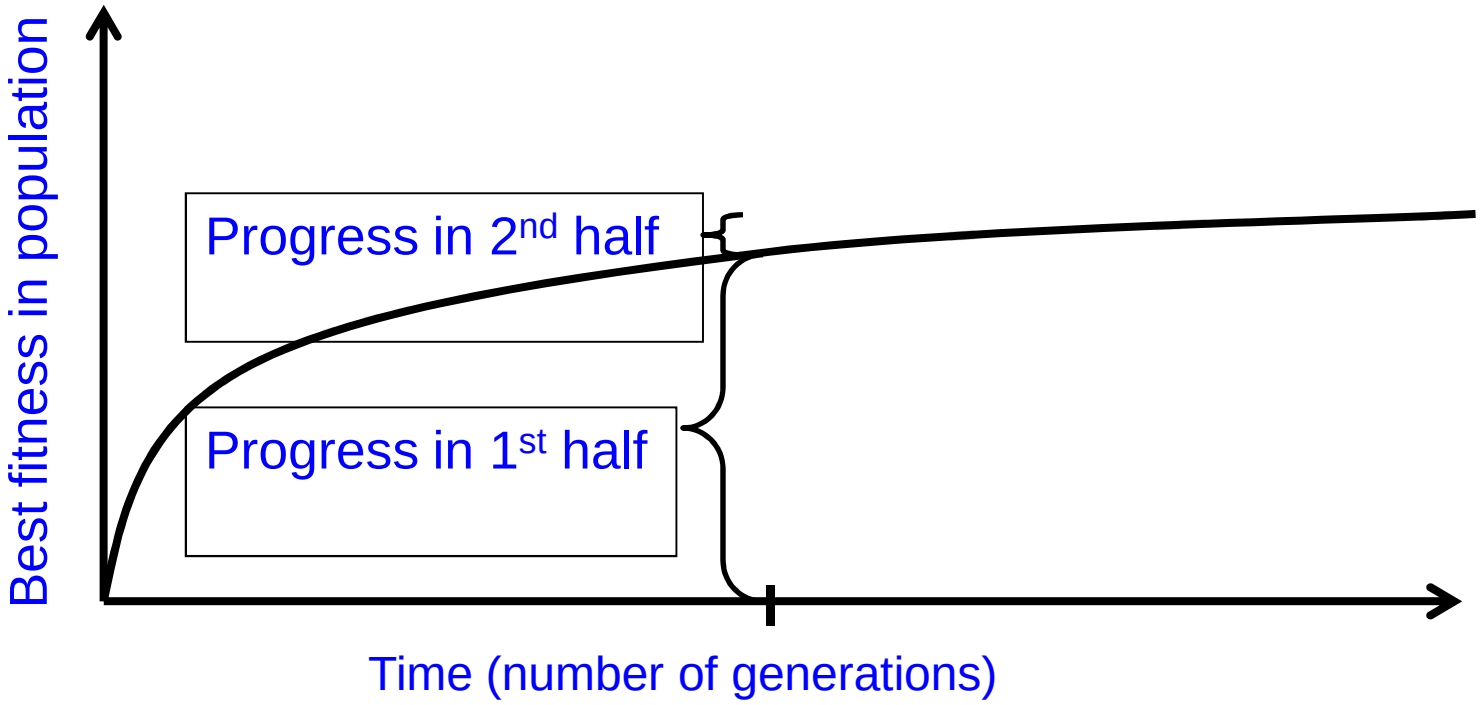
Late phase:
population concentrated on high hills

Typical run: progression of fitness



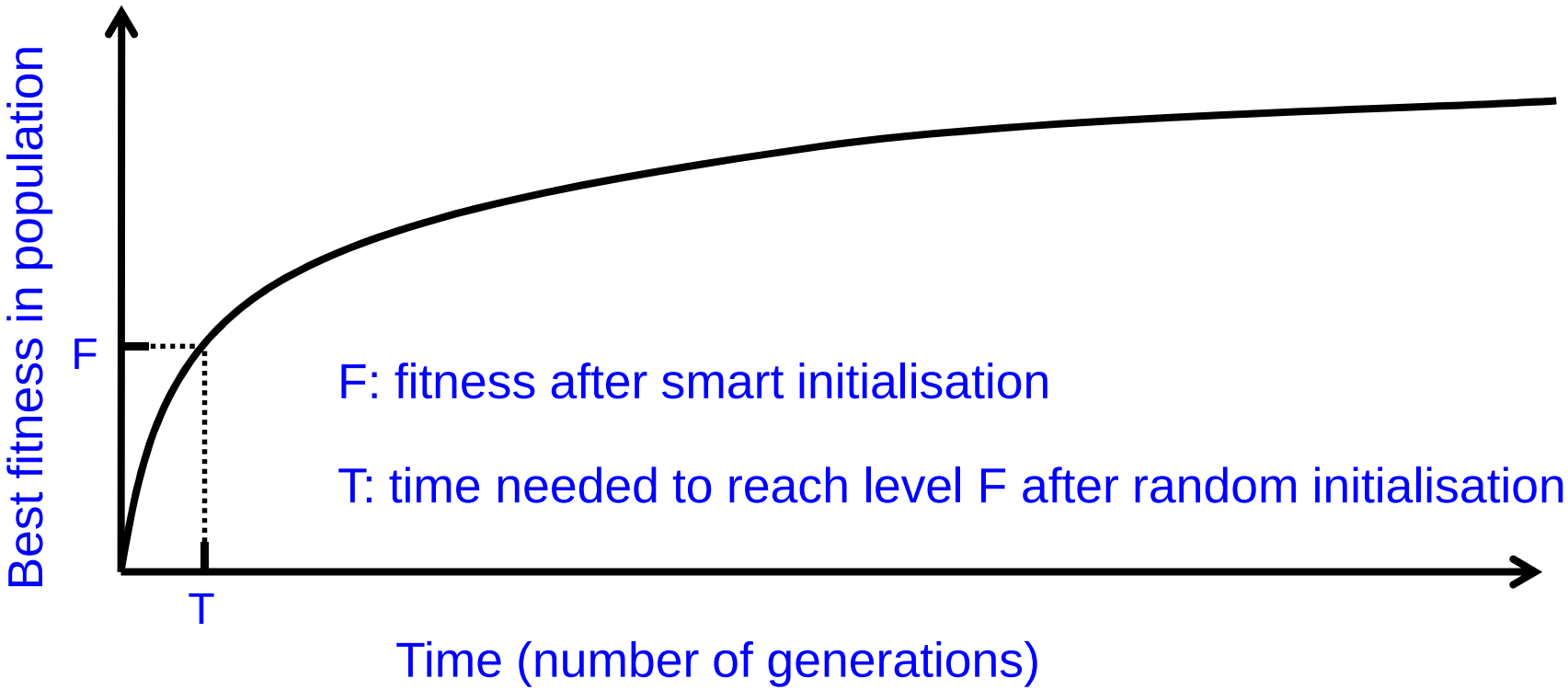
Typical run of an EA shows so-called “anytime behavior”

Are long runs beneficial?



- Answer:
  - it depends how much you want the last bit of progress
  - it may be better to do more shorter runs

Is it worth expending effort on smart initialization?



- Answer : it depends:
  - possibly, if good solutions/methods exist.
  - care is needed, see chapter on hybridisation

Evolutionary Algorithms in Context

- There are many views on the use of EAs as robust problem solving tools
- For most problems a problem-specific tool may:
  - perform better than a generic search algorithm on most instances,
  - have limited utility,
  - not do well on all instances
- Goal is to provide robust tools that provide:
  - evenly good performance
  - over a range of problems and instances

Machine Learning through Genetic Programming

GENETIC ALGORITHMS

- There is no single definitive genetic algorithm, rather algorithms are created from a suite of operators to suit particular applications

- Many of the issues relevant to GAs are also relevant to other evolutionary techniques, those aspects are going to be introduced here.

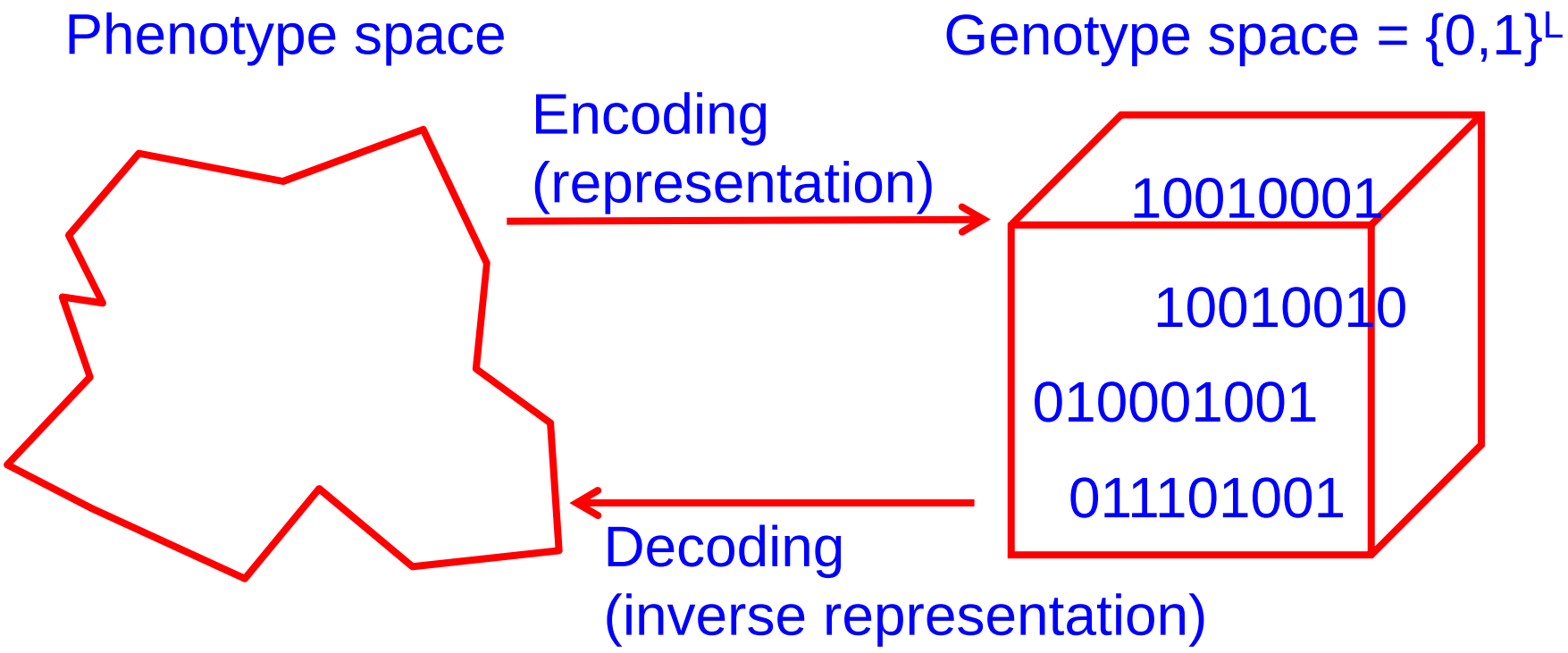
GA Quick Overview

- Developed: USA in the 1970's
- Early names: J. Holland, K. DeJong, D. Goldberg
- Typically applied to:
  - discrete optimization
- Attributed features:
  - not too fast
  - good heuristic for combinatorial problems
- Special Features:
  - Traditionally emphasizes combining information from good parents (crossover)
  - many variants, e.g., reproduction models, operators

Genetic algorithms

- Holland's original GA is now known as the simple genetic algorithm (SGA) or canonical GA
- Other GAs use different:
  - Representations
  - Mutations
  - Crossovers
  - Selection mechanisms

Representation



Representation

- First stage of building any evolutionary algorithm
- Defining the genotype and the mapping from genotype to phenotype
- Getting the representation right is one of the most difficult parts of designing a good EA
  - Good practice and good knowledge of the application domain
- *More common representation are explained but the might not be the best for your application*
  - Mixed representations is a more natural and suitable way of describing and manipulating a solution

Representation

- Binary
  - It has been mistakenly used almost independently of the problem to be solved
  - It consists simply of a string of binary digits
    - Length of the string
    - Phenotype interpretation
  - Defining genotype – phenotype mapping
    - All bit strings denote a valid solution to the problem
    - All possible solutions can be represented

Representation

- Binary
  - Problems with Boolean decision variables, genotype-phenotype mapping is natural, but frequently bit-string are used to encode non-binary information
    - Interpret an 80 bit-string as ten 8-bit integer or five 16-bit real numbers
      - Common mistake, better results can be obtained by using integer or real-valued representations

Representation

- Integer
  - Finding optimal values for a set of variables that all take integer values
    - Evolving a path on a square grid, representation {0,1,2,3} = {North, East, South, West}
- Real-valued or floating-point representation
  - Sensible way: having a real values string
  - Genes come from a continuous rather than discrete distribution
  - Precision is limited by implementation, thus reference is made as floating-point numbers

Representation

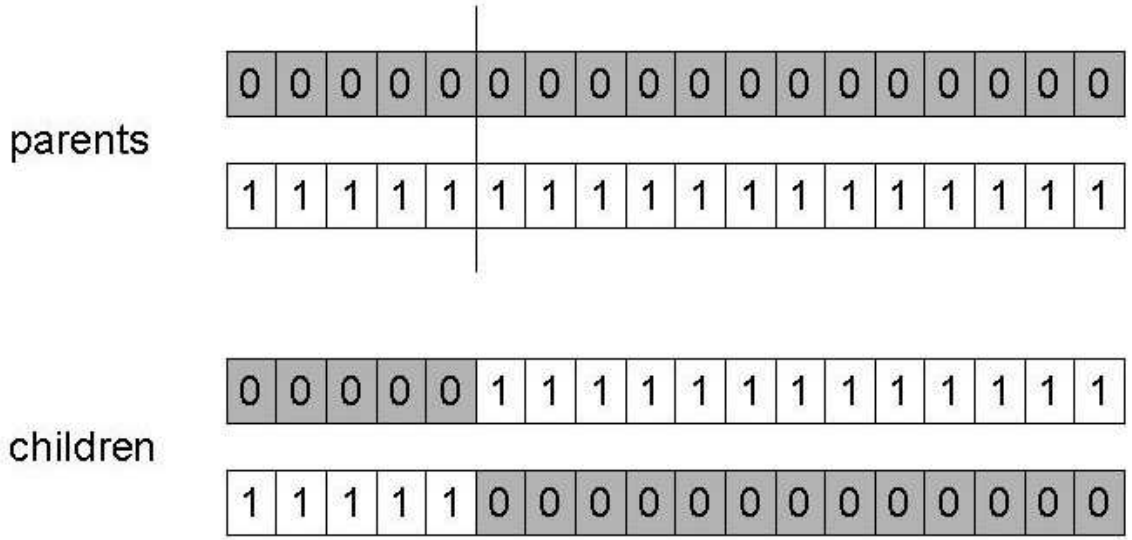
- Permutation
  - Many problems decide on the order in which a sequence of events should occur
  - Ordinary GA strings allows number to occur more than once, such sequences do not represent valid permutations
    - Variation operator must preserve this permutation property
  - Two classes of problems
    - Order
      - E.g. Job shop scheduling, task scheduling. Order matters
    - Adjacency
      - E.g. Travelling sales person, find a complete tour of n cities of minimal length. It can be considered that the starting point of the tour is not important: [1,2,3,4],[2,3,4,1],[3,4,1,2],[4,1,2,3] would be equivalent

SGA reproduction cycle

1. Select parents for the mating pool
   (size of mating pool = population size)
2. Shuffle the mating pool
3. For each consecutive pair apply crossover with probability p_c , otherwise copy parents
4. For each offspring apply mutation (bit-flip with probability p_m independently for each bit)
5. Replace the whole population with the resulting offspring

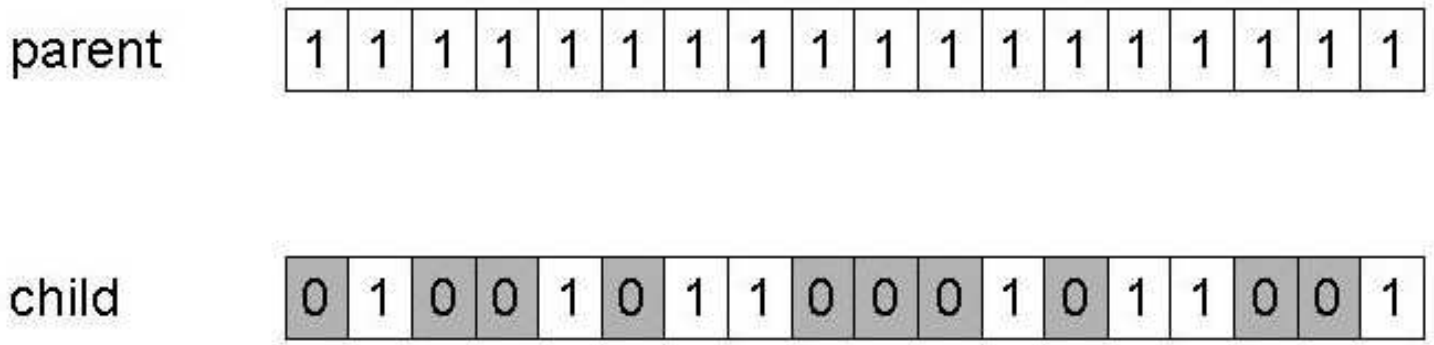
SGA operators: 1-point crossover

- Choose a random point on the two parents
- Split parents at this crossover point
- Create children by exchanging tails
- P_c typically in range (0.6, 0.9)



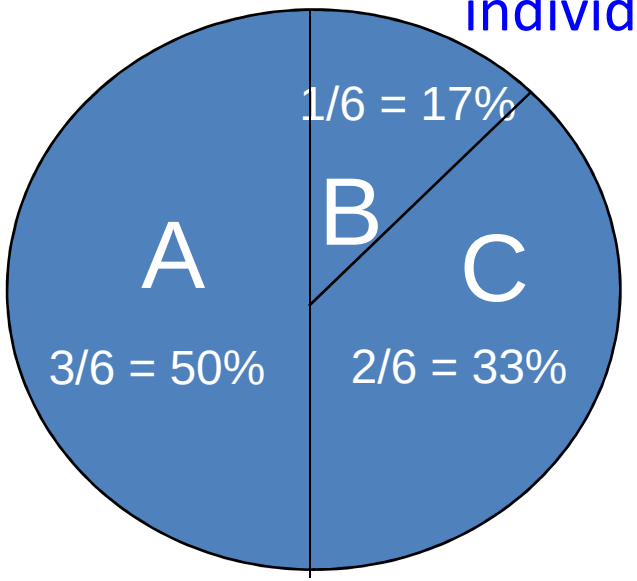
SGA operators: mutation

- Alter each gene independently with a probability p_m
- p_m is called the mutation rate
  - Typically between 1/pop_size and 1/ chromosome_length



SGA operators: Selection

- Main idea: better individuals get higher chance
  - Chances proportional to fitness
  - Implementation: roulette wheel technique
    - » Assign to each individual a part of the roulette wheel
    - » Spin the wheel n times to select n individuals



fitness(A) = 3

fitness(B) = 1

fitness(C) = 2

An example after Goldberg ‘89 (1)

- Simple problem: max x^2 over {0,1,…,31}
- GA approach:
  - Representation: binary code, e.g. 01101 $\leftrightarrow$ 13
  - Population size: 4
  - 1-point xover, bitwise mutation
  - Roulette wheel selection
  - Random initialisation
- We show one generational cycle done by hand

Representation	Bit-strings
Recombination	1-Point crossover
Mutation	Bit flip
Parent selection	Fitness proportional
Survival selection	Generational

Example x^2

String no.	Initial population	x Value	Fitness $f(x) = x^2$	$Prob_i$	Expected count	Actual count
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
Sum			1170	1.00	4.00	4
Average			293	0.25	1.00	1
Max			576	0.49	1.97	2

$$Prob_i = f_i / \sum f_j$$

Expected count = $f_i / \bar{f}$

Example x^2

- Crossover

String no.	Mating pool	Crossover point	Offspring after xover	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 | 1	4	0 1 1 0 0	12	144
2	1 1 0 0 | 0	4	1 1 0 0 1	25	625
2	1 1 | 0 0 0	2	1 1 0 1 1	27	729
4	1 0 | 0 1 1	2	1 0 0 0 0	16	256
Sum					1754
Average					439
Max					729

Example x^2

- Mutation

String no.	Offspring after xover	Offspring after mutation	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 0	1 1 1 0 0	28	784
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 1 0 0	20	400
Sum				2538
Average				634.5
Max				784

- A random number is below mutation rate for each bit position
  - Mutation rate e.g. 0.001

SGA technical summary tableau

Representation	Binary strings
Recombination	N-point or uniform
Mutation	Bitwise bit-flipping with fixed probability
Parent selection	Fitness-Proportionate
Survivor selection	All children replace parents
Speciality	Emphasis on crossover

The simple GA

- Has been subject of many (early) studies
  - still often used as benchmark for novel GAs
- Shows many shortcomings, e.g.
  - Representation is too restrictive
  - Mutation & crossovers only applicable for bit-string & integer representations
  - Selection mechanism sensitive for converging populations with close fitness values
  - Generational population model (step 5 in SGA reproductive cycle) can be improved with explicit survivor selection

Brief History: the ancestors

- 1948, Turing:
  proposes "genetical or evolutionary search"
- 1962, Bremermann
  optimization through evolution and recombination
- 1964, Rechenberg
  introduces evolution strategies
- 1965, L. Fogel, Owens and Walsh
  introduce evolutionary programming
- 1975, Holland
  introduces genetic algorithms
- 1992, Koza
  introduces genetic programming

Evolutionary computing

Machine Learning through Genetic Programming

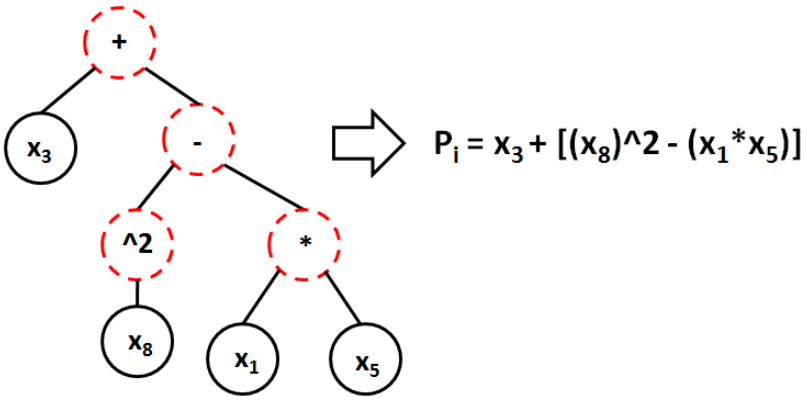
GENETIC PROGRAMMING

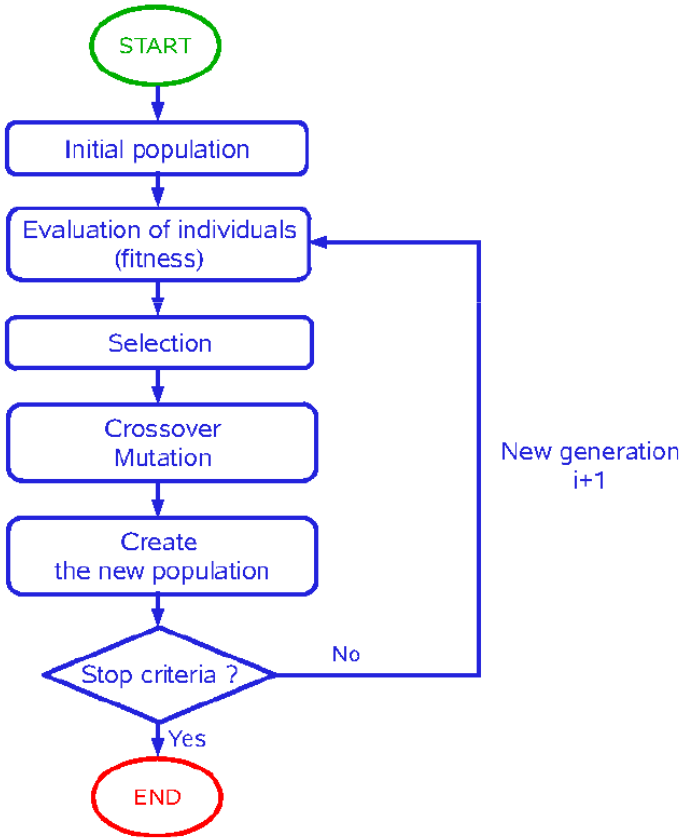
Brief History 1: the ancestors

- 1948, Turing:
  proposes "genetical or evolutionary search"
- 1962, Bremermann
  optimization through evolution and recombination
- 1964, Rechenberg
  introduces evolution strategies
- 1965, L. Fogel, Owens and Walsh
  introduce evolutionary programming
- 1975, Holland
  introduces genetic algorithms
- 1992, Koza
  introduces genetic programming

Main differences with other EA's

- GP: standard EA with tree-based representation





Main differences with other EA's

- Most other EAs target optimization/simulation problems, whereas in GP the common target is a machine learning task

GP quick overview

- Developed: USA in the 1990's
- Early names: J. Koza
- Typically applied to:
  - machine learning tasks (prediction, classification…)
- Attributed features:
  - competes with neural nets and alike
  - needs huge populations (thousands)
  - slow
- Special:
  - non-linear chromosomes: trees, graphs
  - mutation possible but not necessary (disputed!)

A.E. Eiben and J.E. Smith, **Introduction to Evolutionary Computing,** Springer, Natural Computing Series 1st edition, 2003, ISBN: 3-540-40184-9, Corr. 2nd printing, 2007, ISBN: 978-3-540-40184-1

GP technical summary tableau

Representation	Tree structures
Recombination	Exchange of subtrees
Mutation	Random change in trees
Parent selection	Fitness proportional
Survivor selection	Generational replacement

Introductory example: credit scoring

- Bank wants to distinguish good from bad loan applicants
- Model needed that matches historical data

ID	No of children	Salary	Marital status	OK?
ID-1	2	45000	Married	0
ID-2	0	30000	Single	1
ID-3	1	40000	Divorced	1
…	…	…	…	…

Introductory example: credit scoring

- A possible model:

  IF (NOC = 2) AND (S > 80000) THEN good ELSE bad

- In general:

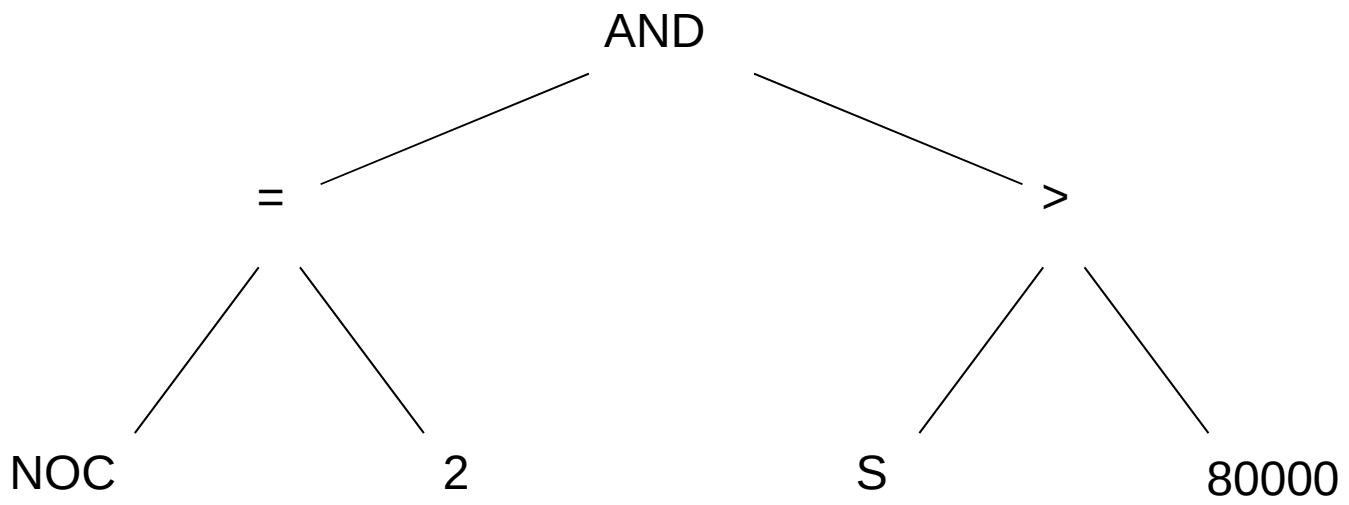
  IF formula THEN good ELSE bad

- Only unknown is the right formula, hence
- Our search space (phenotypes) is the set of formulas
- Natural fitness of a formula: percentage of well classified cases of the model it stands for
- Natural representation of formulas (genotypes) is: parse trees

Introductory example: credit scoring

IF (NOC = 2) AND (S > 80000) THEN good ELSE bad

can be represented by the following tree



GP: is used to search for the best tree (classifier)

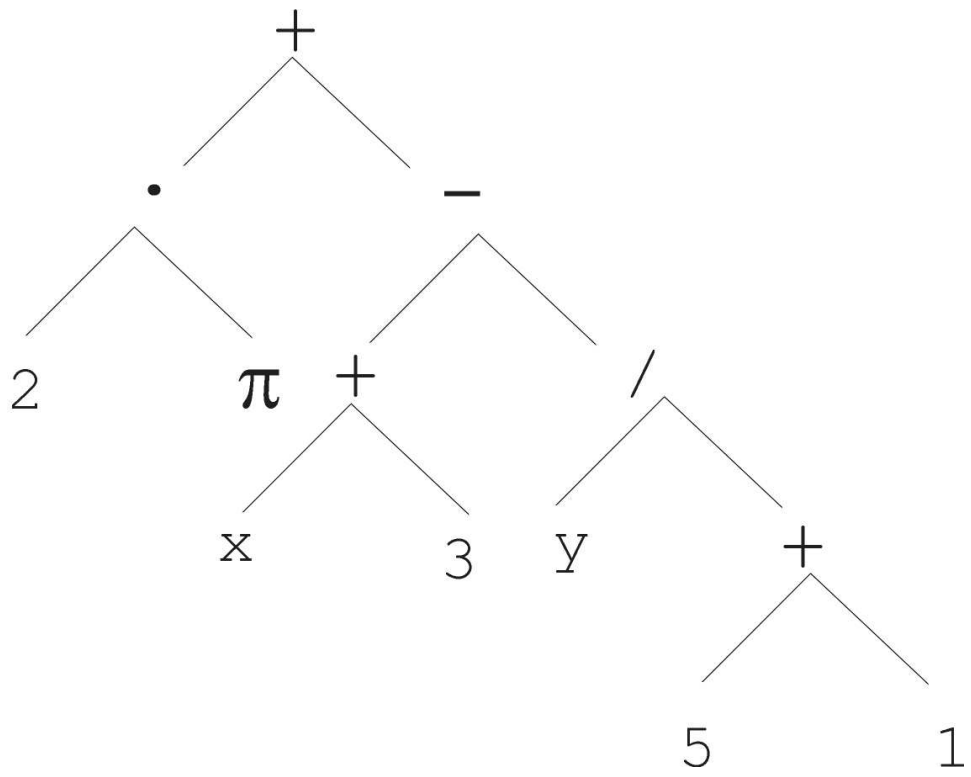
Tree based representation

- Trees are a universal form, e.g. consider
- Arithmetic formula $2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$
- Logical formula $(x \wedge \text{true}) \rightarrow ((x \vee y) \vee (z \leftrightarrow (x \wedge y)))$
- Program

```
i =1;
while (i < 20)
{
        i = i +1
}
```

GP: is used to search for the best tree (classifier)

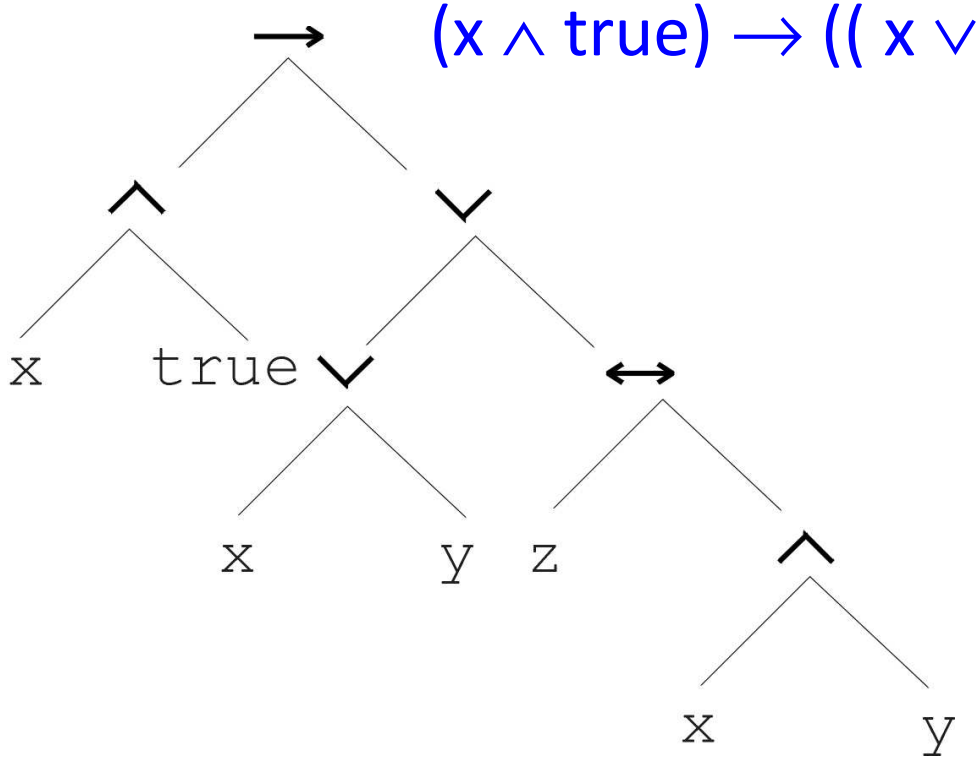
Tree based representation



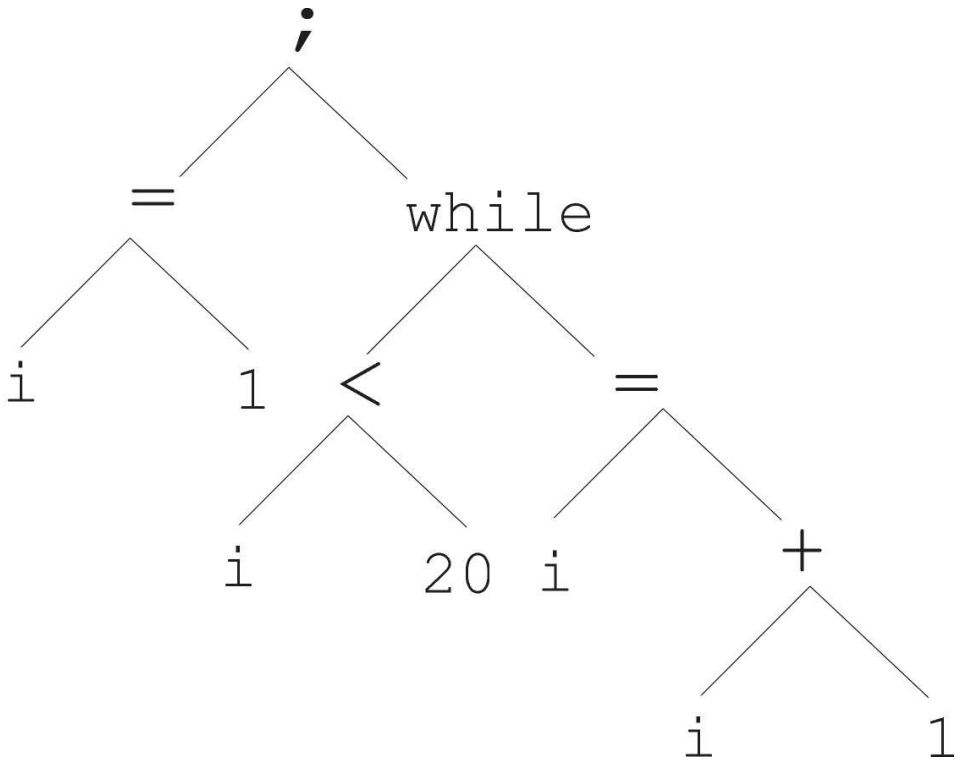
$$2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$$

Tree based representation

$(x \wedge \text{true}) \rightarrow ((\ x \vee y\ ) \vee (z \leftrightarrow (x \wedge y)))$



Tree based representation




```
i =1;
while (i < 20)
{
        i = i +1
}
```

Tree based representation

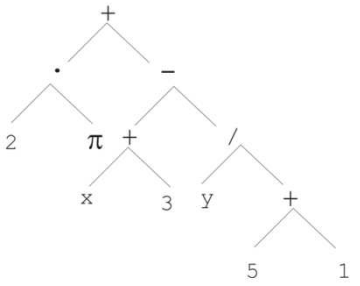
- In GA, ES, EP chromosomes are linear structures (bit strings, integer string, real-valued vectors, permutations)
- Tree shaped chromosomes are non-linear structures
- In GA, ES, EP the size of the chromosomes is fixed
- Trees in GP may vary in depth and width

Tree based representation

- Symbolic expressions can be defined by
  - Terminal set T
  - Function set F (with the arities of function symbols)

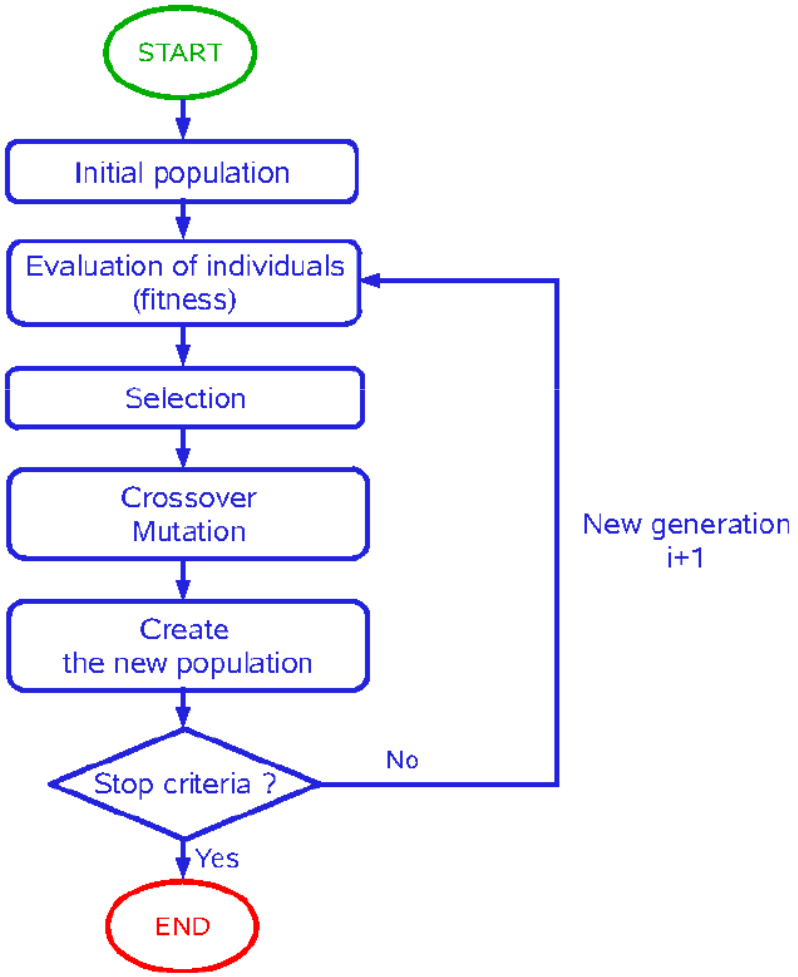
- Adopting the following general recursive definition:
  1. Every $t \in T$ is a correct expression
  2. $f(e_1, \ldots, e_n)$ is a correct expression if $f \in F$, arity(f)=n and $e_1, \ldots, e_n$ are correct expressions
  3. There are no other forms of correct expressions

- In general, expressions in GP are not typed (closure property: any $f \in F$ can take any $g \in F$ as argument)



$$2 \cdot \pi + \left((x+3) - \frac{y}{5+1} \right)$$

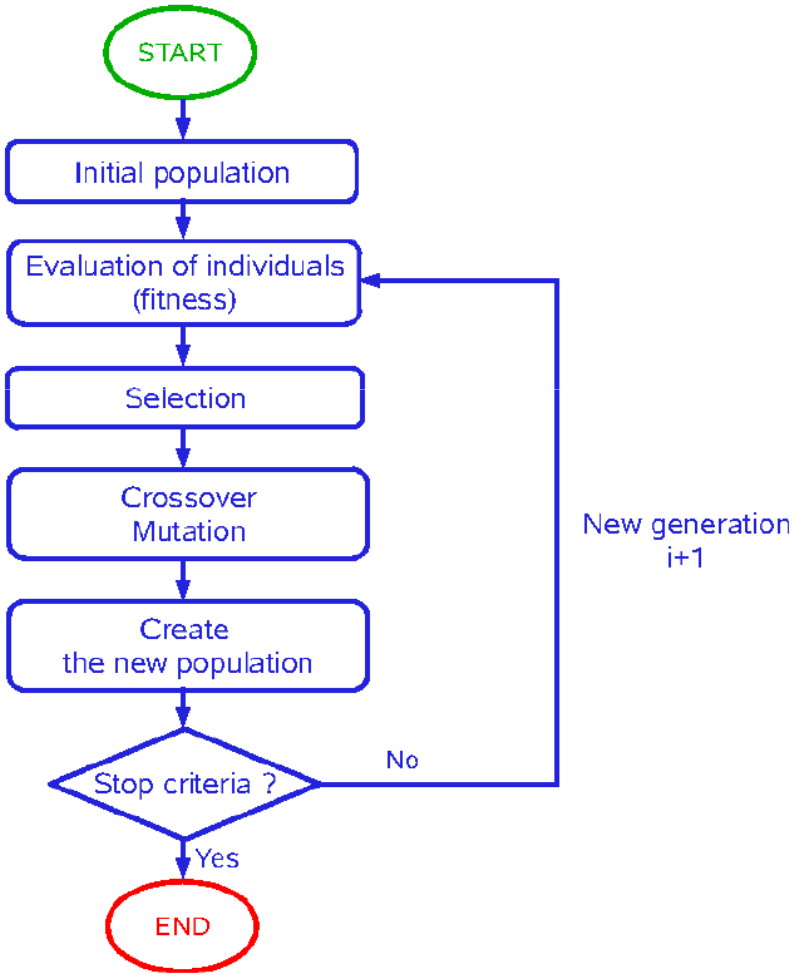
Generic EA


START
Initial population
Evaluation of individuals (fitness)
Selection
Crossover
Mutation
Create the new population
Stop criteria ?
No
Yes
END
New generation i+1

Initialization

- Maximum initial depth of trees D_{max} is set
- Full method (each branch has depth = D_{max}):
  - nodes at depth $d < D_{max}$ randomly chosen from function set F
  - nodes at depth $d = D_{max}$ randomly chosen from terminal set T
- Grow method (each branch has depth $\leq D_{max}$):
  - nodes at depth $d < D_{max}$ randomly chosen from $F \cup T$
  - nodes at depth $d = D_{max}$ randomly chosen from T
- Common GP initialisation: ramped half-and-half, where grow & full method each deliver half of initial population

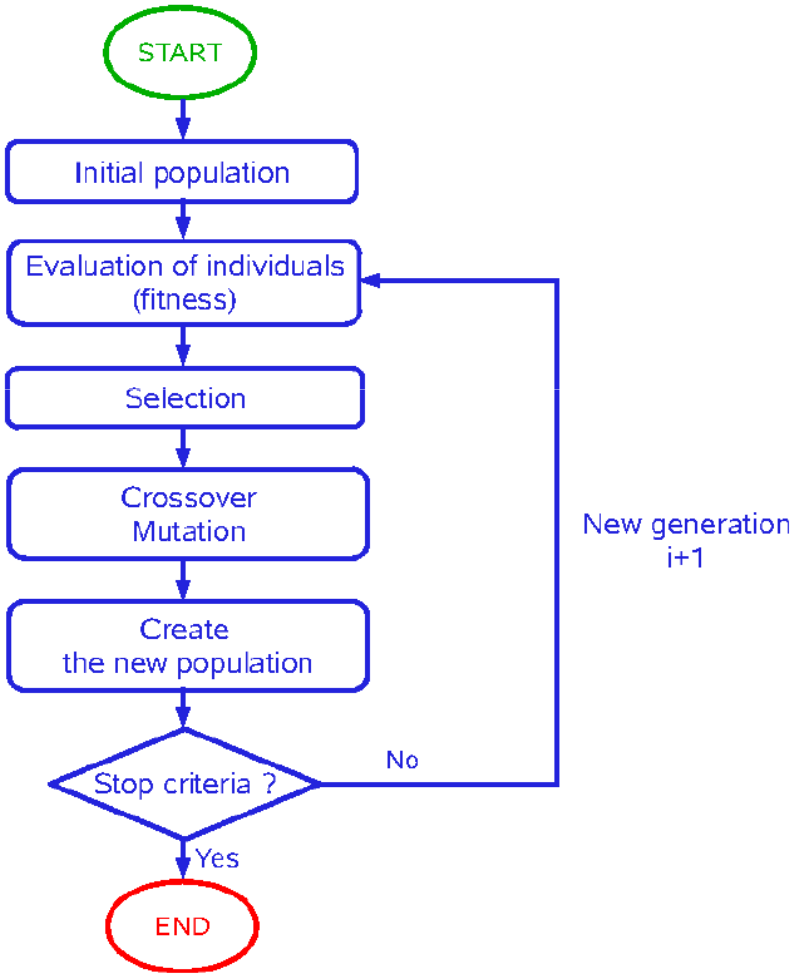
Generic EA


START
Initial population
Evaluation of individuals (fitness)
Selection
Crossover
Mutation
Create the new population
Stop criteria ?
No
Yes
END
New generation i+1

Selection

- Parent selection typically fitness proportionate
- Over-selection in very large populations
  - rank population by fitness and divide it into two groups:
  - group 1: best x% of population, group 2 other (100-x)%
  - 80% of selection operations chooses from group 1, 20% from group 2
  - for pop. size = 1000, 2000, 4000, 8000 x = 32%, 16%, 8%, 4%
  - motivation: to increase efficiency, %'s come from rule of thumb
- Survivor selection:
  - Typical: generational scheme
  - Recently steady-state is becoming popular for its elitism

Generic EA

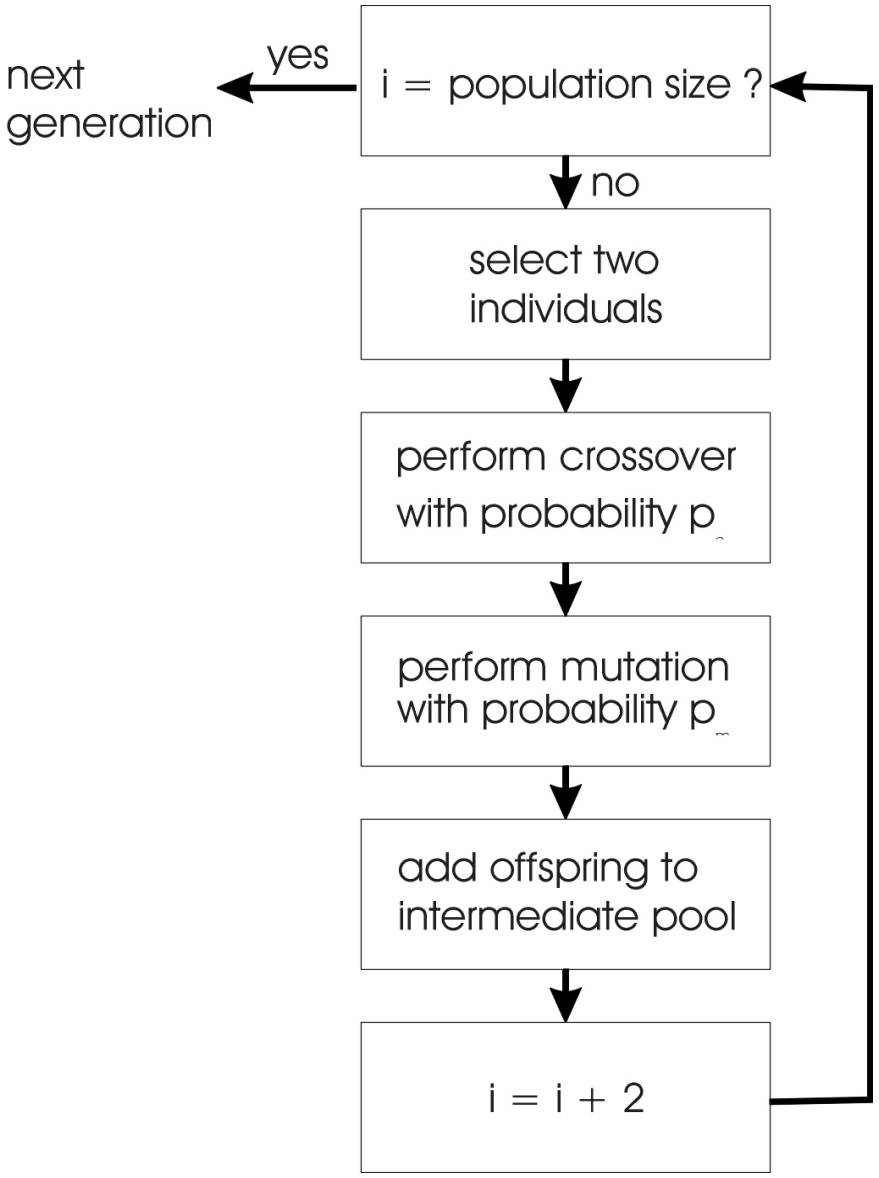

START
Initial population
Evaluation of individuals (fitness)
Selection
Crossover
Mutation
Create the new population
Stop criteria ?
No
Yes
END
New generation i+1

Offspring creation scheme

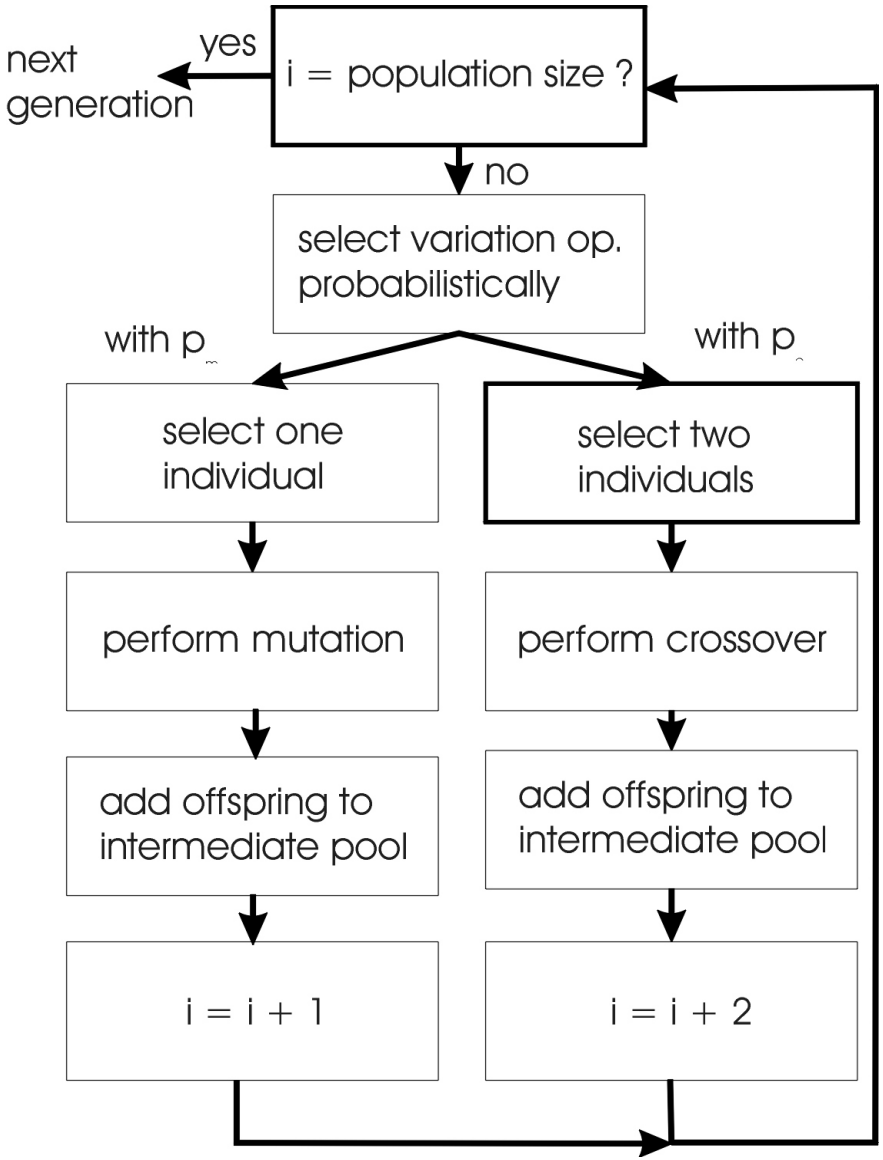
Compare

- GA scheme using crossover AND mutation sequentially (be it probabilistically)

- GP scheme using crossover OR mutation (chosen probabilistically)



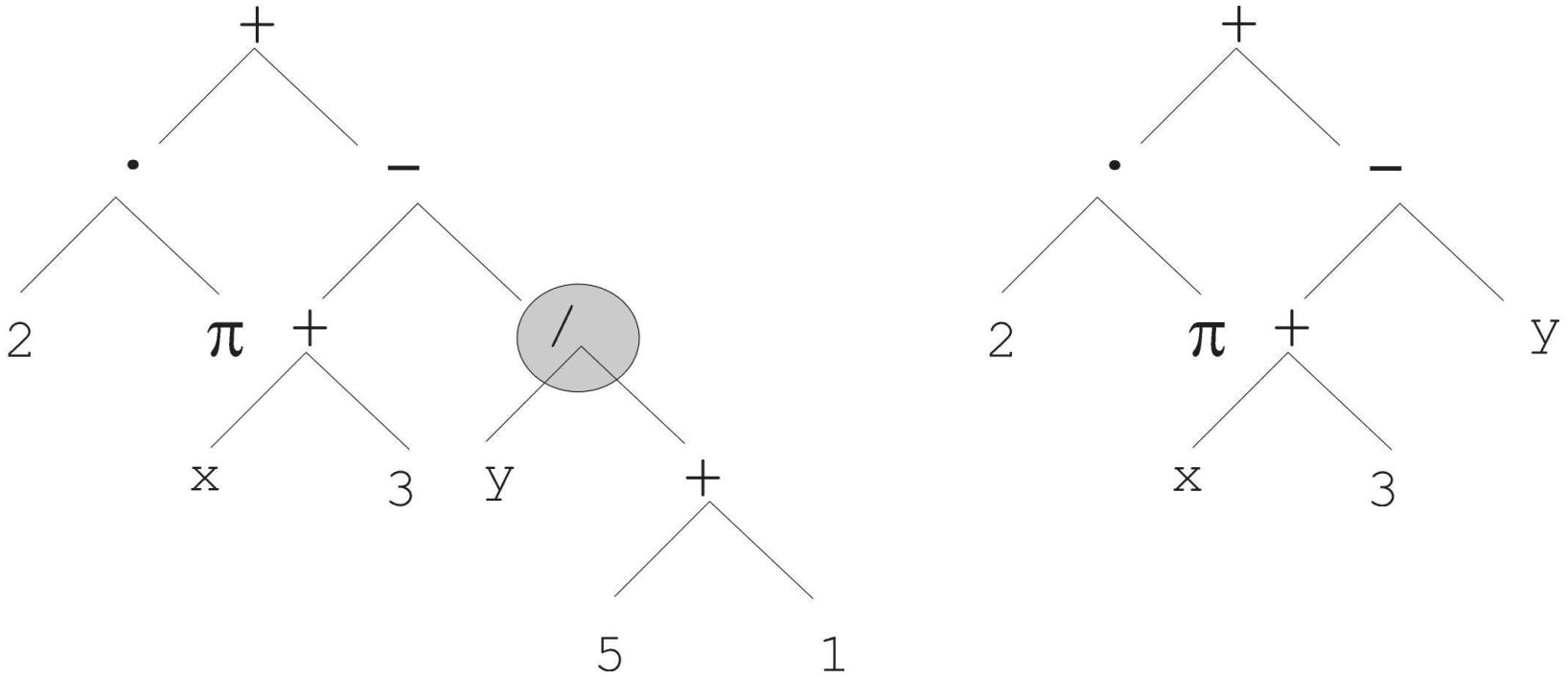
GA flowchart



GP flowchart

Mutation

- Most common mutation: replace randomly chosen subtree by randomly generated tree

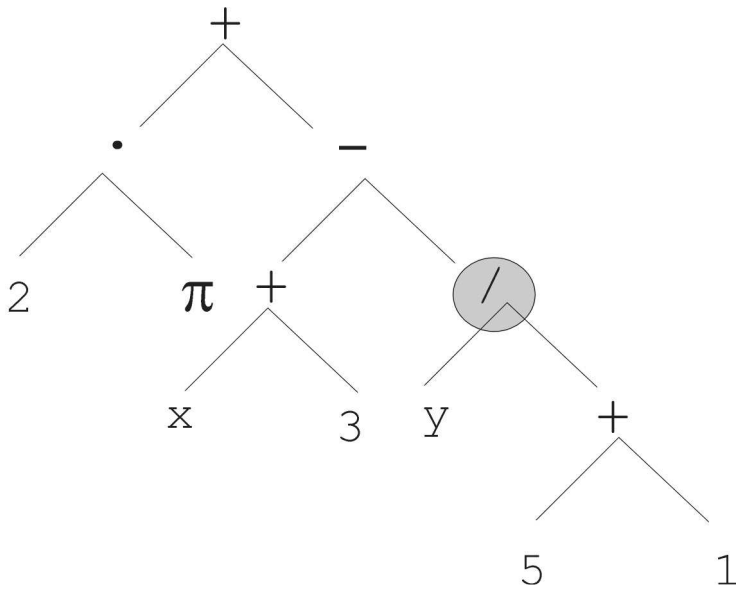


Mutation cont'd

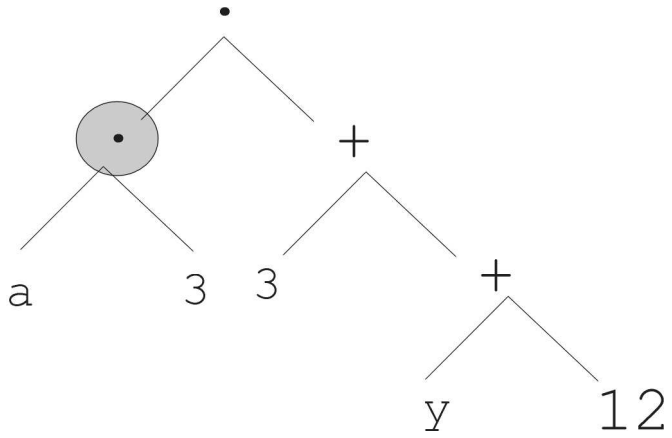
- Mutation has two parameters:
  - Probability p_m to choose mutation vs. recombination
  - Probability to chose an internal point as the root of the subtree to be replaced
- Remarkably p_m is advised to be 0 (Koza'92) or very small, like 0.05 (Banzhaf et al. '98)
- The size of the child can exceed the size of the parent

Recombination

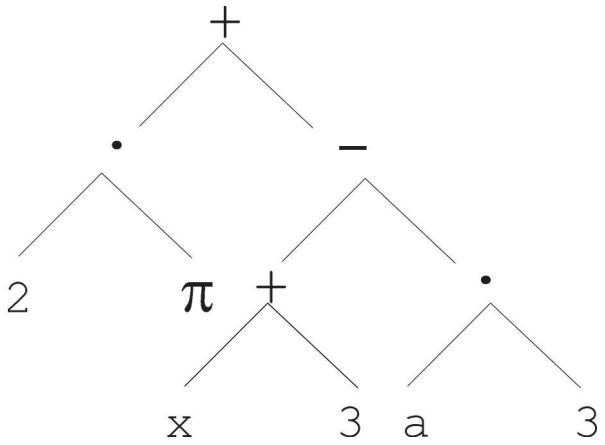
- Most common recombination: exchange two randomly chosen subtrees among the parents
- Recombination has two parameters:
  - Probability p_c to choose recombination vs. mutation
  - Probability to chose an internal point within each parent as crossover point
- The size of offspring can exceed that of the parents



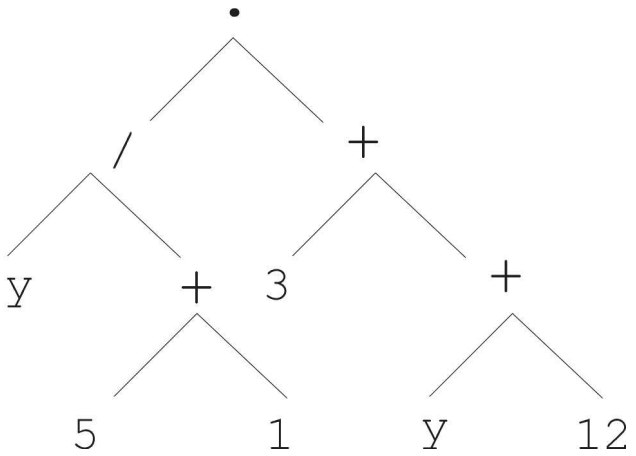
Parent 1



Parent 2



Child 1



Child 2

Discussion

Is GP:

The art of evolving computer programs ?

Means to automated programming of computers?

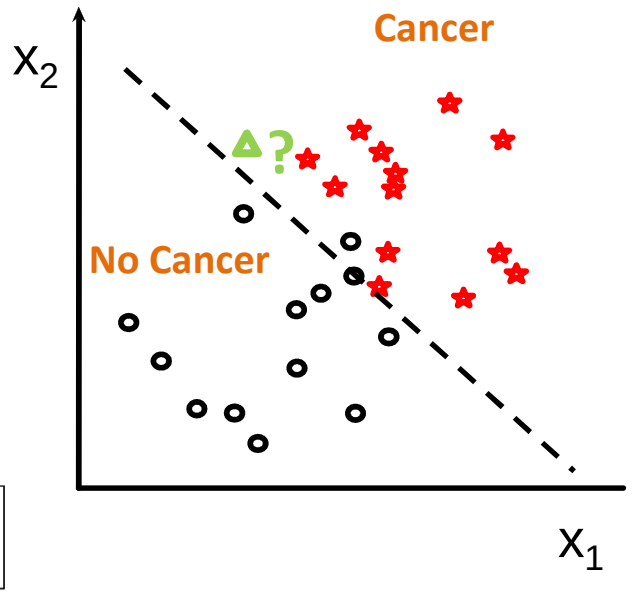
GA with another representation?

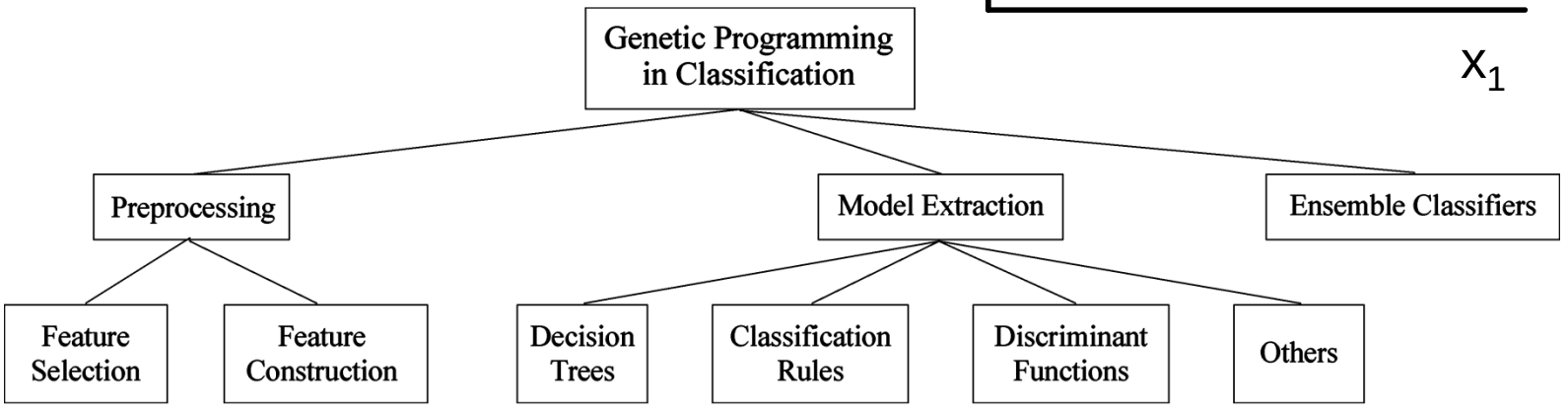
Machine Learning through Genetic Programming

GENETIC PROGRAMMING FOR CLASSIFICATION

GP for classification

- **Problem:** to learn a classification model from data.
  - Appropriate for other ML tasks.

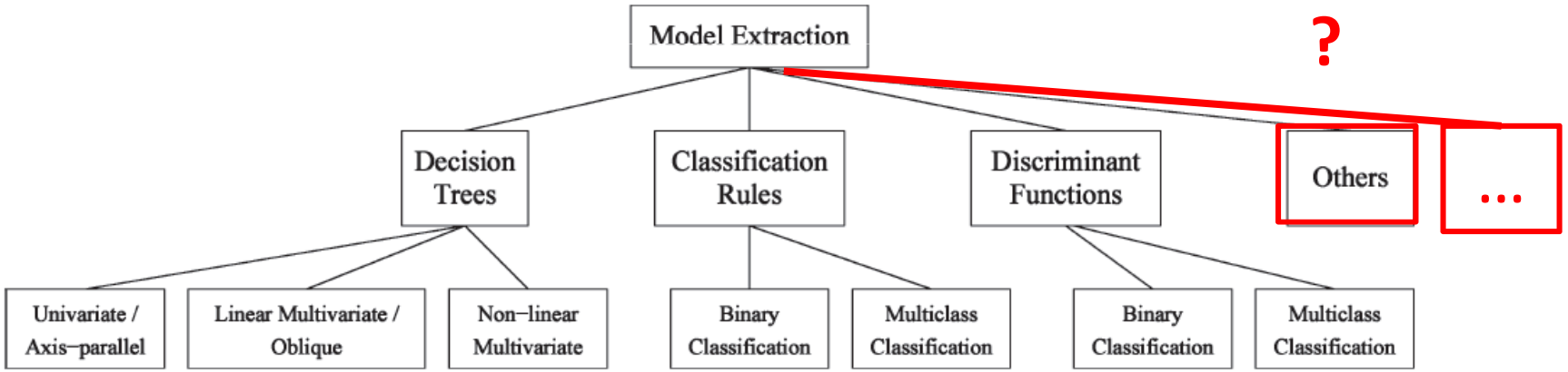




P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

GP for classification

- **Problem:** to learn a classification model from data



P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

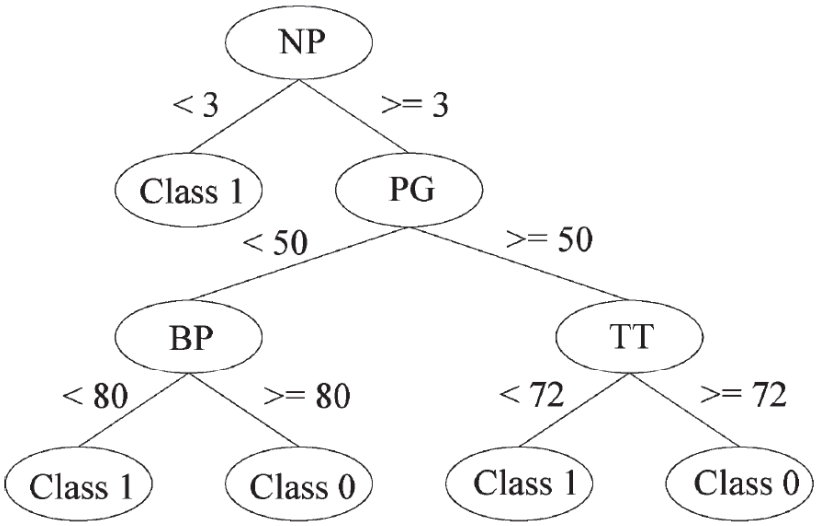
GP for classification

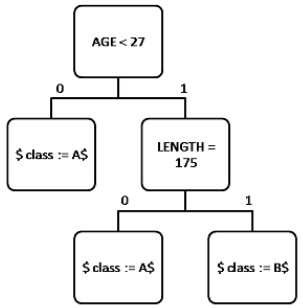
- **General idea:** Design a GP where each individual represents a classifier (or a part of it). Defining a fitness function that evaluates the quality of the classifier.

P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

GP for classification

- **Decision trees:** A natural way to induce classifiers via GP
  - Typically one individual encodes a classifier (DT)
  - Internal nodes test the value of an expression on the attributes
  - Edges of IN are labeled with distinct outcomes
  - Each leaf node has a class label associated



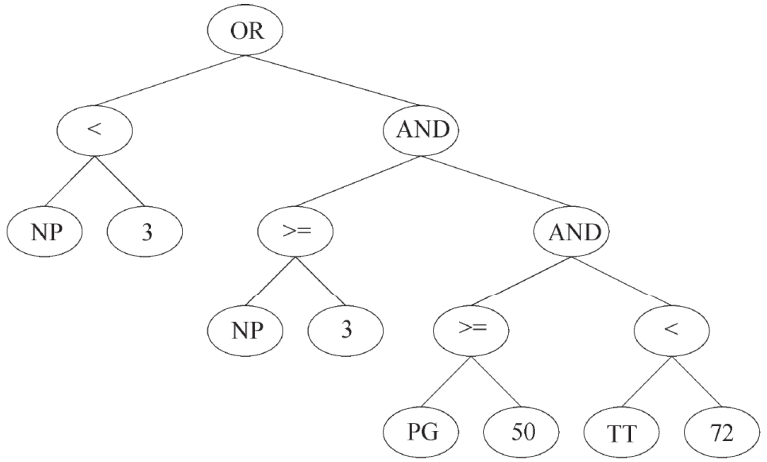


Type of decision trees (approached with GP): axis parallell, oblique, nonlinear

P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

GP for classification

- **Rule-based systems:** Simple and easily interpretable way to represent knowledge
  - Two components: *antecedent* and *consequent*
  - Two main paradigms:
    - Individual=rule
    - Individual=set of rules



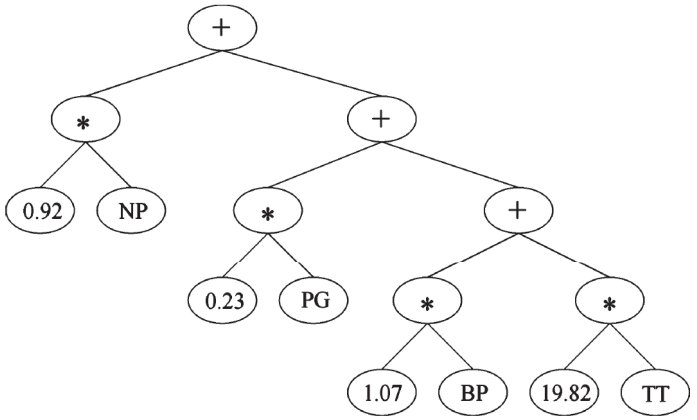
IF ((NP < 3)
OR ((NP ≥ 3) AND (PG ≥ 50) AND (TT < 72)))
THEN Class 1.

Binary vs. Multiclass

P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

GP for classification

- **Discriminant functions:** A discriminative mathematical expression
  - The evaluation of the mathematical expression indicates the class (threshold, usually *0*)



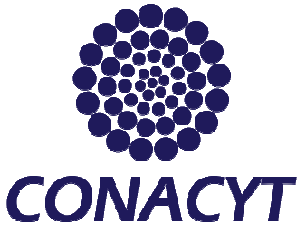
$$0.92^{*}\mathrm{NP} + 0.23^{*}\mathrm{PG} + 1.07^{*}\mathrm{BP} + 19.82^{*}\mathrm{TT}$$

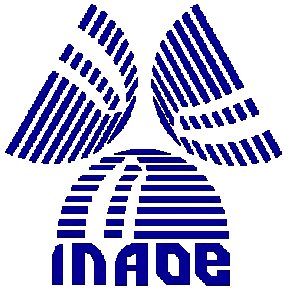
Binary vs. Multiclass

P. G. Espejo, S. Ventura, and F. Herrera. **A Survey on the Application of Genetic Programming to Classification.** IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, VOL. 40(2), 2010

Machine Learning through Genetic Programming

APPLICATIONS





Machine Learning through Genetic Programming: Applications

Hugo Jair Escalante, Karlo Mendoza, Mario Graff, Alicia Morales, Eduardo Morales, Manuel Montes, Mauricio García, Niusvel Acosta, Andres Gago

Laboratorio de
Tecnologías del Lenguaje
Ciencias Computacionales, INAOE

Automatic generation of classification prototypes

Ensemble generation

Term-weighting scheme learning

SOME APPLICATIONS OF GP ON MACHINE LEARNING PROBLEMS

Automatic generation of classification prototypes

Ensemble generation

Term-weighting scheme learning

SOME APPLICATIONS OF GP ON MACHINE LEARNING PROBLEMS

H. J. Escalante, K. Mendoza, M. Graff, A. Morales-Reyes**. Genetic Programming of Prototypes for Pattern Classification.** IbPRIA 2013: 6th Iberian Conference on Pattern Recognition and Image Analysis, Madeira, Portugal. June 5-7, 2013

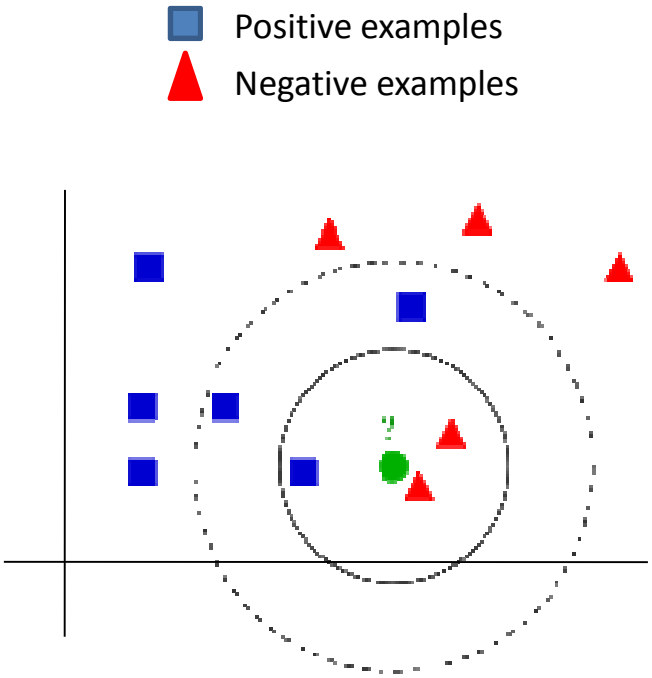
H. J. Escalante, M. Graff, A. Morales-Reyes. **PGGP: Prototype Generation via Genetic Programming.** *Applied Soft Computing*, Minor revisions, 2014

Genetic programming of classification prototypes

- **Problem:** Given a labeled data set of a classification task to select a (**small**) subset of instances such that the classification performance of a particular classifier (KNN) is not degraded significantly

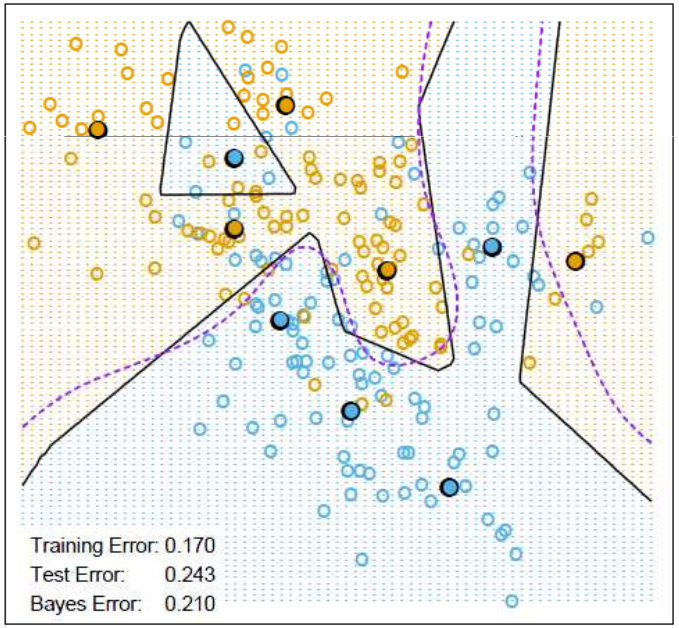
- **Naïve approach:** take as prototype of class C_i to the average of instances that belong to class C_i

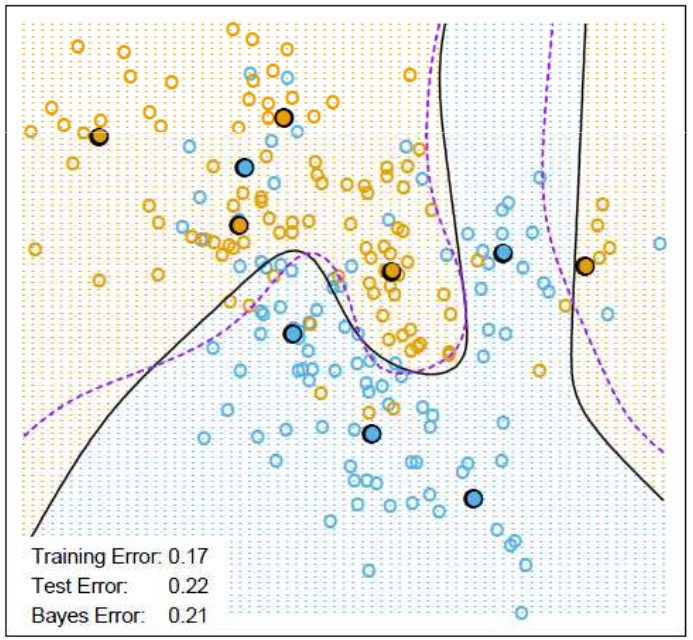
$$\mathbf{p}_i = \frac{1}{N_i} \sum_{\mathbf{x}_j : y_j == C_i}^{N_i} \mathbf{x}_j$$



Genetic programming of classification prototypes

- Prototype generation:





T. Hastie, R. Tibshirani, J. Friedman. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction.** 2nd Edition, Springer, 2009

Genetic programming of classification prototypes

- **Proposed solution:** using GP to learn how to combine instances of the same class to generate prototypes

Naïve approach

$$\mathbf{p}_i = \frac{1}{N_i} \sum_{\mathbf{x}_j : y_j == C_i}^{N_i} \mathbf{x}_j$$

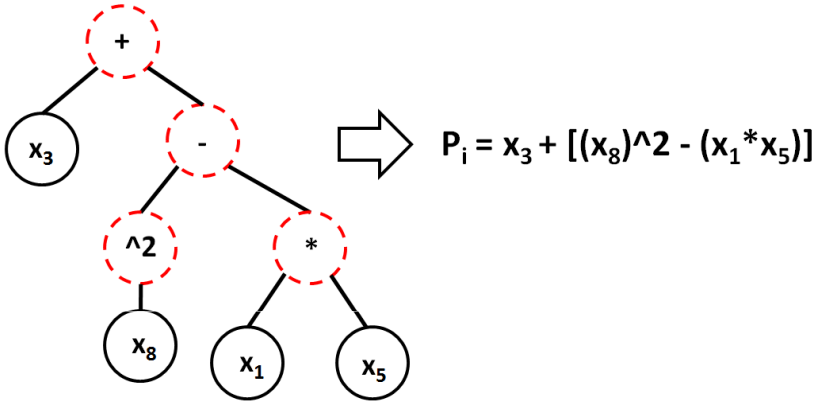
Genetic program

$$\mathbf{p}_i = f(\mathbf{x}_j), \quad ? \quad \mathbf{x}_j : y_j == C_i$$

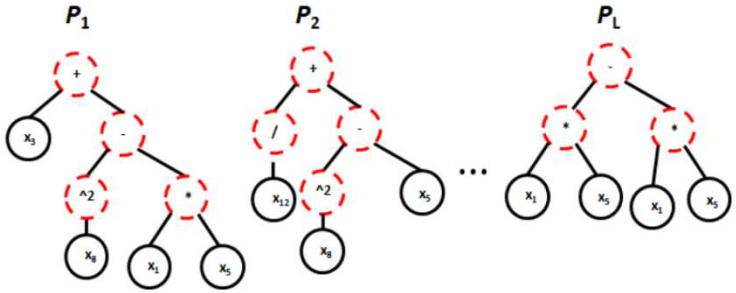
H. J. Escalante, K. Mendoza, M. Graff, A. Morales-Reyes. **Genetic Programming of Prototypes for Pattern Classification.** IbPRIA 2013: 6th Iberian Conference on Pattern Recognition and Image Analysis, Madeira, Portugal. June 5-7, 2013

Genetic programming of classification prototypes

- Each tree codifies the prototype for a particular class



- An individual is a set of prototypes (a set of trees)

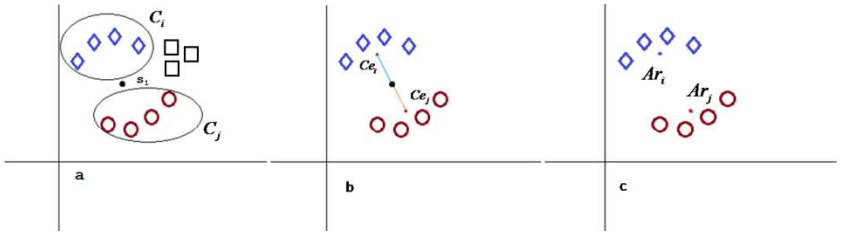


Genetic programming of classification prototypes

- Standard GP,
  - Two-crossover operators
  - Mutation
  - New (mitosis) operator

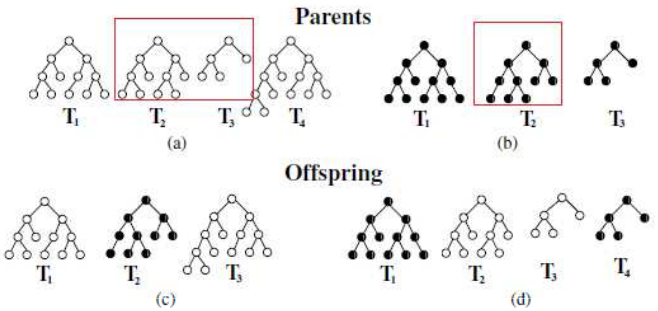
Algorithm 2 Overview of the *(GP²)* algorithm.

Require: Training data set $T = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_N, y_N)\}$;
  Set $\mathcal{S} = \{\}$;
  for $i = 1 \rightarrow k$ **do**
    Split training data (T) into development (D) and validation (V) sets;
    {where $T = D \cup V$, $D \cap V = \emptyset$;}
    $\mathcal{U} \leftarrow GPGP(D, V)$;
    $\mathcal{S} = S \cup U$
  end for
  return $\mathcal{S}$

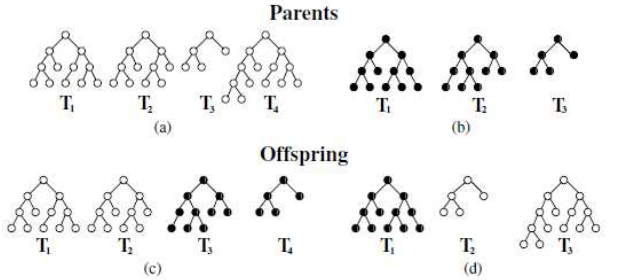


Mitosis

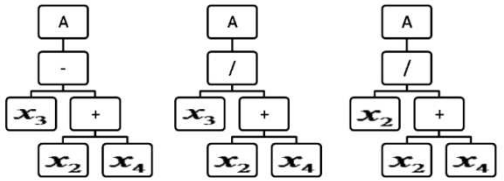
Crossover 1



Crossover 2



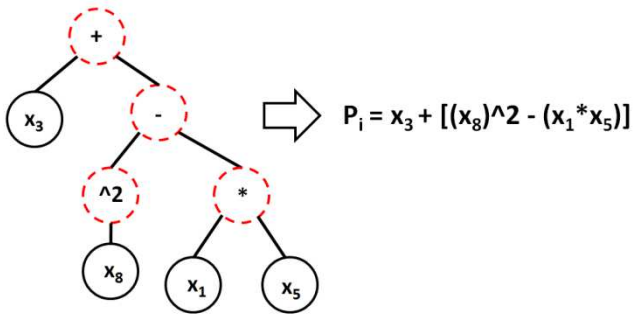
Mutation



$x_3 - (x_2 + x_4)$ $x_3 / (x_2 + x_4)$ $x_2 / (x_2 + x_4)$

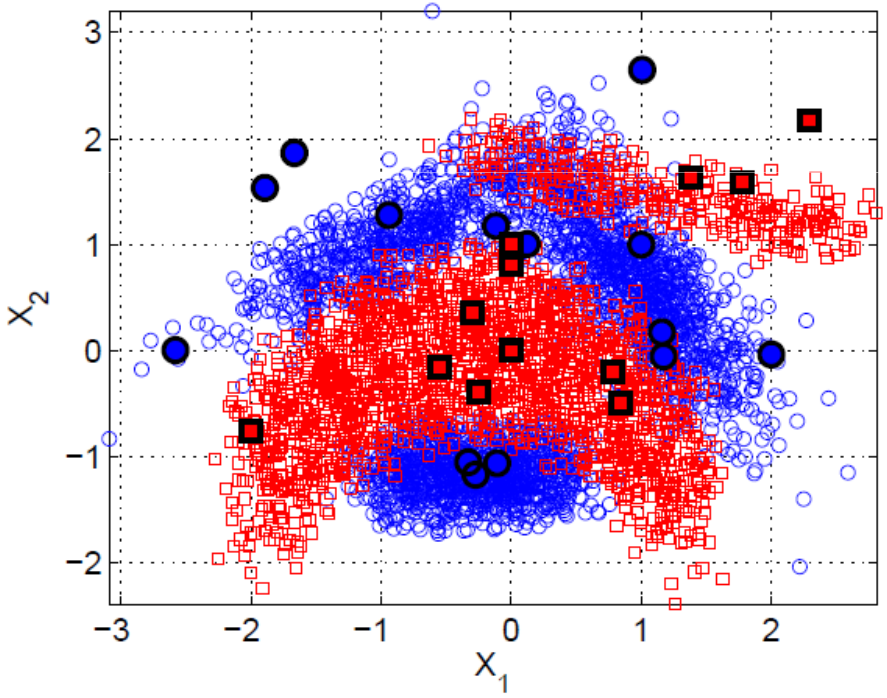
Genetic programming of classification prototypes

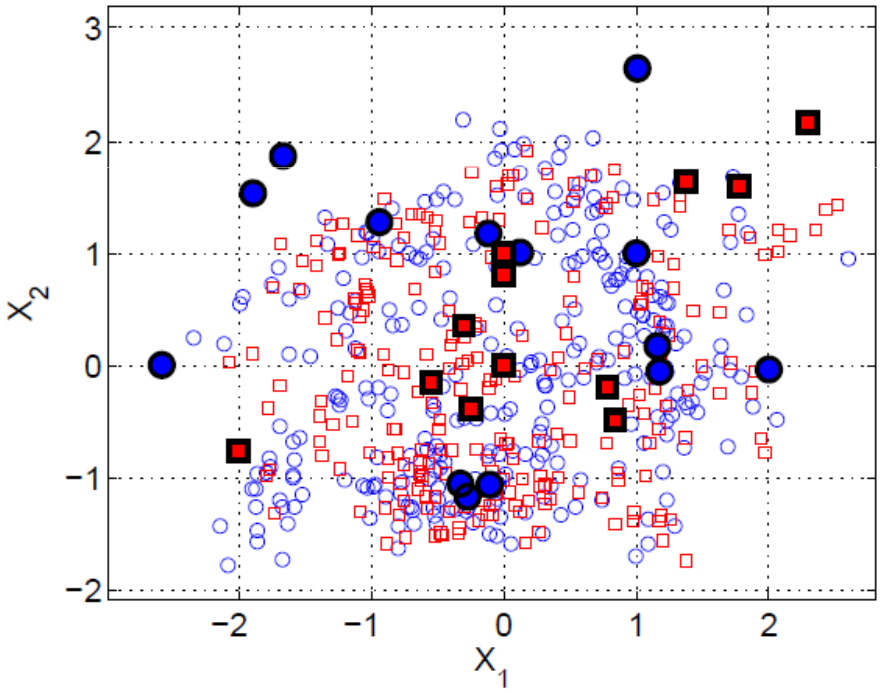
- Fitness function: accuracy in a validation set
- Terminals: all of the instances in the training set (except those used in the validation set)
- Function set: *, /, +, -, ^2, sqrt



Genetic programming of classification prototypes

- Sample solution:





H. J. Escalante, M. Graff, A. Morales-Reyes. **PGGP: Prototype Generation via Genetic Programming.** *Applied Soft Computing*, Minor revisions, 2014

Genetic programming of classification prototypes

- Sample prototypes:

sum([0.0232, -0.12],sum([-0.0097, -0.0883],abs(abs(sum(x(4632,:),x(1884,:))))))
sum([-0.17, -0.068],sum([-0.22, -0.23],sin(cos(my-power(x(1557,:))))))
sum(x(64,:),sin(cos(my-power(x(1557,:)))))
sum([-0.23, -0.18],sum([-0.29, 0.036],sum([-0.060, 0.031],abs(abs(sum(x(4632,:),x(1884,:))))))))
sum([-0.29, 0.036],sum([-0.06, 0.031],abs(abs(sum(x(4632,:),x(1884,:))))))
abs(x(4686,:))
minT(x(2412,:),x(351,:))
sum(abs(cos(tan(x(1224,:)))),my-power-3(mult(mult(x(2296,:),x(2250,:)),sin(x(1456,:)))))

H. J. Escalante, M. Graff, A. Morales-Reyes. **PGGP: Prototype Generation via Genetic Programming.** *Applied Soft Computing*, Minor revisions, 2014

Genetic programming of classification prototypes

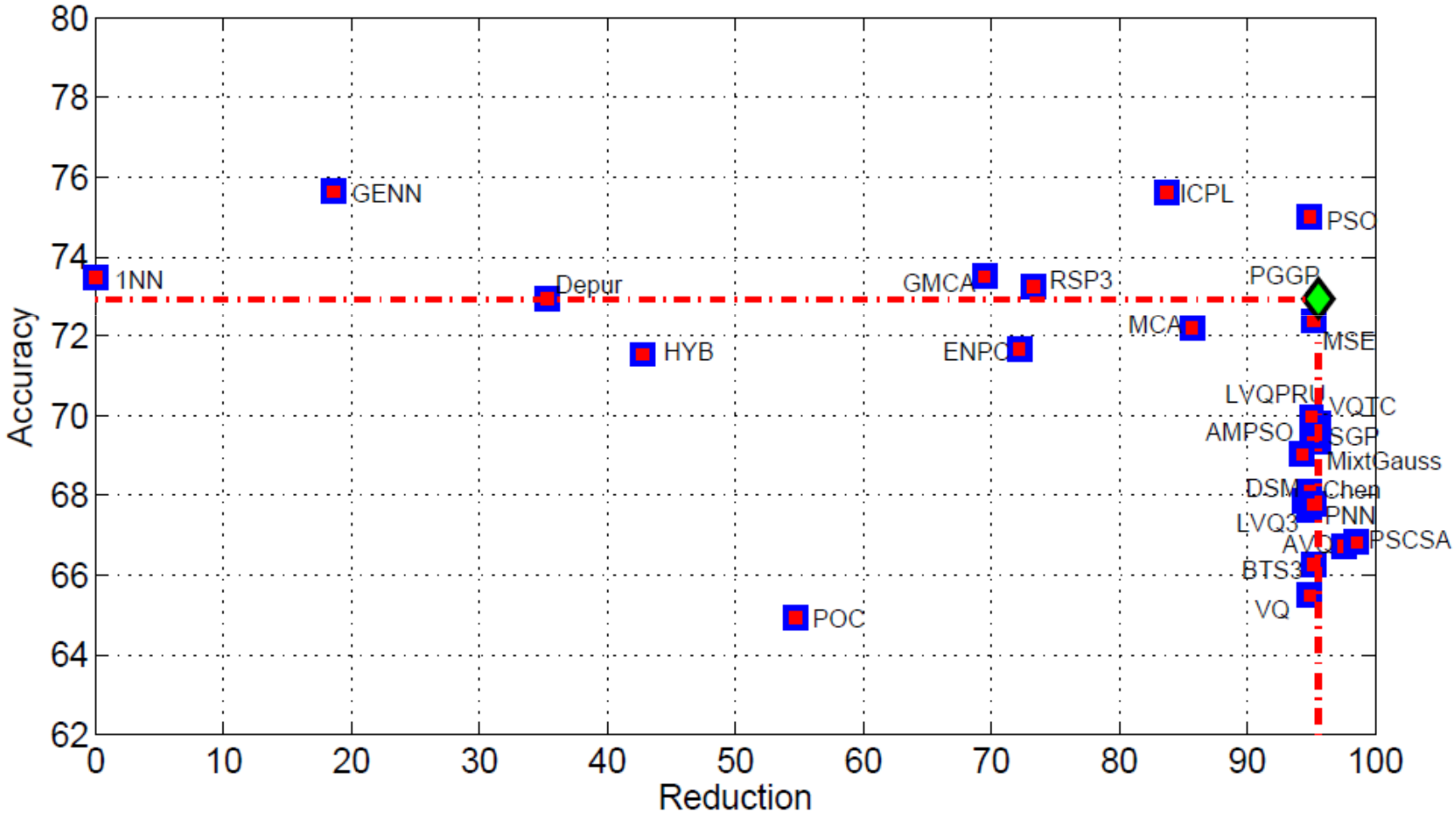
- Results: comparison with 24 other methods in a benchmark of 59 data sets for classification

	Test accuracy			Training set reduction		
Measure	All	Small	Large	All	Small	Large
PGGP	74.54%±5.18	72.95%±16.25	**81.47%±20.01**	96.81%	95.58%	99.42%
GENN	**78.48%±18.57**	**75.64%±15.45**	81.33%±21.70	17.19%	18.62%	15.76%
PSCSA	66.94%±20.39	66.82%±18.74	67.07%±22.05	**99.23%**	**98.58%**	**99.88%**
1NN	77.04%±19.44	73.48%±16.64	80.60%±22.24	0%	0%	0%

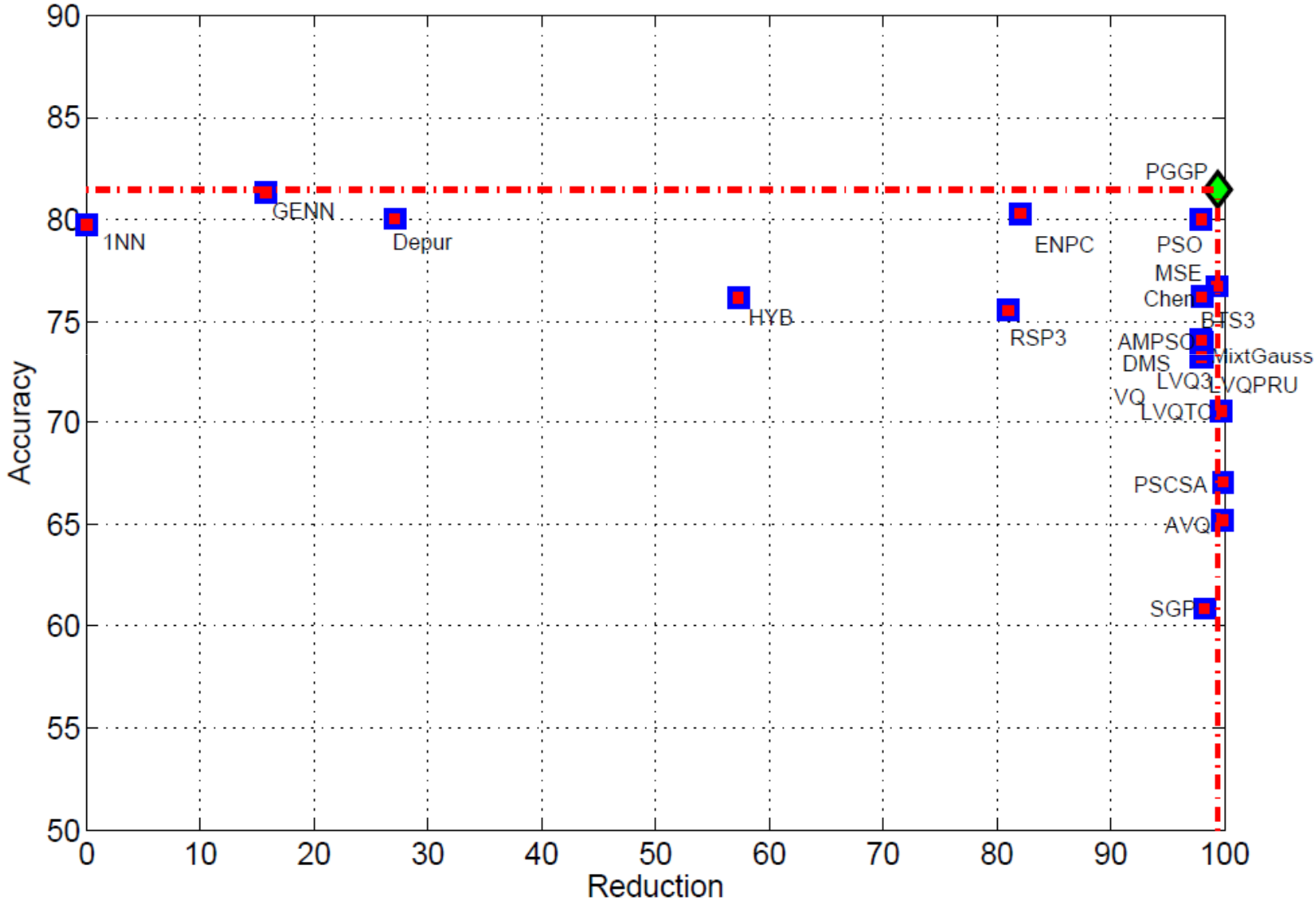
http://sci2s.ugr.es/pgtax/code.php

H. J. Escalante, K. Mendoza, M. Graff, A. Morales-Reyes. **Genetic Programming of Prototypes for Pattern Classification.** IbPRIA 2013: 6th Iberian Conference on Pattern Recognition and Image Analysis, Madeira, Portugal. June 5-7, 2013

Accuracy vs reduction (small)

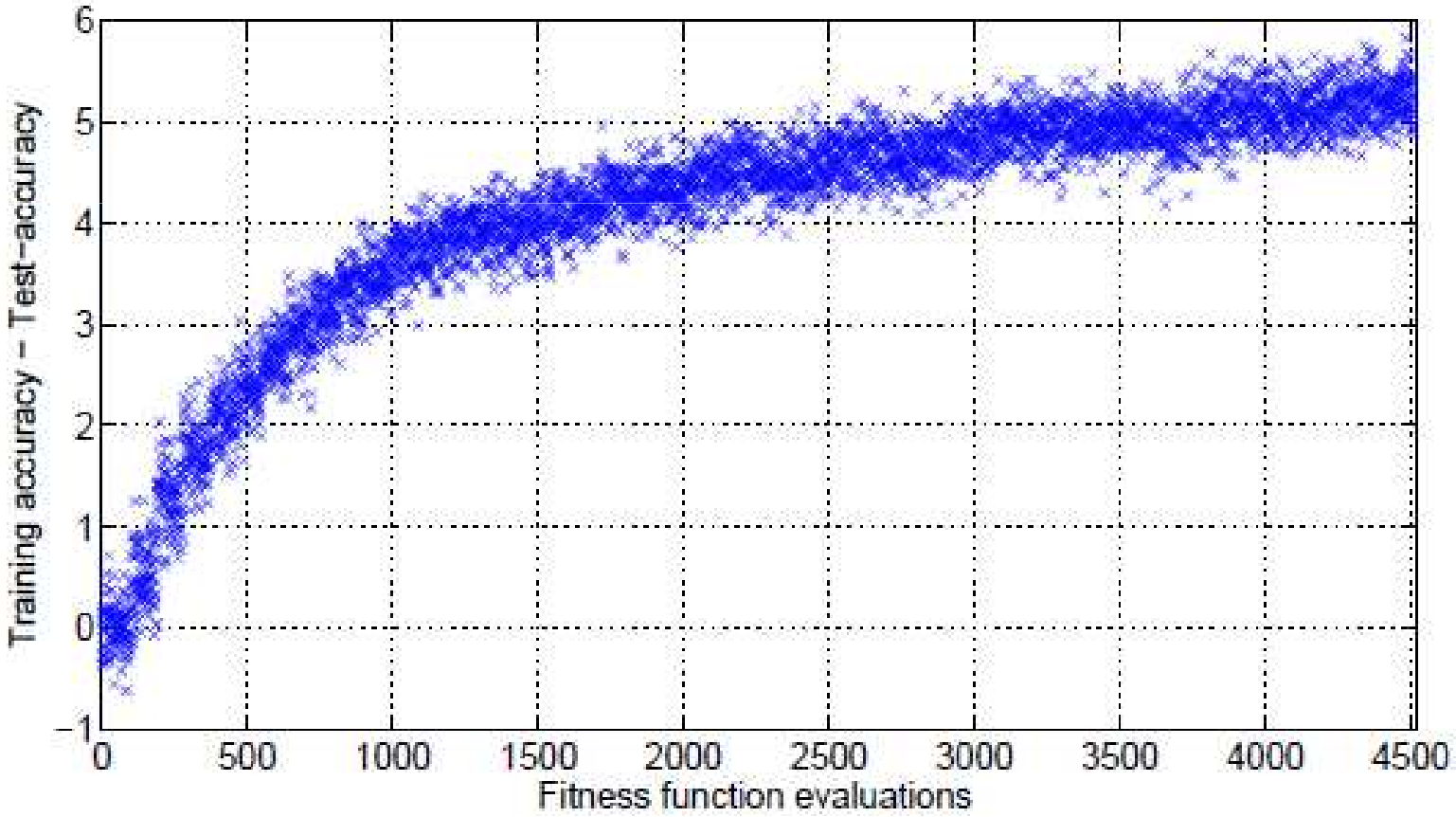


Accuracy vs reduction (large)



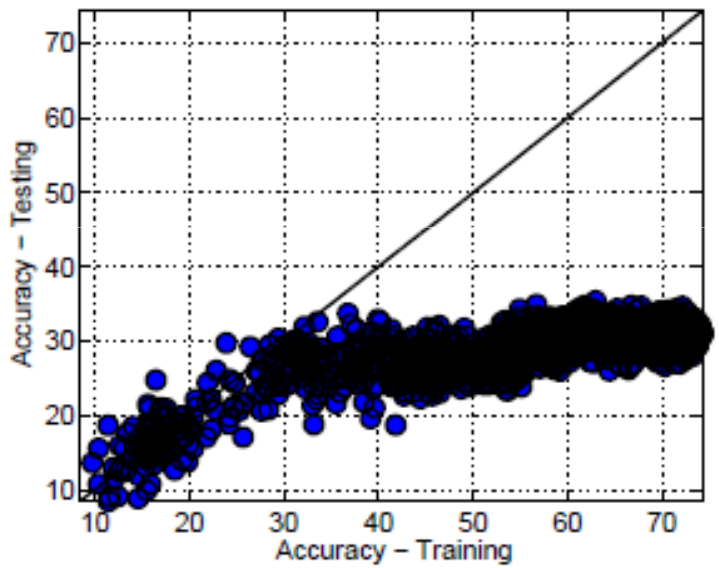
Genetic programming of classification prototypes

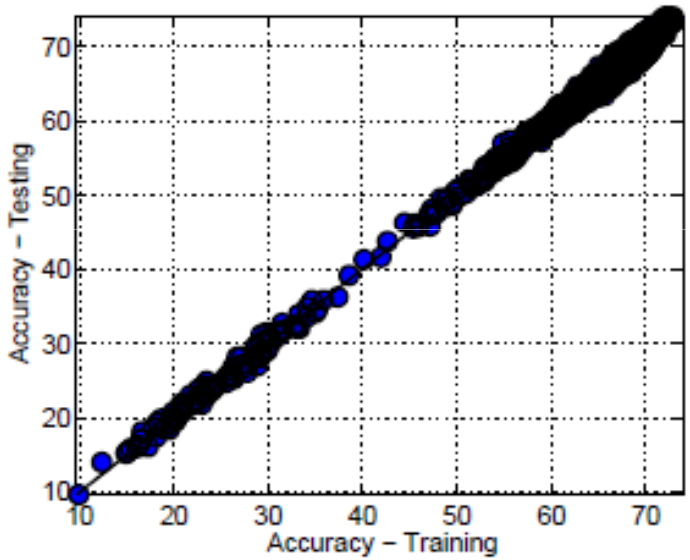
- Overfitting:



Genetic programming of classification prototypes

- Overfitting:





- Training set accuracy vs. Test set accuracy for the autos (205 instances and 6 classes) and car (1728 instances and 4 classes) data sets.

Genetic programming of classification prototypes

- Promising results with GP
- Too much work to do yet!
  - Overfitting is one of the main issues
  - Computationally expensive
  - Conflicting objectives
- Extensions:
  - Multi-objective prototype generation
  - Simultaneous generation of features and prototypes

Automatic generation of classification prototypes

Ensemble generation

Term-weighting scheme learning

SOME APPLICATIONS OF GENETIC PROGRAMMING IN MACHINE LEARNING

H. J. Escalante, N. Acosta, A. Morales, A. Gago. **Genetic Programming of Heterogeneous Ensembles for Classification,** Progress in Pattern Recognition, Image Analysis, Computer Vision and Applications,, CIARP, Havana, Cuba, November 20-23, Proceedings, Part I, LNCS 8258, pp. 9—16, Springer, 2013.

N. Acosta, A. Morales, H. J. Escalante, A. Gago. **Learning to Assemble Classifiers via Genetic Programming.** International Journal of Pattern Recognition and Artificial Intelligence, 2014.

Introduction

- Ensembles classification principle:
  - Combining classifiers models to improve overall performance
- Homogeneous ensembles
  - Combine outputs of the same classifier trained in different data
- Heterogeneous ensembles
  - Combining classifiers of different nature such as decision trees, neural networks, etcetera
- Normally, combination of classifiers' outputs is carried out by voting or linear combination of scores.

Can we do it better than that?

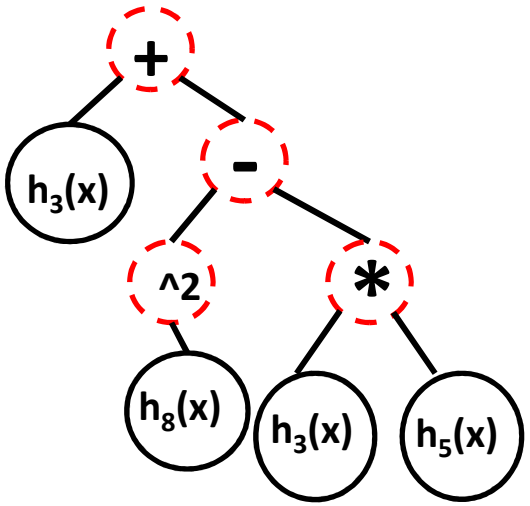
Introduction

- An evolutionary algorithm is proposed to learn a fusion function that combines the outputs of heterogeneous classifiers in order to maximize the accuracy of the ensemble
  - Genetic programming
    - A population of fusion functions is evolved

H. J. Escalante, N. Acosta, A. Morales, A. Gago. **Genetic Programming of Heterogeneous Ensembles for Classification,** Progress in Pattern Recognition, Image Analysis, Computer Vision and Applications,, CIARP, Havana, Cuba, November 20-23, Proceedings, Part I, LNCS 8258, pp. 9—16, Springer, 2013.

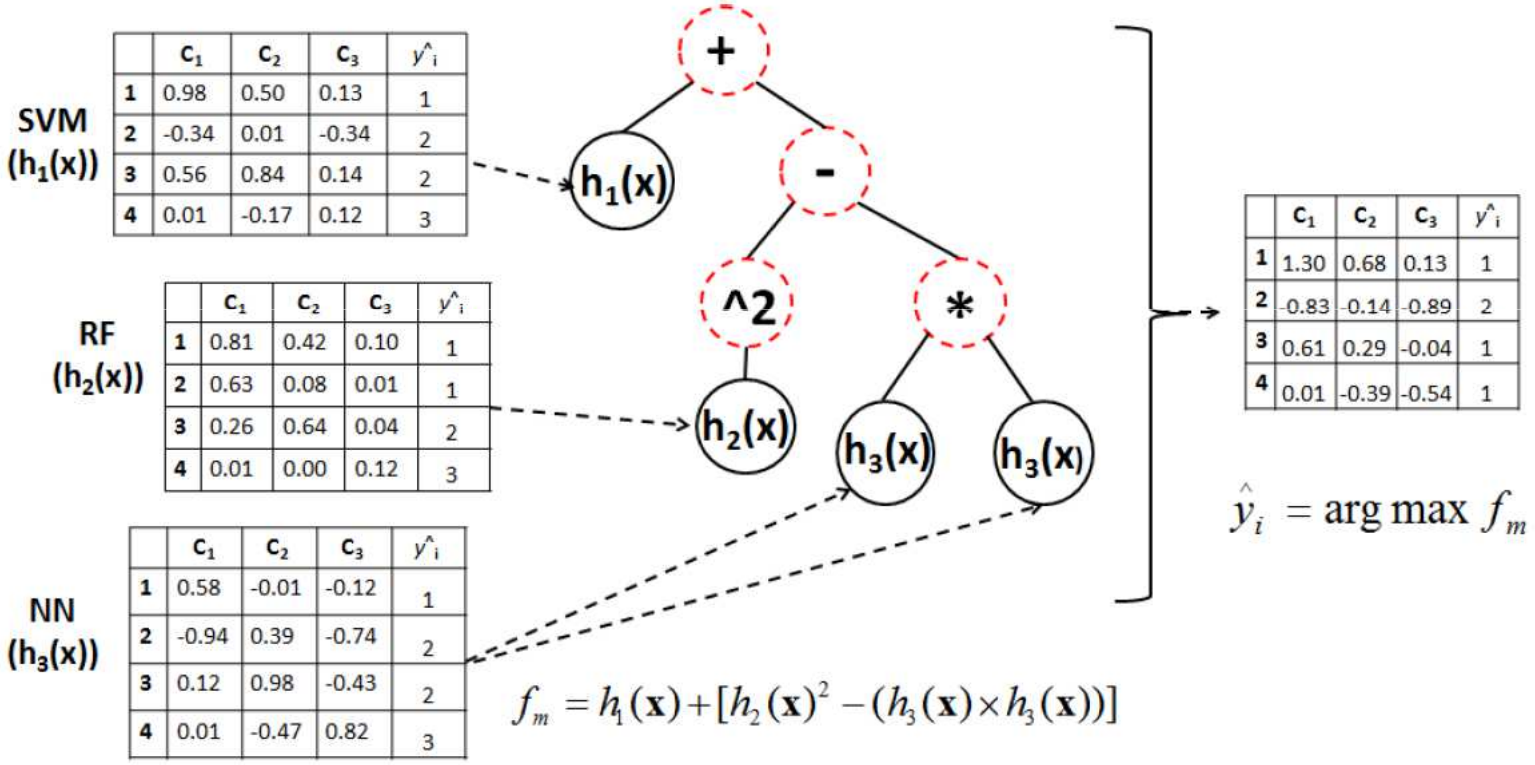
Genetic programming of ensembles

- GP encodes solutions using trees
  - Nodes are classification models with constants (incorporating a weighting factor)
  - Non-leaf nodes are operators $\{+, -, \times, \div, \sqrt{}, log10\}$



$$f_m = h_3(\mathbf{x}) + [h_8(\mathbf{x})^2 - (h_3(\mathbf{x}) \times h_5(\mathbf{x}))]$$

Genetic programming of ensembles



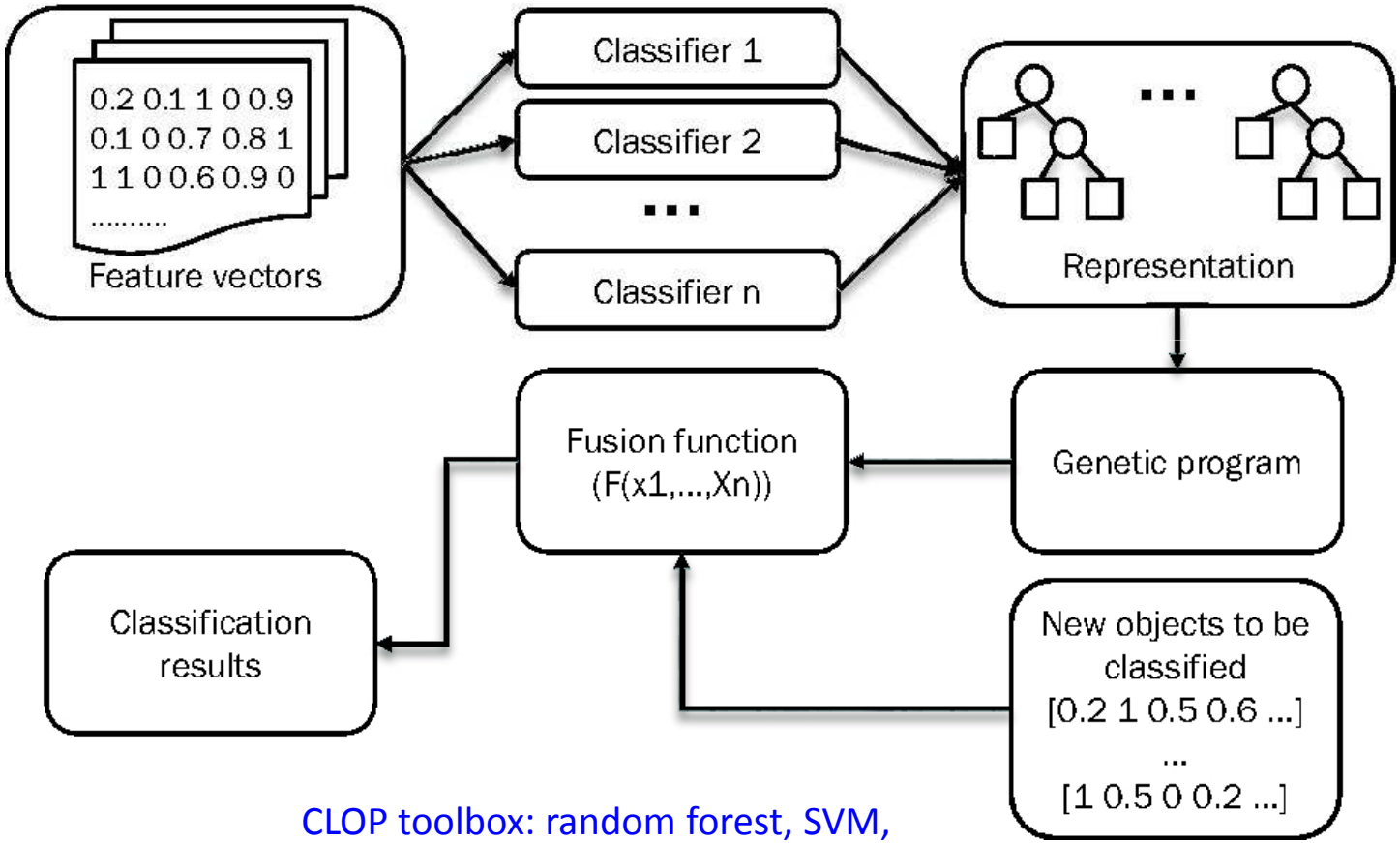
	c_1	c_2	c_3	$\hat{y}_i$
1	0.98	0.50	0.13	1
2	-0.34	0.01	-0.34	2
3	0.56	0.84	0.14	2
4	0.01	-0.17	0.12	3

	c_1	c_2	c_3	$\hat{y}_i$
1	0.81	0.42	0.10	1
2	0.63	0.08	0.01	1
3	0.26	0.64	0.04	2
4	0.01	0.00	0.12	3

	c_1	c_2	c_3	$\hat{y}_i$
1	0.58	-0.01	-0.12	1
2	-0.94	0.39	-0.74	2
3	0.12	0.98	-0.43	2
4	0.01	-0.47	0.82	3

	c_1	c_2	c_3	$\hat{y}_i$
1	1.30	0.68	0.13	1
2	-0.83	-0.14	-0.89	2
3	0.61	0.29	-0.04	1
4	0.01	-0.39	-0.54	1

Genetic programming of ensembles



CLOP toolbox: random forest, SVM,
Klogistic, linear kridge, non-linear kridge,
1NN, 3NN, Naïve Bayes, gkridge, NN

Experimental results

- GP configuration
  - GPLAP framework
  - Standard crossover and mutation operators
  - Population initialization: ramped-half-and-half
  - Population size 50 individuals
  - 100 generations
  - 10 runs
- Experimental data
  - UCI- data sets
  - SCEF – object recognition problem
    - 6244 image-regions
    - 737 attributes
    - 10 classes
    - Dataset division
      - 3615 testing
      - 2629 training

N. Acosta, A. Morales, H. J. Escalante, A. Gago. **Learning to Assemble Classifiers via Genetic Programming.** International Journal of Pattern Recognition and Artificial Intelligence, 2014.

Experimental results - SCEF

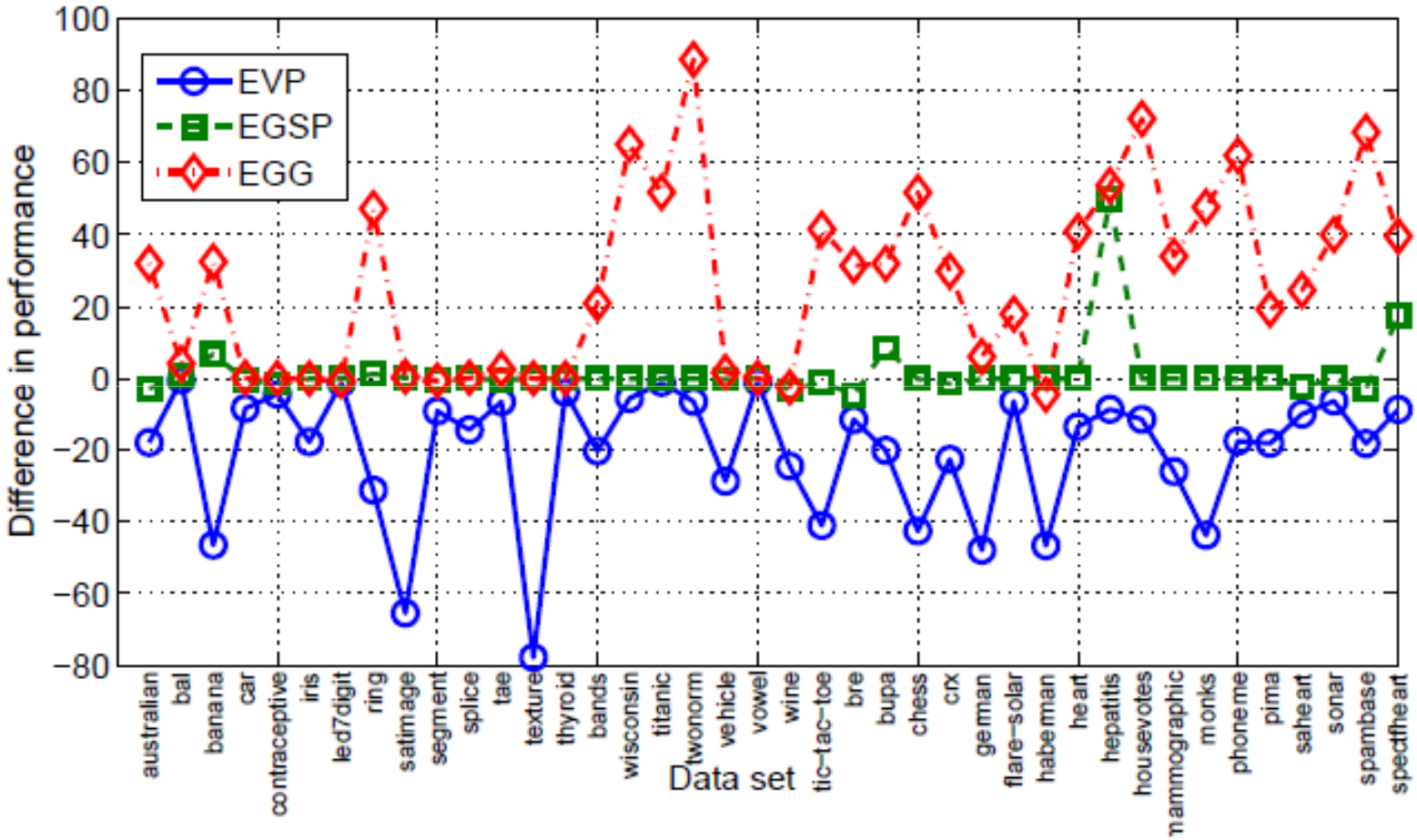
Table 2. Results (%) obtained by individual classifiers in terms of accuracy/f_1 measure over SCEF data set.

	RF	SVM	Klogistic	Kridge-l	Kridge-n	1NN	3NN	N.Bayes	Gkridge	Neural N
Acc.	**90.70**	55.10	70.60	13.64	74.70	69.30	69.10	26.50	20.60	55.80
f_1	**79.30**	49.90	62.80	2.400	63.10	60.10	57.40	21.60	3.421	37.70

Table 3. Results (%) obtained by different strategies over SCEF data set when optimizing accuracy (top) and f_1 (bottom). AVE: raw fusion; GPE-a: GP uses only sums; GPE: proposed GP.

	AVE	AVE-Top5	GPE-a	GPE-a-Top5	GPE	GPE-Top5
Acc.	31.50	81.40	90.80(0.001)	91.10(0.002)	**92.30(0.002)**	91.20(0.001)
f_1.	27.2	71.90	80.40(0.001)	80.40(0.001)	**85.30(0.003)**	80.55(0.003)

Experimental results - UCI



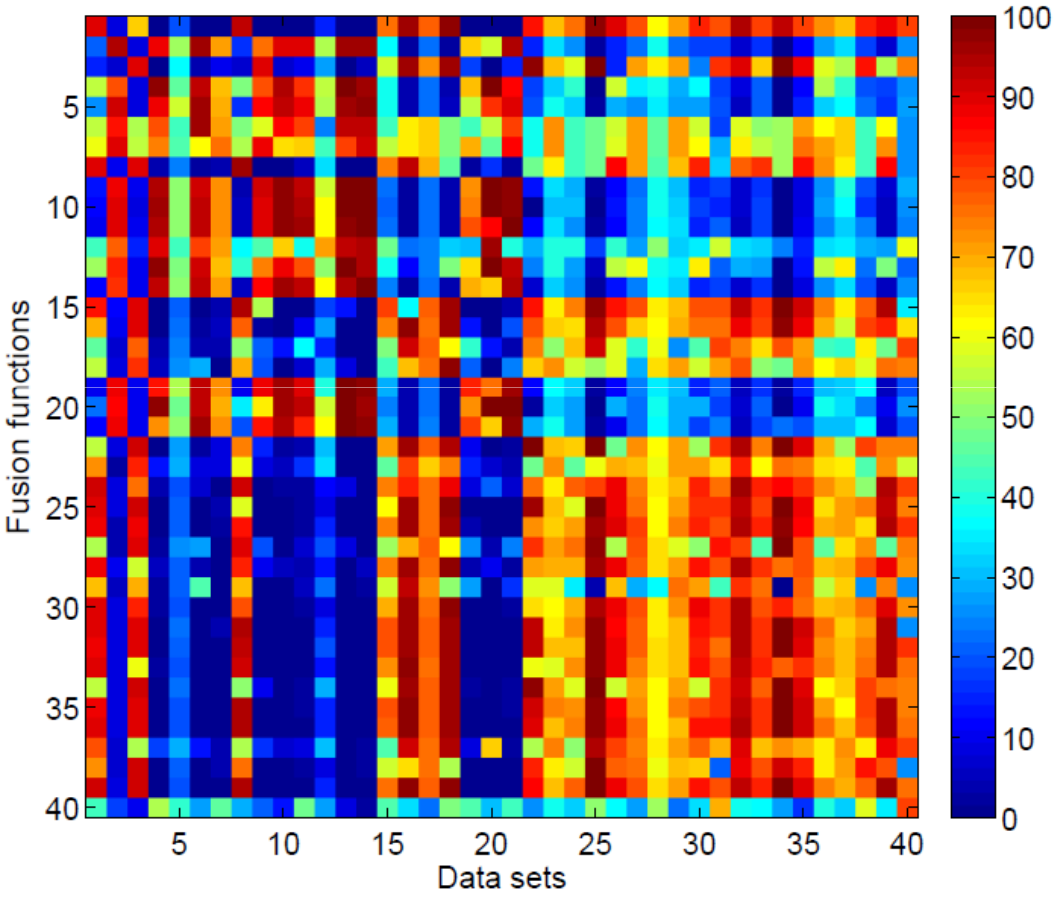
	B-Classifier	EVP	EGSP	EGG
Acc.	61.01(25.18)	40.23(27.91)	62.68(24.45)	**85.75(12.21)**
f_1.	63.50(26.32)	31.06(29.00)	62.91(26.71)	**84.49(14.94)**

Experimental results - UCI

Table 4. Average and standard deviation performances for B-Classifier: best classifier; AVE: raw fusion; GPE-a: GP using only sums; GPE: proposed GP.

	B-Classifier	AVE	GPE-a	GPE
Acc.	$61.01(25.18)_{(1,0,-)}$	$40.23(27.91)_{(-,-,-)}$	$62.68(24.45)_{(0,1,-)}$	$\mathbf{85.75(12.21)}_{(1,1,1)}$
f_1.	$63.50(26.32)_{(1,0,-)}$	$31.06(29.00)_{(-,-,-)}$	$62.91(26.71)_{(0,1,-)}$	$\mathbf{84.49(14.94)}_{(1,1,1)}$

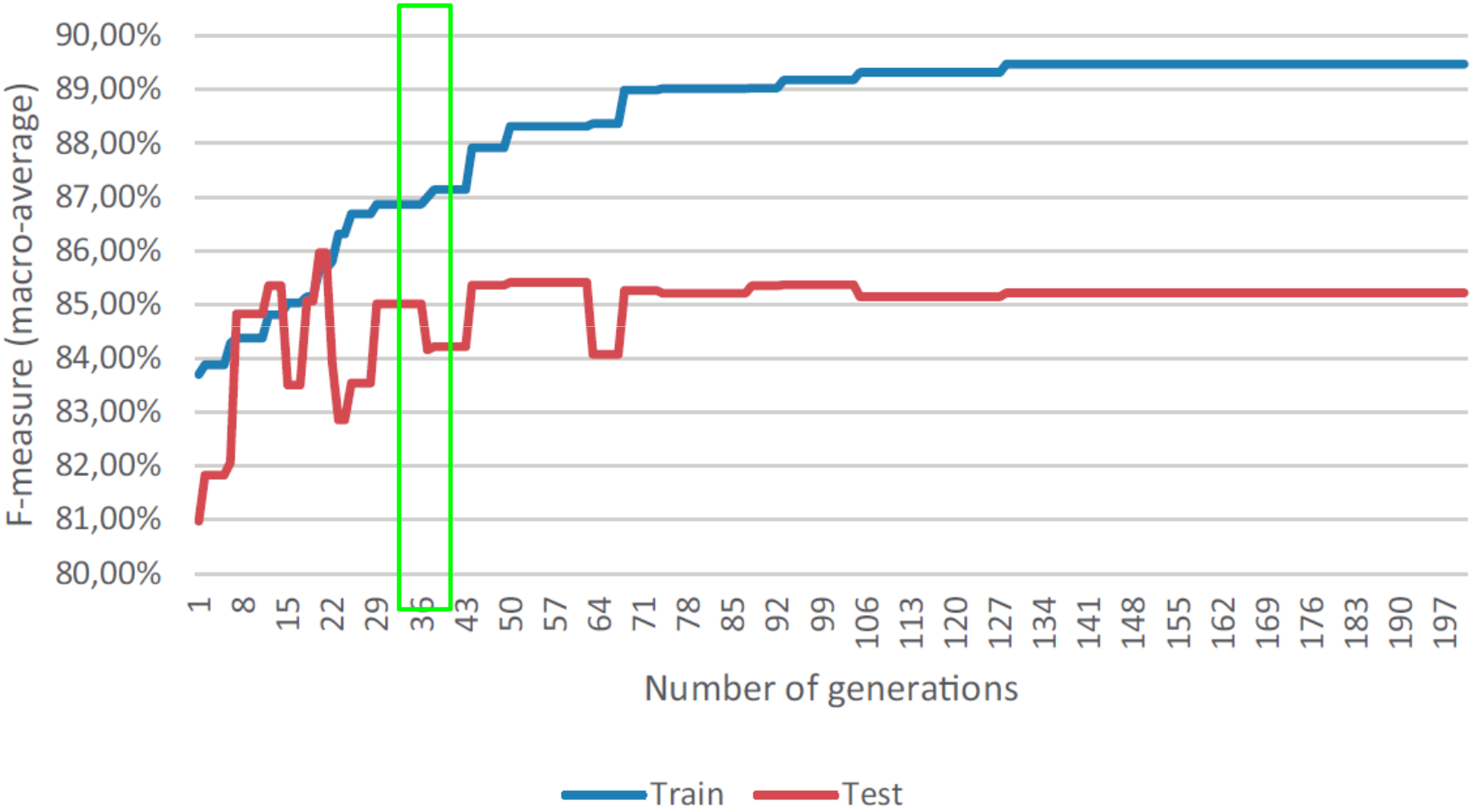
Experimental results - UCI



- Generalization of learned functions

ID	Measure	Accuracy	f-measure
1	Perf. Best classifier	59.70	62.12
2	Perf. when using the ad-hoc weight	87.35	86.87
3	Avg. performance overall data sets	48.81	43.51
4	Maximum performance	97.08	97.47
5	Avg. number of improved data sets	17.22	15.22

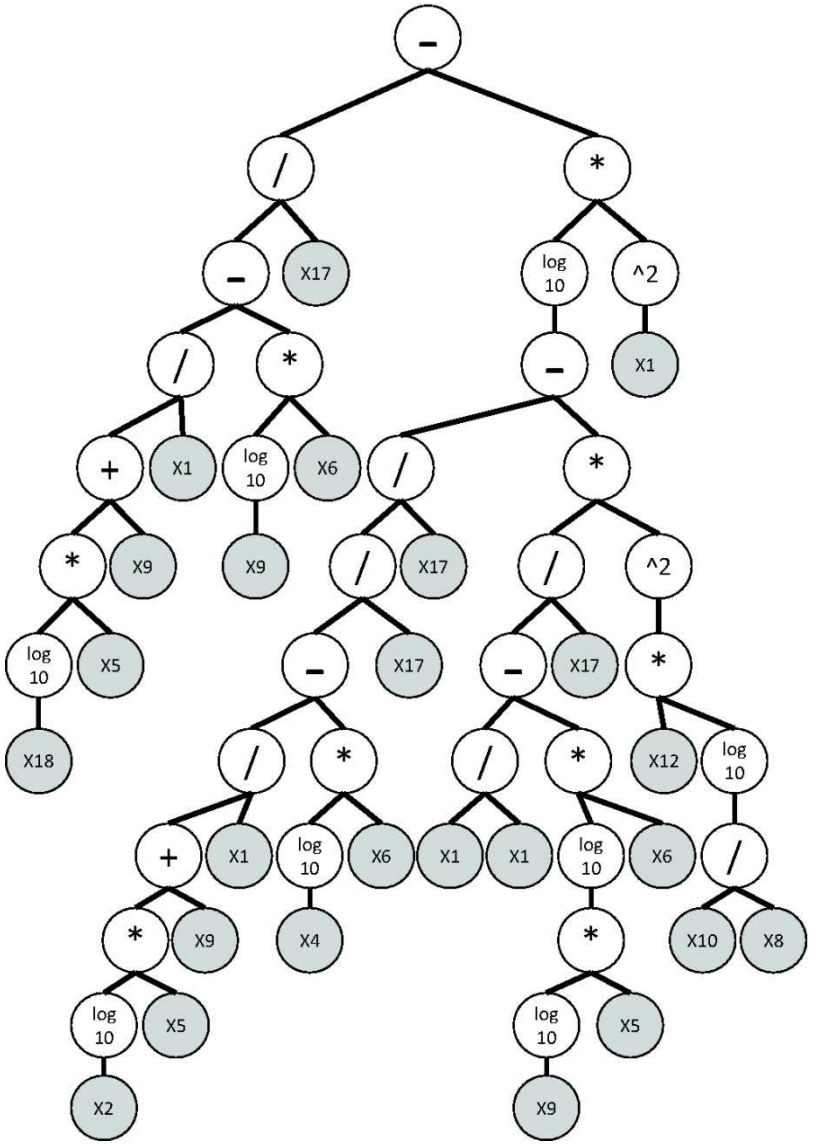
Experimental results: overfitting?



Experimental results

- Individual

example after 200 generations



Conclusions

- GP to build heterogeneous ensembles has shown to be an effective way to improve their performance
- Both performance metrics were improved when ensembles use all considered classifiers
- Varying some configuration GP parameters allows to modify the overall ensemble performance
  - Population size
  - Number of generations

Automatic generation of classification prototypes

Ensemble generation

Term-weighting scheme learning

SOME APPLICATIONS OF GP ON MACHINE LEARNING PROBLEMS

H. J. Escalante, M. García-Limón, A. Morales, M. Graff, M. Montes-y-Gómez, E. Morales. **Learning term-weighting schemes for text classification.** *Submitted, October 6, 2014*

M. García-Limón, H. J. Escalante, M. Montes-y-Gómez, A. Morales, E. Morales. **Genetic programming of text representations.** *GECCO Comp'14, pp. 1459-1460, (Late-breaking abstract, Oral presentation), Vancouver, Canada, July, 12-17, 2014.*

Aprendizaje de Esquemas de Pesado para la Clasificación de Textos

Hugo Jair Escalante

Laboratorio de
Tecnologías del Lenguaje
Coordinación de Ciencias Computacionales
Instituto Nacional de Astrofísica, Óptica y
Electrónica
INAOE

Hugo Jair Escalante, Mauricio García-Limón, Alicia Morales, Mario Graff, Manuel Montes-y-Gómez, Eduardo Morales. **Learning term-weighting schemes for text classification.** *Submitted, October 6, 2014*

Información textual

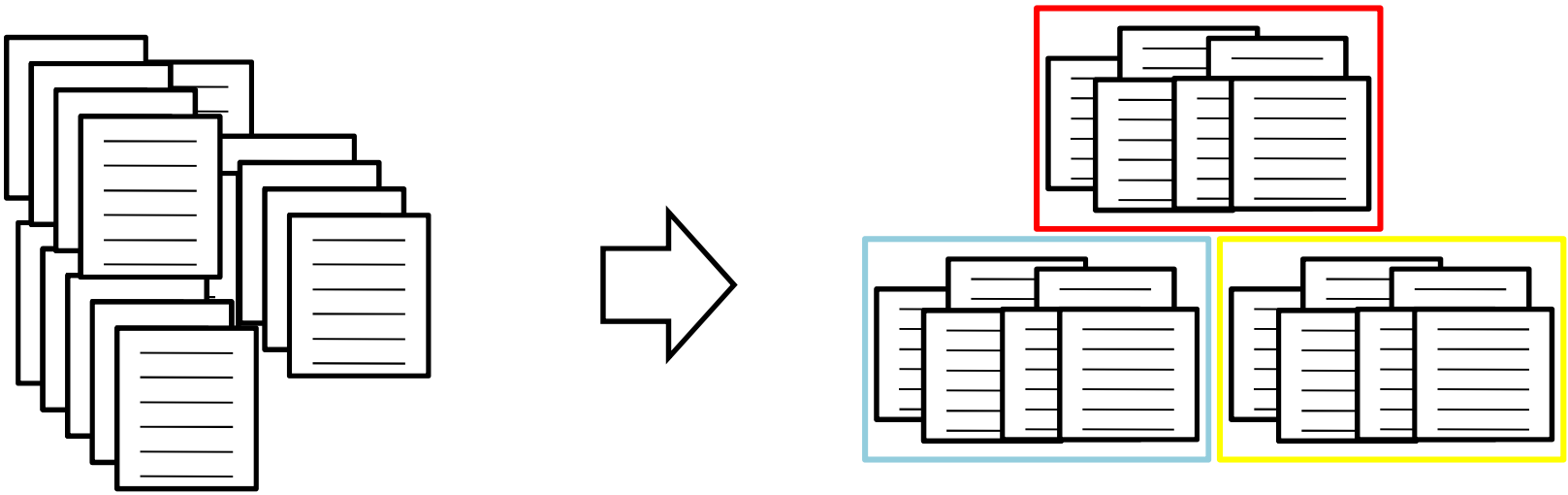
- Crecimiento considerable de información textual disponible electrónicamente
- Incapacidad de los seres humanos para procesar tanta información en tiempos razonables
- Minería de textos: *"el proceso de derivar información de alta calidad a partir del texto"*

http://www.qmee.com

Clasificación de textos (TC)

- La asignación de categorías predefinidas a documentos de texto.

- **Categorías**: dependen del objeto de interés que se quiera modelar, e.g., contenido/temática, estilo de escritura, polaridad, etc.



Documents (e.g., news articles)

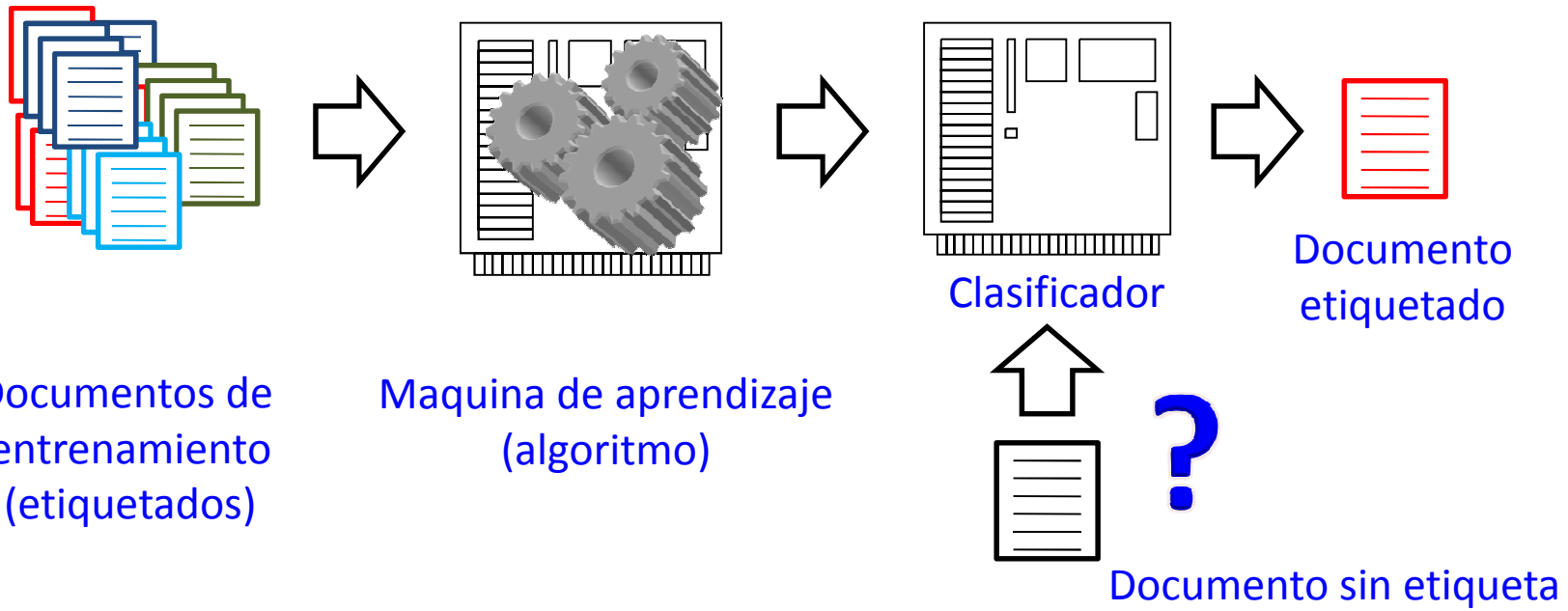
Categories/classes
(e.g., sports, religion, economy)

Clasificación de textos

- Muy probablemente, TC es el tópico más estudiado dentro de las tecnologías del lenguaje humano:
  - Se puede considerar un problema resuelto en ciertos escenarios: e.g., clasificación de noticias y filtrado de spam
- Sin embargo, muchas variantes son problemas abiertos; igualmente, tareas actuales pueden verse como TC y están lejos de ser resueltas:
  - Cross-domain, multilingüe, …
  - Perfilado de autores, atribución de autoría, minería de opiniones, detección de sarcasmo, …

Clasificación de textos

- Si bien la clasificación de textos es un proceso que requiere de procesamiento de alto nivel semántico y herramientas lingüísticas, el enfoque estadístico ha probado ser muy competitivo en muchos escenarios:

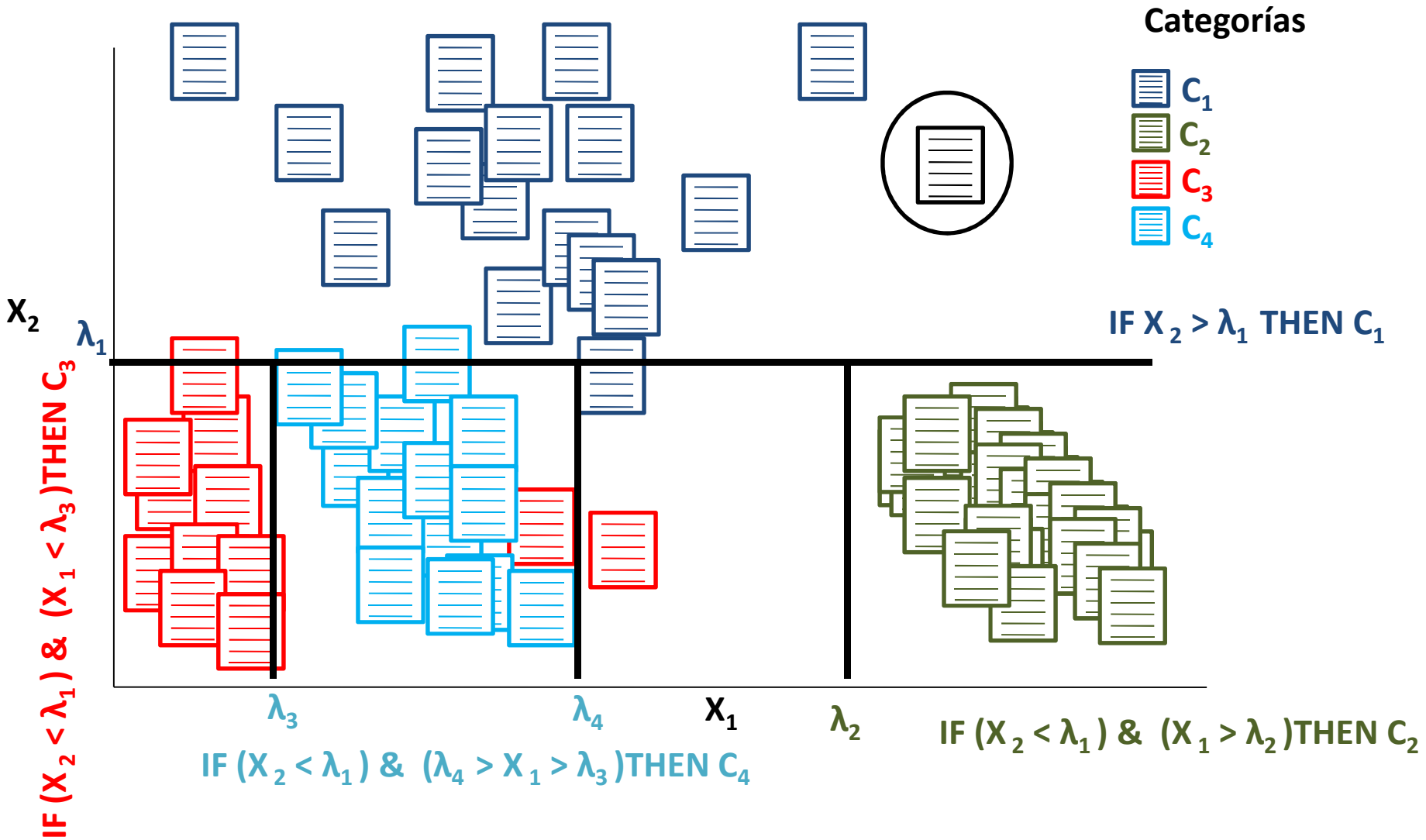


Clasificación de textos

- Se asume que los documentos pueden ser representados como puntos en un espacio (VSM)

- Se usan modelos de aprendizaje computacional para hacer inferencia sobre nuevos documentos

Clasificación de textos



Representación de documentos (VSM)

Términos en el vocabulario
(Unidades básicas, expresando el *contenido del documento*)

	t_1	t_1	…	$t_{\|V\|}$
d_1				
d_2				
:		$w_{i,j}$		
d_m				

Documentos en el corpus
(*un vector/fila por documento*)

V: vocabulario del corpus *(i.e., todos los términos diferentes que ocurren en el corpus)*

Peso indicando la contribución del Término *j* en el documento *i*.

¿Qué términos son *buenos* atributos?

¿Cómo extraerlos/identificarlos?

¿Cómo estimar el peso?

Representación de documentos

- Esquemas no Supervisados: Son los esquemas tradicionales, propuestos para recuperación de información, e.g., *tf, tf-idf*, *Booleano*, etc.

- Esquemas Supervisados: Son esquemas que incorporan información discriminativa, diseñados particularmente para clasificación, e.g., *tf-ig, tf-chi*^{2}, etc.

Acronym	Name	Formula	Description	Ref.
B	Boolean	$x_{i,j} = 1_{\{\#(t_i,d_j)>0\}}$	Indicates the prescense/abscense of terms	[2]
TF	Term-Frequency	$x_{i,j} = \#(t_i, d_j)$	Accounts for the frequency of occurrence of terms	[2]
TF-IDF	TF - Inverse Document Frequency	$x_{i,j} = \#(t_i, d_j) \times \log(\frac{N}{df(t_j)})$	An TF scheme that penalizes the frequency of terms across the collection	[2]
TF-IG	TF - Information Gain	$x_{i,j} = \#(t_i, d_j) \times IG(t_j)$	TF scheme that weights term occurrence by its information gain across the corpus.	[11]
TF-CHI	TF - Chi-square	$x_{i,j} = \#(t_i, d_j) \times CHI(t_j)$	TF scheme that weights term occurrence by its χ^2 statistic	[11]
TF-RF	TF - Relevance Frequency	$x_{i,j} = \#(t_i, d_j) \times \log(2 + \frac{TP}{\max(1,TN)})$	TF scheme that weights term occurrence by its χ^2 statistic	[10]

Aprendizaje de Esquemas de Pesado

- Problemática:
  - Esquemas de pesado se han propuesto manualmente,
  - Casi siempre se usan esquemas no-supervisados,
  - Se usa un mismo esquema de pesado para casi todas las tareas.

- ¿Es posible generar de forma automática esquemas de pesado para TC, con los que se obtengan mejores resultados que con esquemas estándar?

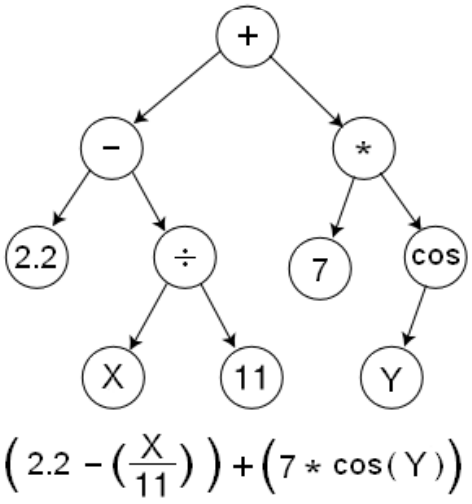
Aprendizaje de Esquemas de Pesado

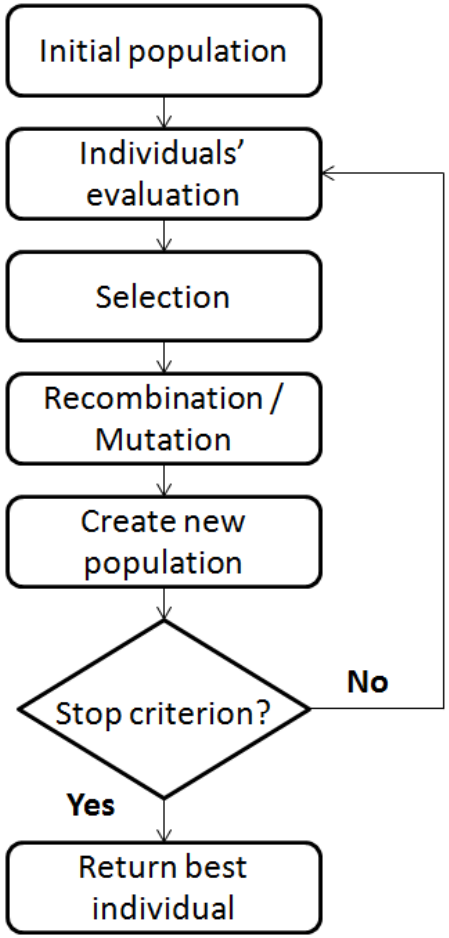
- Idea: proveer a un método de aprendizaje con primitivas para generar esquemas de pesado y aprender esquemas de pesado ad-hoc para la clasificación de textos

- Implementación usando programación genética (PG): Codificar esquemas de pesado como arboles y usar PG para encontrar el esquema que maximiza la precisión de clasificación en datos de entrenamiento.

PG- panorama general

- Algoritmo evolutivo donde los individuos pueden ser estructuras de datos complejas (e.g., arboles, grafos, etc.)





J. Koza. **Genetic Programming: On the Programming of Computers by Means of Natural Selection**. MIT Press, 1992.

Aprendizaje de esquemas de pesado con PG

Variable	Meaning
$\mathbf{W}_1$	N, Constant matrix, the total number of training documents.
$\mathbf{W}_2$	$\|V\|$, Constant matrix, the number of terms.
$\mathbf{W}_3$	CHI, Matrix containing in each row the vector of χ^2 weights for the terms.
$\mathbf{W}_4$	IG, Matrix containing in each row the vector of information gain weights for the terms.
$\mathbf{W}_5$	$TF - IDF$, Matrix with the TF-IDF term weighting scheme.
$\mathbf{W}_6$	TF, Matrix containing the TF term-weighting scheme.
$\mathbf{W}_7$	FGT, Matrix containing in each row the global term-frequency for all terms.
$\mathbf{W}_8$	TP, Matrix containing in each row the vector of true positives for all terms.
$\mathbf{W}_9$	FP, Matrix containing in each row the vector of false positives.
$\mathbf{W}_{10}$	TN, Matrix containing in each row the vector of true negatives.
$\mathbf{W}_{11}$	FN, Matrix containing in each row the vector of false negatives.
$\mathbf{W}_{12}$	$Accuracy$, Matrix in which each row contains the accuracy obtained when using the term as classifier.
$\mathbf{W}_{13}$	$Accuracy\_Balance$, Matrix containing the AC_Balance each (term, class).
$\mathbf{W}_{14}$	BNS, An array that contains the value for each BNS per (term, class).
$\mathbf{W}_{15}$	$DFreq$, Document frequency matrix containing the value for each (term, class).
$\mathbf{W}_{16}$	$FMeasure$, F-Measure matrix containing the value for each (term, class).
$\mathbf{W}_{17}$	$OddsRatio$, An array containing the OddsRatio term-weighting.
$\mathbf{W}_{18}$	$Power$, Matrix containing the Power value for each (term, class).
$\mathbf{W}_{19}$	$ProbabilityRatio$, Matrix containing the ProbabilityRatio each (term, class).
$\mathbf{W}_{20}$	Max_Term, Matrix containing the vector with the highest repetition for each term.
$\mathbf{W}_{21}$	RF, Matrix containing the RF vector.
$\mathbf{W}_{22}$	$TF \times RF$, Matrix containing $TF \times RF$.

Aprendizaje de esquemas de pesado con PG

- Función objetivo: maximizar f_1-measure usando validación cruzada en el conjunto de entrenamiento
  - low-rank linearized SVM (LLSVM)

- Se usa una implementación de PG estándar con parámetros por default

Silva, S., Almeida, J., 2003. **GPLAB-a genetic programming toolbox for matlab.** Proceedings of the Nordic MATLAB conference. pp. 273–278.

Zhang, K., Lan, L., Wang, Z., Moerchen, F., **Scaling up kernel SVM on limited resources: A low-rank linearization approach.** Proceedings ofthe 15th International Conference on Artificial Intelligence and Statistics(AISTATS), 2012.

Resultados experimentales

- Experimentos con 3 tipos de tareas:
  - Clasificación temática
  - Atribución de autoría
  - Clasificación de imágenes (BVW)
- Se redujo el vocabulario por frecuencia para acelerar el proceso de optimización
- Se realizaron 5-corridas del programa genético por cada conjunto de datos

Text categorization				
Data set	**Classes**	**Terms**	**Train**	**Test**
Reuters-8	8	23583	5339	2333
Reuters-10	10	25283	6287	2811
20-Newsgroup	20	61188	11269	7505
TDT-2	30	36771	6576	2818
WebKB	4	7770	2458	1709
Classic-4	4	5896	4257	2838
CADE-12	12	193731	26360	14618
Authorship attribution				
Data set	**Classes**	**Terms**	**Train**	**Test**
CCA	10	15587	500	500
Poetas	5	8970	71	28
Football	3	8620	52	45
Business	6	10550	85	90
Poetry	6	8016	145	55
Travel	4	11581	112	60
Cricket	4	10044	98	60
Image Classification				
Data set	**Classes**	**Terms**	**Train**	**Test**
Caltech-101	101	12000	1530	1530
Caltech-tiny	5	12000	75	75

Resultados experimentales

- Clasificación temática:

	PG- Avg.		Best baseline		
Data set	f_1	Acc.	f_1	Acc.	Baseline
Reuters-8	**90.56±1.43**	**91.35±1.99**	86.94	88.63	TF
Reuters-10	**88.21±2.69**	91.84±1.01	85.24	**93.25**	TFIDF
20-Newsgroup	**66.23±3.84**	**67.97±4.16**	59.21	61.99	TF
TDT-2	**96.95±0.41**	**96.95±0.57**	95.20	95.21	TFIDF
WebKB	**88.79±1.26**	**89.12±1.30**	87.49	88.62	B
Classic-4	**94.75±1.08**	**95.42±0.67**	94.68	94.86	TF
CADE-12	**41.03±4.45**	**53.80±4.0**	39.30	41.89	TF

Resultados experimentales

- Atribución de autoría:

	PG- Avg.		**Best baseline**		
Data set	f_1	Acc.	f_1	Acc.	Baseline
CCA-10	$\mathbf{70.32^{+}_{-}2.73}$	$\mathbf{73.72^{+}_{-}2.14}$	65.90	73.15	TF-IG
Poetas	$\mathbf{72.23^{+}_{-}1.49}$	$\mathbf{72.63^{+}_{-}1.34}$	70.61	71.84	TF-IG
Football	$76.37^{+}_{-}9.99$	$83.76^{+}_{-}4.27$	**76.45**	**83.78**	TF-CHI
Business	$\mathbf{78.08^{+}_{-}4.87}$	$\mathbf{83.58^{+}_{-}1.57}$	73.77	81.49	TF-CHI
Poetry	$\mathbf{70.03^{+}_{-}7.66}$	$74.05^{+}_{-}7.38$	59.93	**76.71**	B
Travel	$\mathbf{73.92^{+}_{-}10.26}$	$\mathbf{78.45^{+}_{-}6.72}$	71.75	75.32	TF-CHI
Cricket	$88.10^{+}_{-}7.12$	$\mathbf{92.06^{+}_{-}3.29}$	**89.81**	91.89	TF-CHI

Resultados experimentales

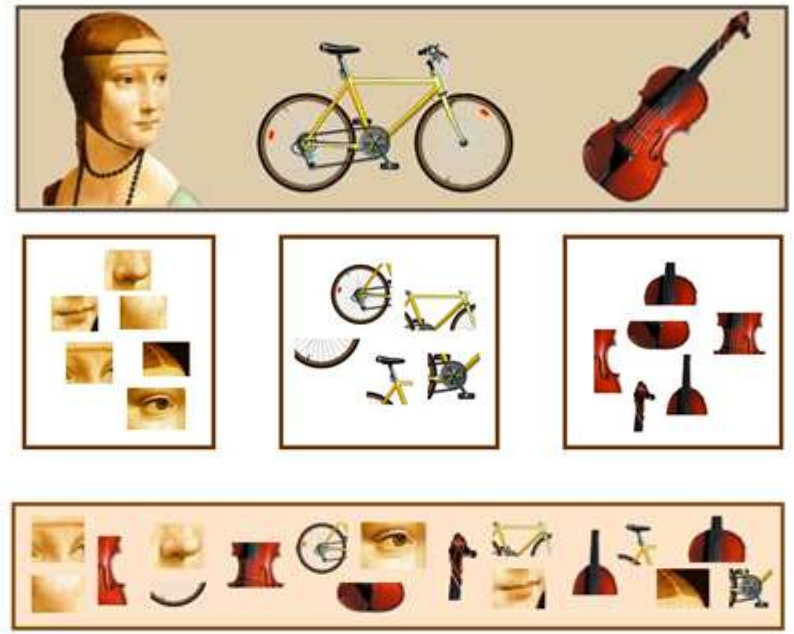
- Categorización de imágenes:

	PG- Avg.		Best baseline		
Data set	f_1	Acc.	f_1	Acc.	Baseline
Caltech-101	$\mathbf{61.91 \pm 1.41}$	$\mathbf{64.02 \pm 1.42}$	58.43	60.28	B
Caltech-tiny	$\mathbf{89.70 \pm 2.44}$	$\mathbf{91.11 \pm 2.36}$	85.65	86.67	TF

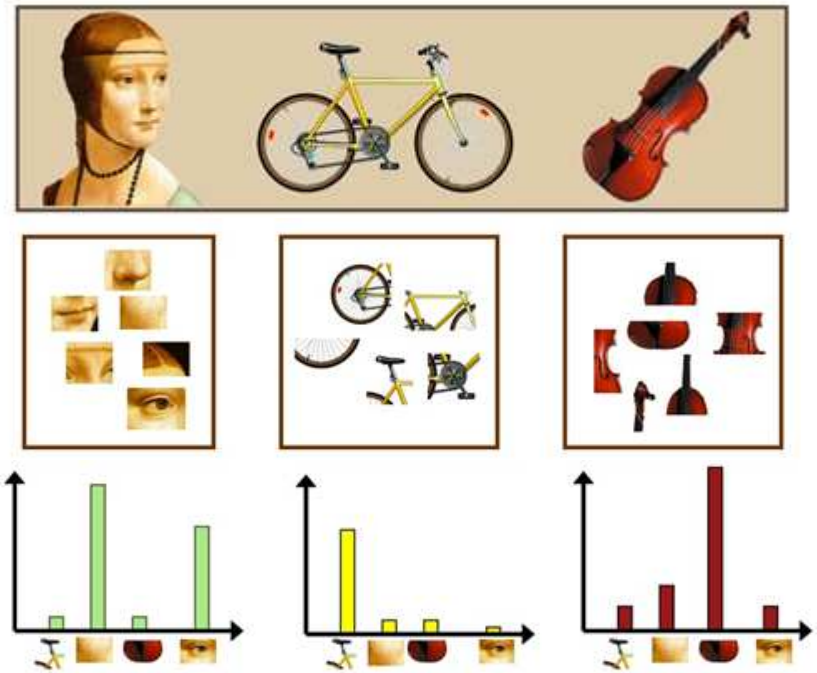
(Bolsa de palabras visuales)

- Idea: representar a imágenes como histogramas que cuentan la frecuencia con que ocurren descriptores visuales prototípicos (las palabras visuales)

Aprendizaje de vocabulario visual



Representación de imágenes

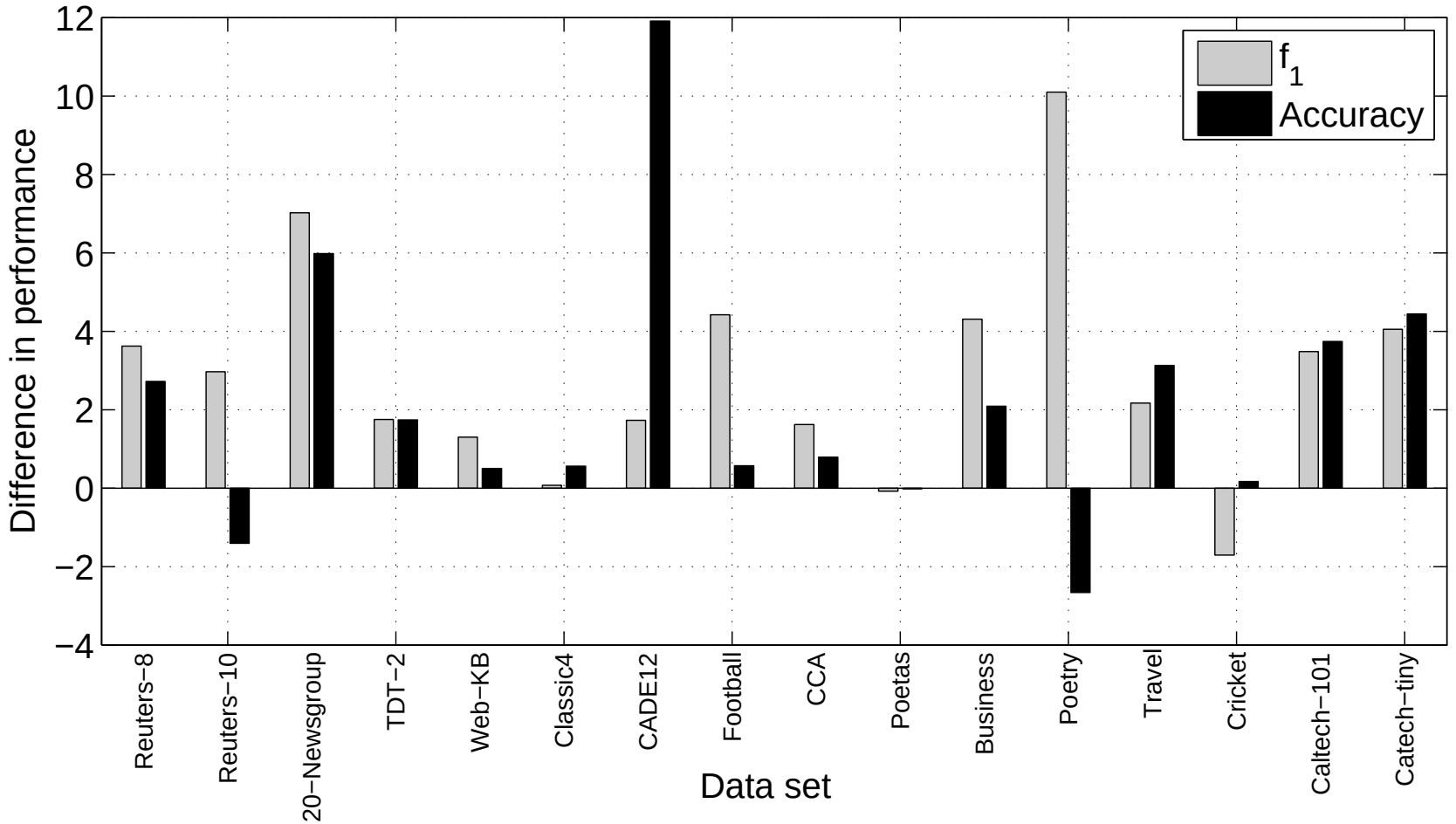


Resultados experimentales

- Categorización de imágenes:

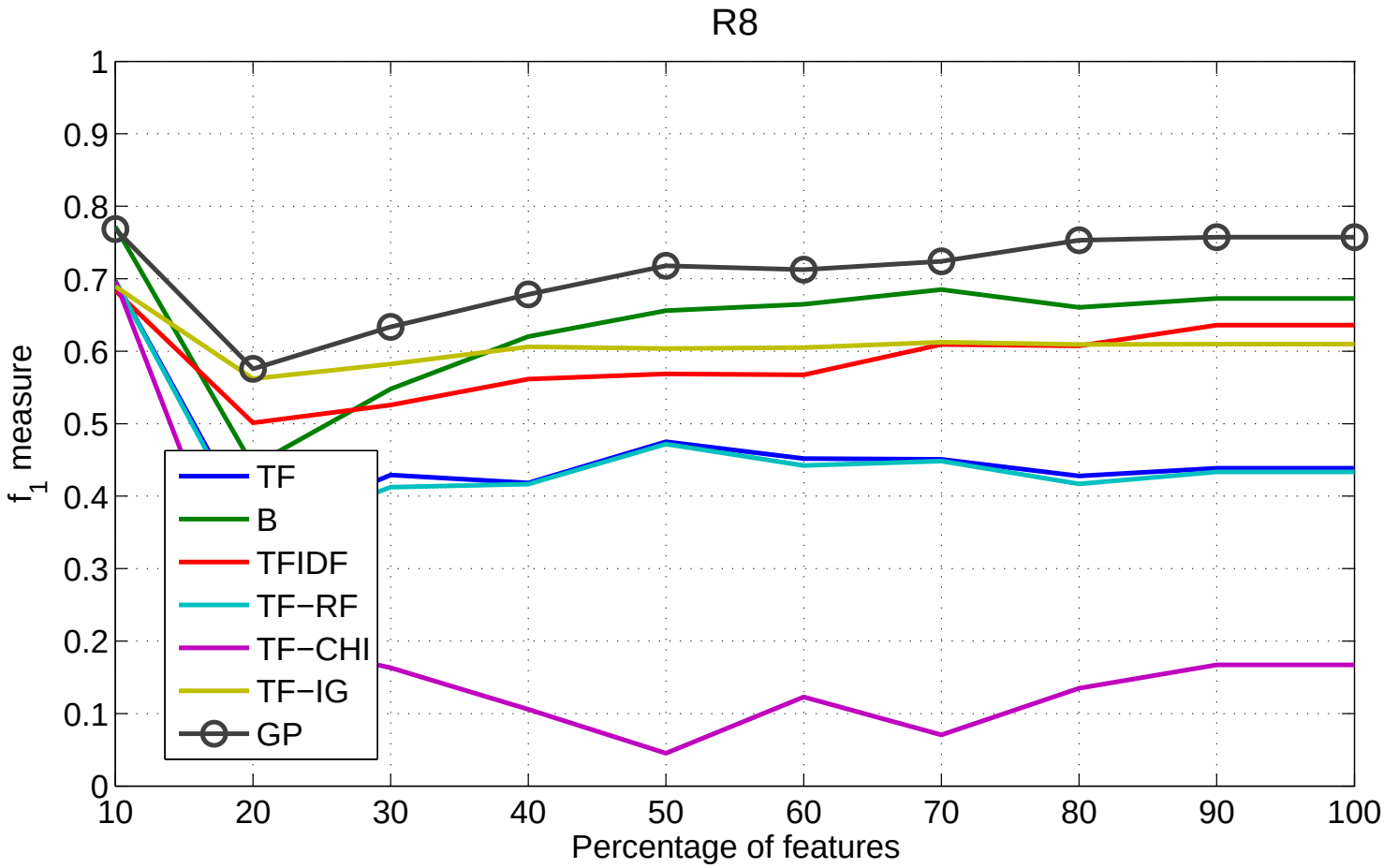
	PG- Avg.		Best baseline		
Data set	f_1	Acc.	f_1	Acc.	Baseline
Caltech-101	$\mathbf{61.91 \pm 1.41}$	$\mathbf{64.02 \pm 1.42}$	58.43	60.28	B
Caltech-tiny	$\mathbf{89.70 \pm 2.44}$	$\mathbf{91.11 \pm 2.36}$	85.65	86.67	TF

Resultados experimentales



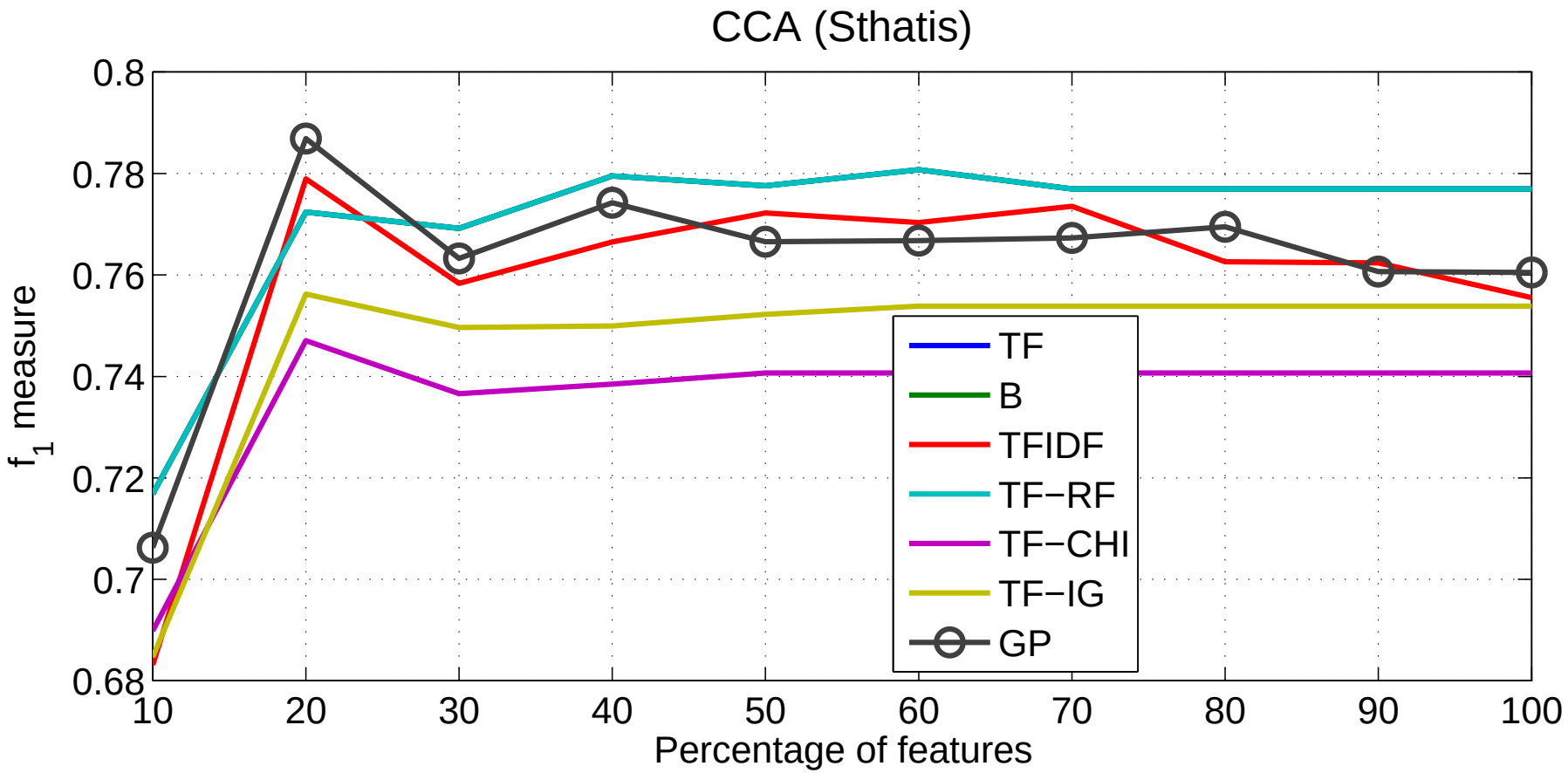
Resultados experimentales

- Variando la cantidad de términos utilizados



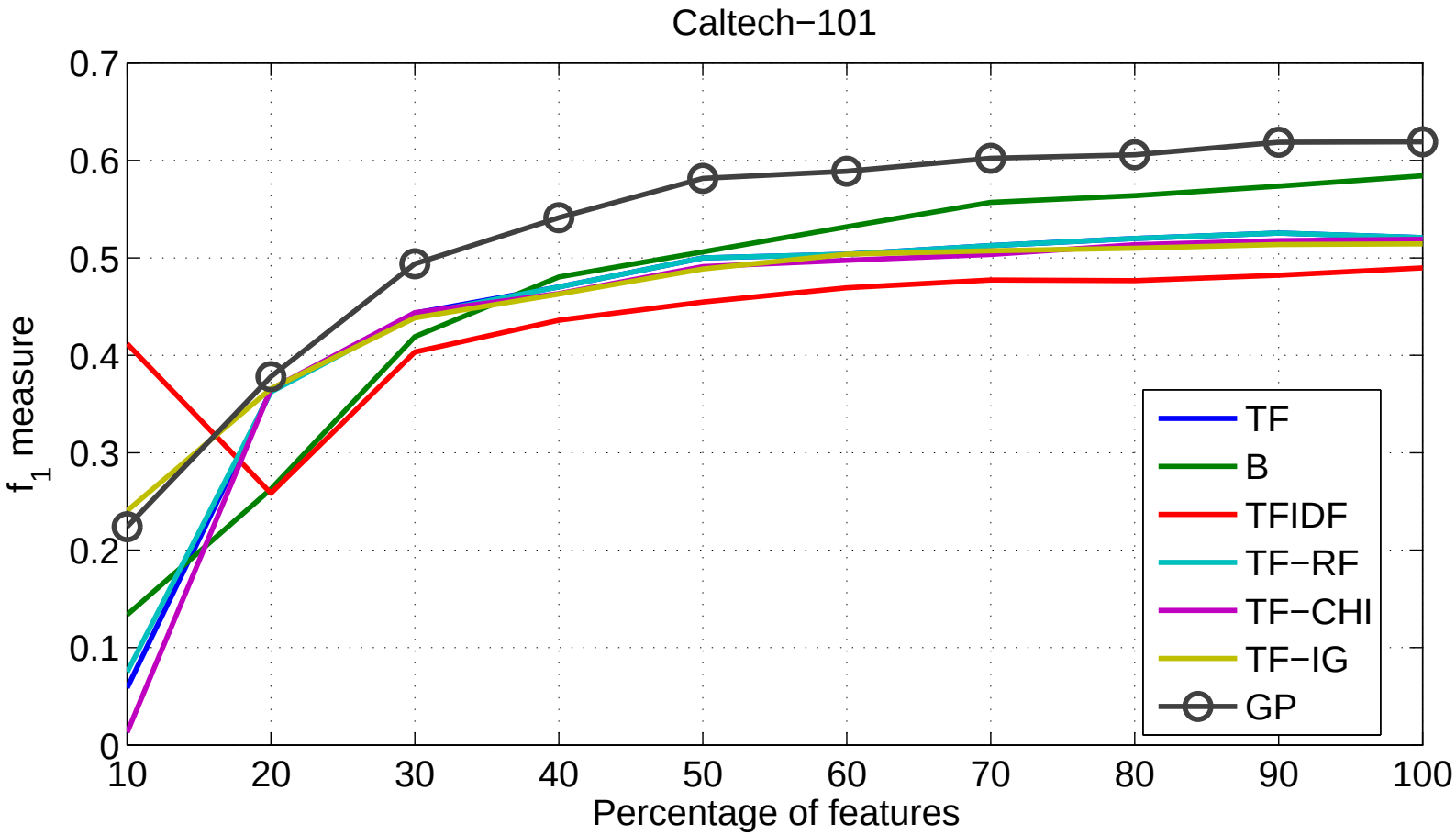
Resultados experimentales

- Variando la cantidad de términos utilizados



Resultados experimentales

- Variando la cantidad de términos utilizados



Resultados experimentales

- ¿Cómo son los pesados aprendidos?

Text categorization			
ID	**Data set**	**Learned TWS**	**Formula**
1	Reuters-8	-(sqrt(TFIDF),div(log2(sqrt(ProbR)),RF))	$\sqrt{\mathbf{W}_5} - \frac{\log\sqrt{\mathbf{W}_{19}}}{\mathbf{W}_{21}}$
2	Reuters-10	sqrt(div(pow2(sqrt(TFIDF)),div(pow2(TF-RF),pow2(TF-RF))))	$\sqrt{\frac{(\sqrt{\mathbf{W}_5})^2}{\frac{\mathbf{W}_{22}^2}{\mathbf{W}_{22}^2}}}$
3	20-Newsgroup	sqrt(sqrt(div(TF,GLOBTF)))	$\sqrt{\sqrt{\frac{\mathbf{W}_6}{\mathbf{W}_7}}}$
4	TDT-2	sqrt(×(sqrt(sqrt(TFIDF)), sqrt(×(sqrt(TFIDF),IG))))	$\sqrt{\sqrt{\sqrt{\mathbf{W}_5} \times \sqrt{\sqrt{\mathbf{W}_5} \times \mathbf{W}_4}}}$
5	WebKB	div(TF-RF,+(+(+(RF,TF-RF),FMEAS),FMEAS))	$\frac{\mathbf{W}_{22}}{(((\mathbf{W}_{21}+\mathbf{W}_{22})+\mathbf{W}_{16})+\mathbf{W}_{16})}$
6	Classic-4	×(ProbR,TFIDF)	$\mathbf{W}_5 \times \mathbf{W}_{19}$
7	CADE-12	div(TF,sqrt(log2(ACCU)))	$\frac{\mathbf{W}_6}{\sqrt{\log \mathbf{W}_{12}}}$

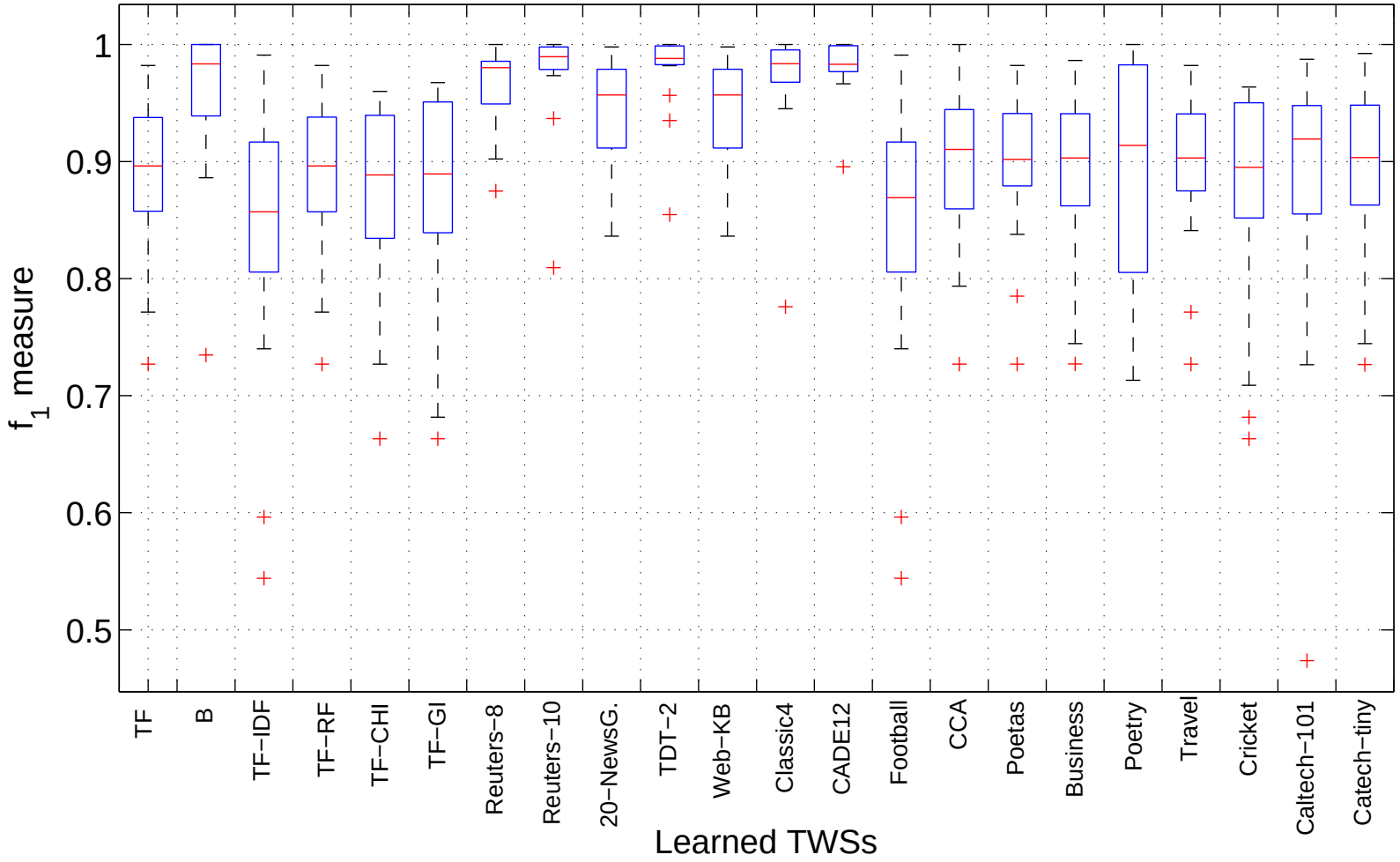
Resultados experimentales

- ¿Cómo son los pesados aprendidos?

Authorship attribution			
ID	**Data set**	**Learned TWS**	**Formula**
8	CCA	-(IG,plus(TF-RF,TFIDF))	$\mathbf{W}_4 - (\mathbf{W}_{22} + \mathbf{W}_5)$
9	Poetas	-(-(RF,TF-RF),TF-IDF)	$(\mathbf{W}_{21} - \mathbf{W}_{22}) - \mathbf{W}_5$
10	Football	div(TF-RF,pow2(ODDSR))	$\frac{\mathbf{W}_{22}}{\mathbf{W}_{17}^2}$
11	Business	minus(TF-RF,PROBR)	$\mathbf{W}_{22} - \mathbf{W}_{19}$
12	Poetry	div(TF,log2(div(TF,log(TF-RF))))	$\frac{\mathbf{W}_6}{\log \frac{\mathbf{W}_6}{\log \mathbf{W}_{22}}}$
13	Travel	+(-(TF-RF,-(-(TF-RF,-(TF-RF,POWER)),POWER)),TF)	$(\mathbf{W}_{22} - ((\mathbf{W}_{22} - (\mathbf{W}_{22} - \mathbf{W}_{18})) - \mathbf{W}_{18})) + \mathbf{W}_6$
14	Cricket	×(IG,TF-RF)	$\mathbf{W}_4 \times \mathbf{W}_{22}$
Image Classification			
ID	**Data set**	**Learned TWS**	**Formula**
15	Caltech-101	sqrt(sqrt(-(ODDSR,sqrt(sqrt(TF-RF)))))	$\sqrt{\sqrt{\mathbf{W}_{17} - \sqrt{\sqrt{\mathbf{W}_{22}}}}}$
16	Caltech-tiny	sqrt(-(TF-RF,ACBAL))	$\sqrt{\mathbf{W}_{22} - \mathbf{W}_{13}}$

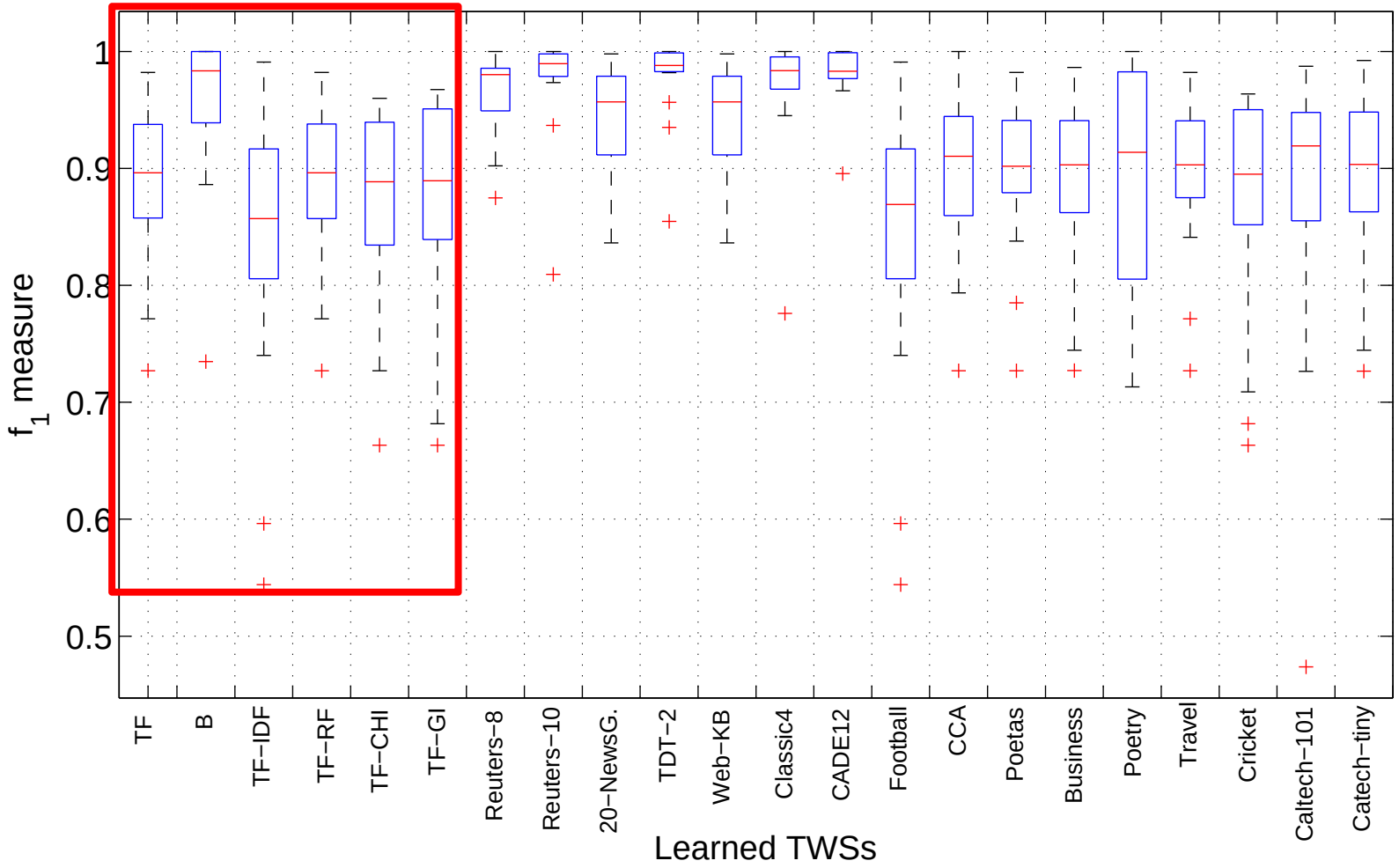
Resultados experimentales

- Generalidad de los pesados aprendidos I



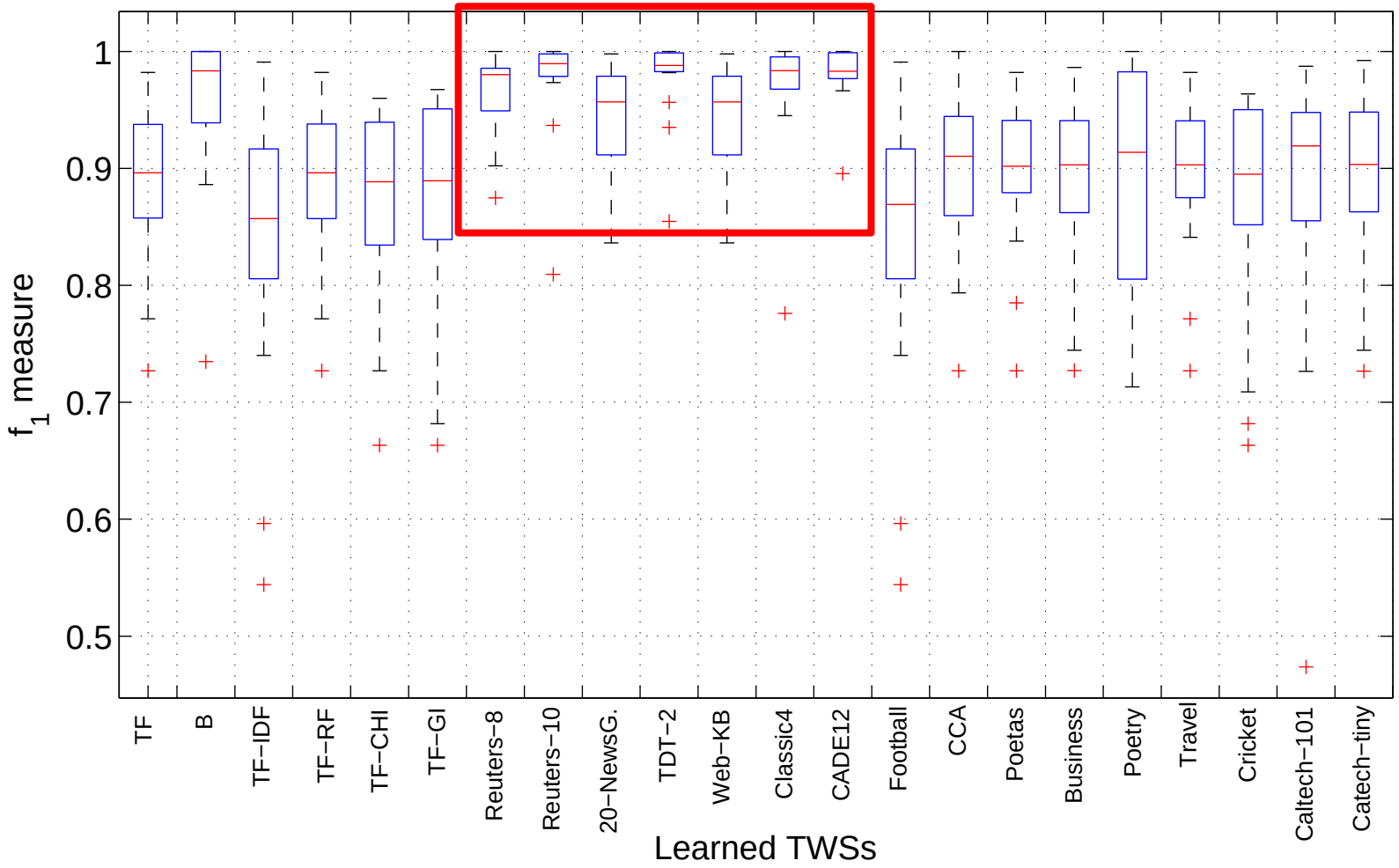
Resultados experimentales

- Generalidad de los pesados aprendidos I



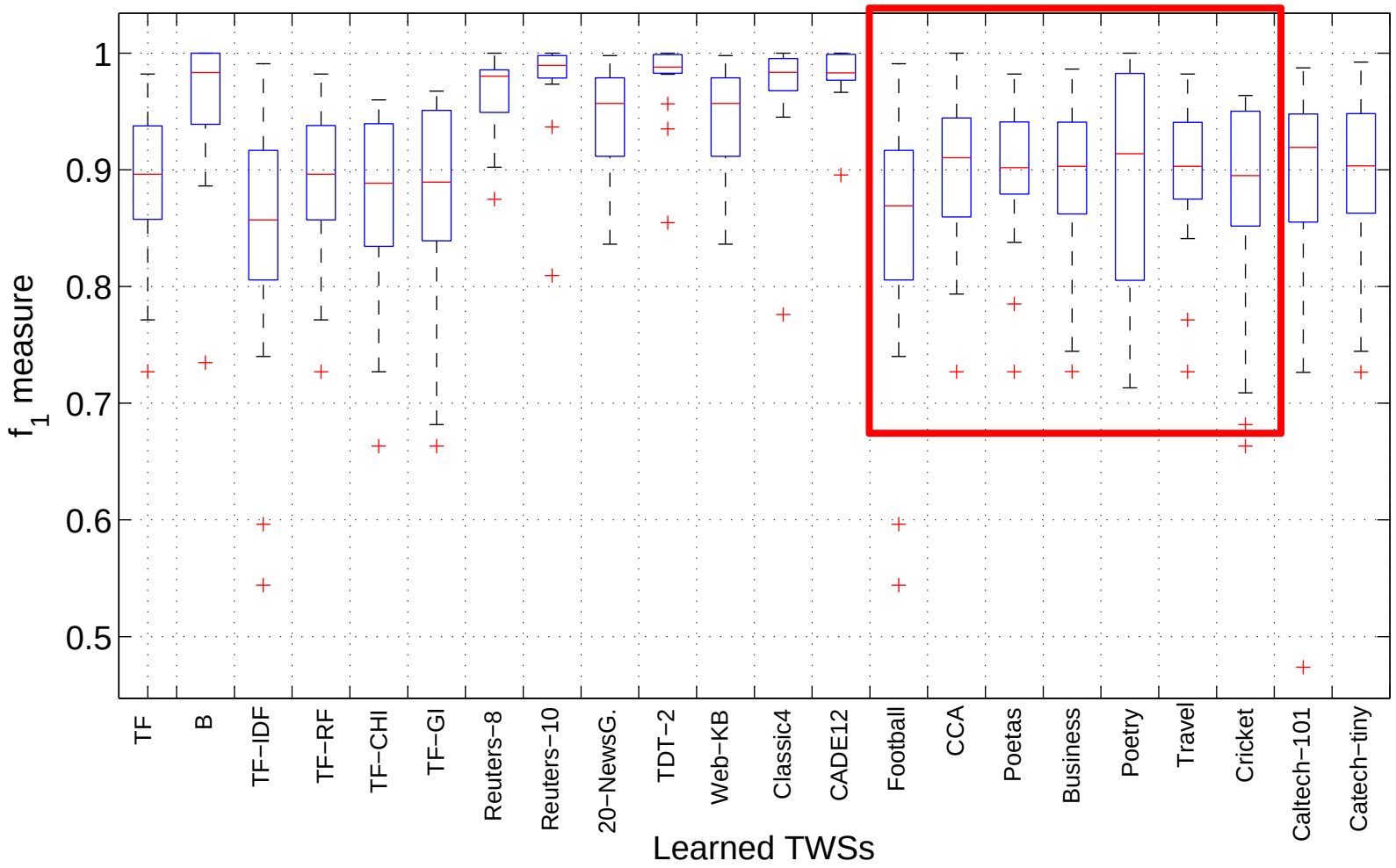
Resultados experimentales

- Generalidad de los pesados aprendidos I



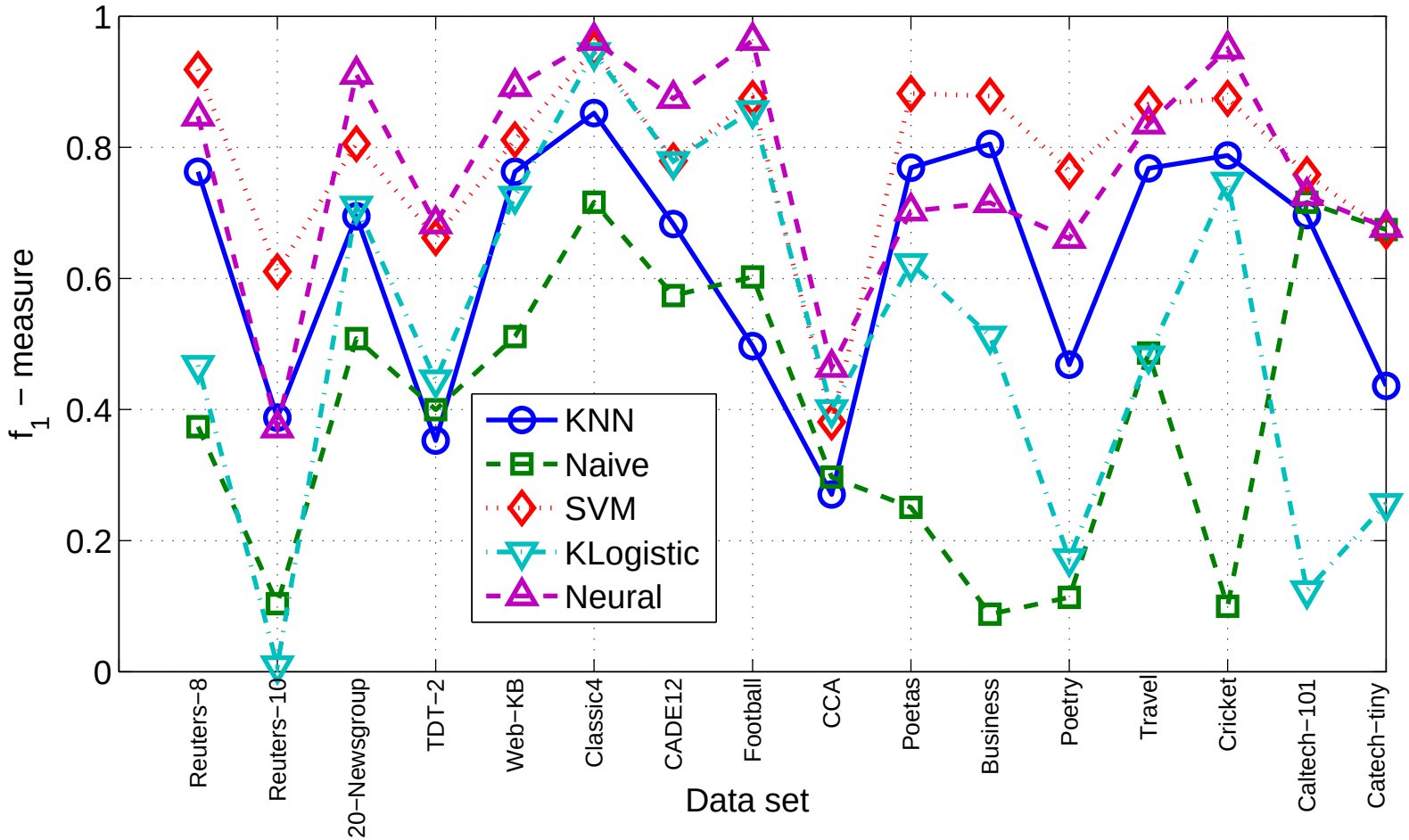
Resultados experimentales

- Generalidad de los pesados aprendidos I



Resultados experimentales

- Generalidad de los pesados aprendidos II



Conclusiones (I): generales

- La definición de esquemas de pesado tiene un impacto importante en tareas de clasificación de imágenes

- En atribución de autoría esquemas de pesado supervisado son más efectivos

Conclusiones (II): de nuestro método

- Esquemas de pesado aprendidos mejoran el desempeño de esquemas tradicionales y supervisados
- La efectividad de los esquemas de pesado no varia considerablemente, cuando se varía el tamaño del vocabulario (CT)
- En atribución de autoría es mejor idea seleccionar un tamaño de vocabulario apropiado antes de aplicar nuestro método
- Esquemas de pesado aprendidos para una modalidad pueden aplicarse a otra modalidad diferente, sin degradar el rendimiento notablemente (principalmente CT y CI)
- Los esquemas aprendidos son fáciles de analizar

Trabajo futuro

- El método es sumamente costoso (comp.): acelerar la búsqueda por medio de modelos surrogados
- Estudiar con detalle el impacto que tiene el esquema de pesado en otras tareas de visión computacional o incluso en procesamiento de voz
- Aprendizaje de representaciones con otro tipo de paradigmas

¿Preguntas?

Hugo Jair Escalante, Mauricio García-Limón, Alicia Morales, Mario Graff, Manuel Montes-y-Gómez, Eduardo Morales. **Learning term-weighting schemes for text classification.** *Knowledge based systems, 2015*

Conclusions

- GP is a competitive solution for several (open) problems in machine learning and related fields

- Computationally expensive, and no guarantees on finding the *optimum model,* Yet:
  - In these cases it is unkown whether there is an optimum solution
  - It is infeasible to use standard/exact optimization techniques

References

- Genetic Programming of Prototypes for Pattern Classification. Hugo Jair Escalante, Karlo Mendoza, Mario Graff. In J.M. Sanches, L. Micó, and J.S. Cardoso (Eds.): IbPRIA 2013, IbPRIA 2013: 6th Iberian Conference on Pattern Recognition and Image Analysis, Madeira, Portugal. June 5-7, 2013, LNCS 7887, pp. 100–107, 2013
- Simultaneous Generation of Prototypes and Features through Genetic Programming. Mauricio García-Limón, Hugo Jair Escalante, Eduardo Morales, Alicia Morales. GECCO '14 Proceedings of the 2014 conference on Genetic and evolutionary computation, pp. 517-524, (Full paper, Oral presentation), Vancouver, Canada, July, 12-17, 2014.
- Genetic programming of text representations. Mauricio García-Limón, Hugo Jair Escalante, Manuel Montes-y-Gómez, Alicia Morales, Eduardo Morales. GECCO Comp'14 Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion, pp. 1459-1460, (Late-breaking abstract, Oral presentation), Vancouver, Canada, July, 12-17, 2014.
- An Evolutionary Multi-Objective Approach for Prototype Generation. Alejandro Rosales, Hugo Jair Escalante, Jesus A. Gonzalez, Carlos A. Coello, Carlos A. Reyes. CEC'14: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1100—1007, (Full paper, Oral presentation), Beijing, China, July 6-11, 2014.
- Object Recognition with Naive Bayes-KNN via Prototype Generation. Hugo Jair Escalante, Mauricio Sotomayor, Manuel Montes, A. Pastor López-Monroy. In J.F. Martínez Trinidad, J. A. Carrasco-Ochoa, J. A. Olvera-López, J. Salas Rodríguez, C. Y. Suen (Eds.): Pattern Recognition - 6th Mexican Conference, MCPR 2014, Cancun, Mexico, June 25-28, 2014. Proceedings. Springer 2014 Lecture Notes in Computer Science ISBN 978-3-319-07490-0
- Evolutionary Multi-Objective Approach for Prototype Generation and Feature Selection. Alejandro Rosales-Pérez, Jesús A. González, Carlos A. Coello Coello, Carlos A. Reyes García, Hugo Jair Escalante. CIARP 14: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, LNCS, Vol. 8827, pp. 424-431, 2014,
- Towards Simultaneous Prototype and Feature Generation. Mauricio Alfonso García Limón, Hugo Jair Escalante Balderas and Eduardo Morales Manzanares. Proceedings of the XVI IEEE Autumn Meeting of Power, Electronics and Computer Science ROPEC 2014 INTERNACIONAL, pp. 393—398, 2014. (slides)
- Learning to Assemble Classifiers via Genetic Programming. Niusvel Acosta-Mendoza, Alicia Morales-Reyes, Hugo Jair Escalante, Andres Gago-Alonso. International Journal of Pattern Recognition and Artificial Intelligence, Accepted, 2014 (Impact factor: 0.558 -JCR)

Human action recognition

Object recognition

Content-based image retrieval

APPLICATIONS OF GP 2: COMPUTER VISION

Human action recognition

Object recognition

Content-based image retrieval

APPLICATIONS OF GP 2: COMPUTER VISION

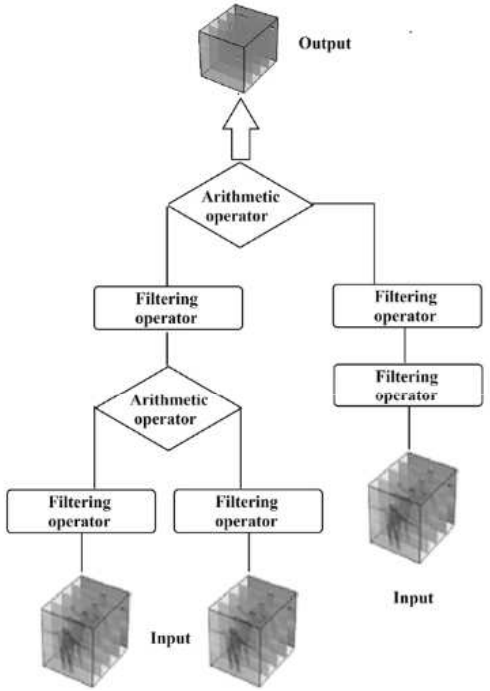
Human action recognition

- Problem: to learn a spatio-temporal descriptor to represent videos for human action recognition
- An STD must (effectively) represent the content of a video
- Traditional STDs are handcrafted by processing the 3D signal

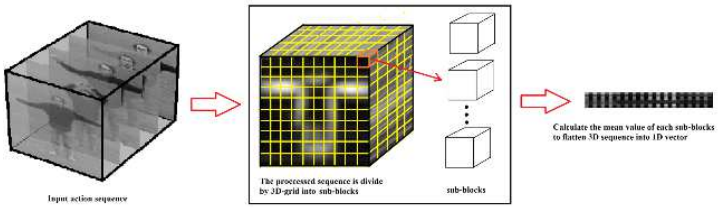
Li Liu, Ling Shao and Peter Rockett. **Genetic Programming-Evolved Spatio-Temporal Descriptor for Human Action Recognition.** In Proceedings British Machine Vision Conference 2012. Pages 18.1– 18, 2012.

Human action recognition with GP

- Proposal: to learn a STD from data via GP
  - Standard GP (GPLab)
  - Fitness function: cross-validation SVM error
  - Crossover (90%) & mutation (10%)



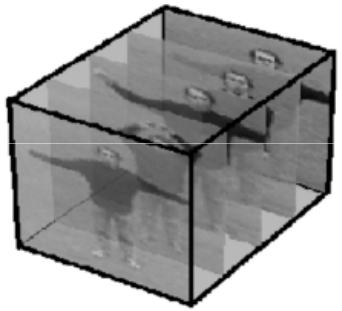
- Preprocessing: Normalization of all videos including an action



Li Liu, Ling Shao and Peter Rockett. **Genetic Programming-Evolved Spatio-Temporal Descriptor for Human Action Recognition.** In Proceedings British Machine Vision Conference 2012. Pages 18.1– 18, 2012.

Human action recognition with GP

- Terminals: Cubes of dim. 100x100x70



- Function set: filters and arithmetic operators

Operator name	Inputs	Function description	Operator type
GauPy1	1	The first level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on input sequences	Filter
GauPy2	1	The second level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on sequences obtained from first level of a Gaussian pyramid	Filter
GauPy3	1	The third level of a Gaussian pyramid which applies a 3D Gaussian filter with $\sigma = 2$ on sequences obtained from second level of a Gaussian pyramid	Filter
LapPy1	1	The first level of a Laplacian pyramid which applies a subtraction between GauPy1 and GauPy2	Filter
LapPy2	1	The second level of a Laplacian pyramid which applies a subtraction between GauPy2 and GauPy3	Filter
Wavelet1	1	The first leveHenryl of a wavelet pyramid which appliesa CDF '9/7' wavelet filter on input sequences	Filter
Wavelet2	1	The second level of a wavelet pyramid which appliesa CDF '9/7' wavelet filter once again on sequences obtained from first level of wavelet pyramid	Filter
Dof	1	Subtraction between the adjacent frames of input sequences	Filter
absDof	1	Subtraction between the adjacent frames of input sequences and then taking the absolute values of the processed sequences	Filter
Med	1	Apply median filter with filtering window size 5×5 on the input sequences	Filter
Mean	1	Apply mean filter with filtering window size 5×5 on the input sequences	Filter
Maxfilter	1	Use 3D maxpooling technique with pooling window size $4 \times 4 \times 4$ on the input sequences	Filter
Add	2	Add the input sequences	Arithmetic
Sub	2	Subtract the input sequences	Arithmetic
Mult	2	Multiply the input sequences	Arithmetic
absSub	2	Absolute subtraction of the input sequences	Arithmetic

Li Liu, Ling Shao and Peter Rockett. **Genetic Programming-Evolved Spatio-Temporal Descriptor for Human Action Recognition.** In Proceedings British Machine Vision Conference 2012. Pages 18.1– 18, 2012.

Human action recognition with GP

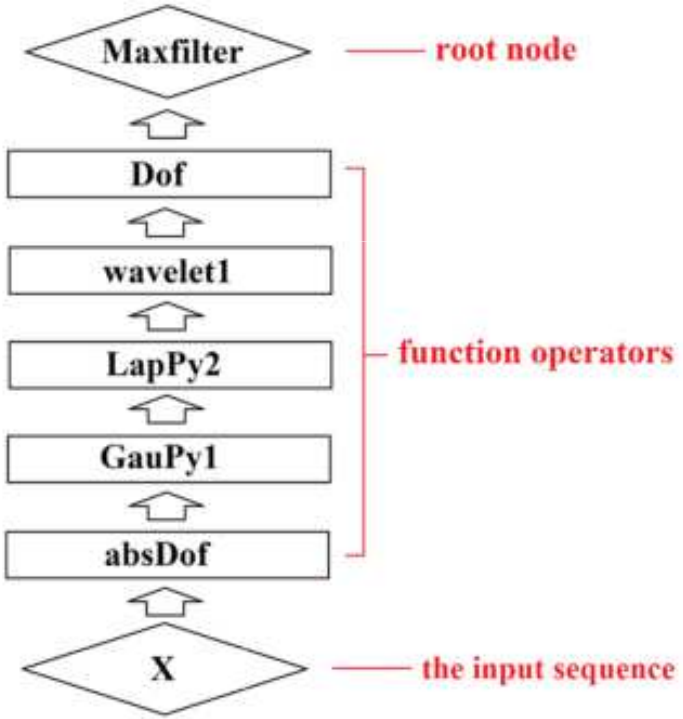
- Results

Table 2: Action recognition accuracies (%) on the mixed KTH-Weizmann dataset, independent KTH and independent Weizmann datasets, respectively, for different methods

Methods \ Database	KTH-Weizmann	KTH	Weizmann
GP-based descriptor	96.9	93.8	100
Niebles *et al.* [23]	-	83.3	72.8
Jhuang *et al.* [9]	-	91.7	98.8
Fathi and Mori [5]	-	90.5	100
Ji *et al.* [10]	-	90.2	-
Schindler and van Gool [26]	-	92.7	-
Liu and Shah [18]	-	94.2	-

Table 3: Comparison of action recognition accuracies (%) on the IXMAS dataset for different methods

Methods \ Actions	Cam 1	Cam 2	Cam 3	Cam 4	Cam 5	Cam 1-5 fusion
GP-based descriptor	86.5	89.3	88.1	84.7	78.3	93.6
Varma and Babu [32]	76.4	74.5	73.6	71.8	60.4	81.3
Liu and Shah [18]	76.7	73.3	72.1	73.1	-	82.8
Wu *et al.* [35]	81.9	80.1	77.1	77.6	73.4	88.2
Weinland *et al.* [33]	-	-	-	-	-	93.3



Best descriptor found after weeks of processing

Li Liu, Ling Shao and Peter Rockett. **Genetic Programming-Evolved Spatio-Temporal Descriptor for Human Action Recognition.** In Proceedings British Machine Vision Conference 2012. Pages 18.1– 18, 2012.

Human action recognition with GP

- A very effective descriptor was obtained *automatically*
- Results comparable to state of the art (compared to works using handcrafted STDs)
- Time consuming process (is this an issue nowadays?)
- Main drawback: interpretation of operators?, generality?

Li Liu, Ling Shao and Peter Rockett. **Genetic Programming-Evolved Spatio-Temporal Descriptor for Human Action Recognition.** In Proceedings British Machine Vision Conference 2012. Pages 18.1– 18, 2012.

Human action recognition

Object recognition

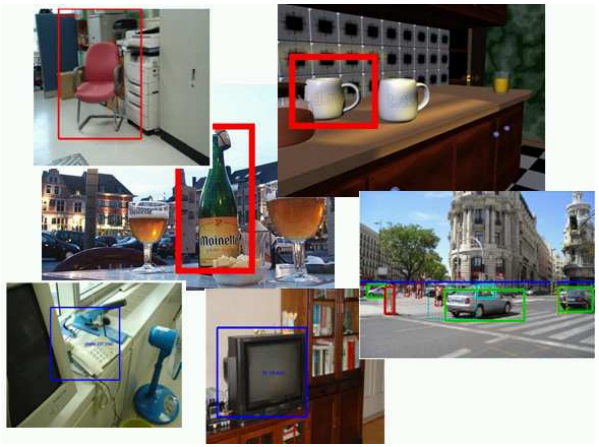
Content-based image retrieval

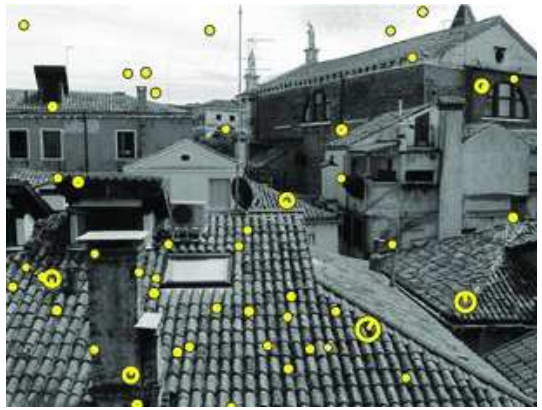
APPLICATIONS OF GP 2: COMPUTER VISION

Object recognition

- **Problem:** to learn a new visual descriptor to be used for object recognition and related tasks

- Traditional visual descriptors (e.g., SIFT) are handcrafted:
  - Feature extraction of local features
  - Image description of local patches centered around keypoints
  - Image matching/OR using local information





Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

Object recognition with GP

- Proposal: to learn a visual descriptor from data
  - Standard GP (GPLab)
  - Fitness function: F-measure (on the # of matches)

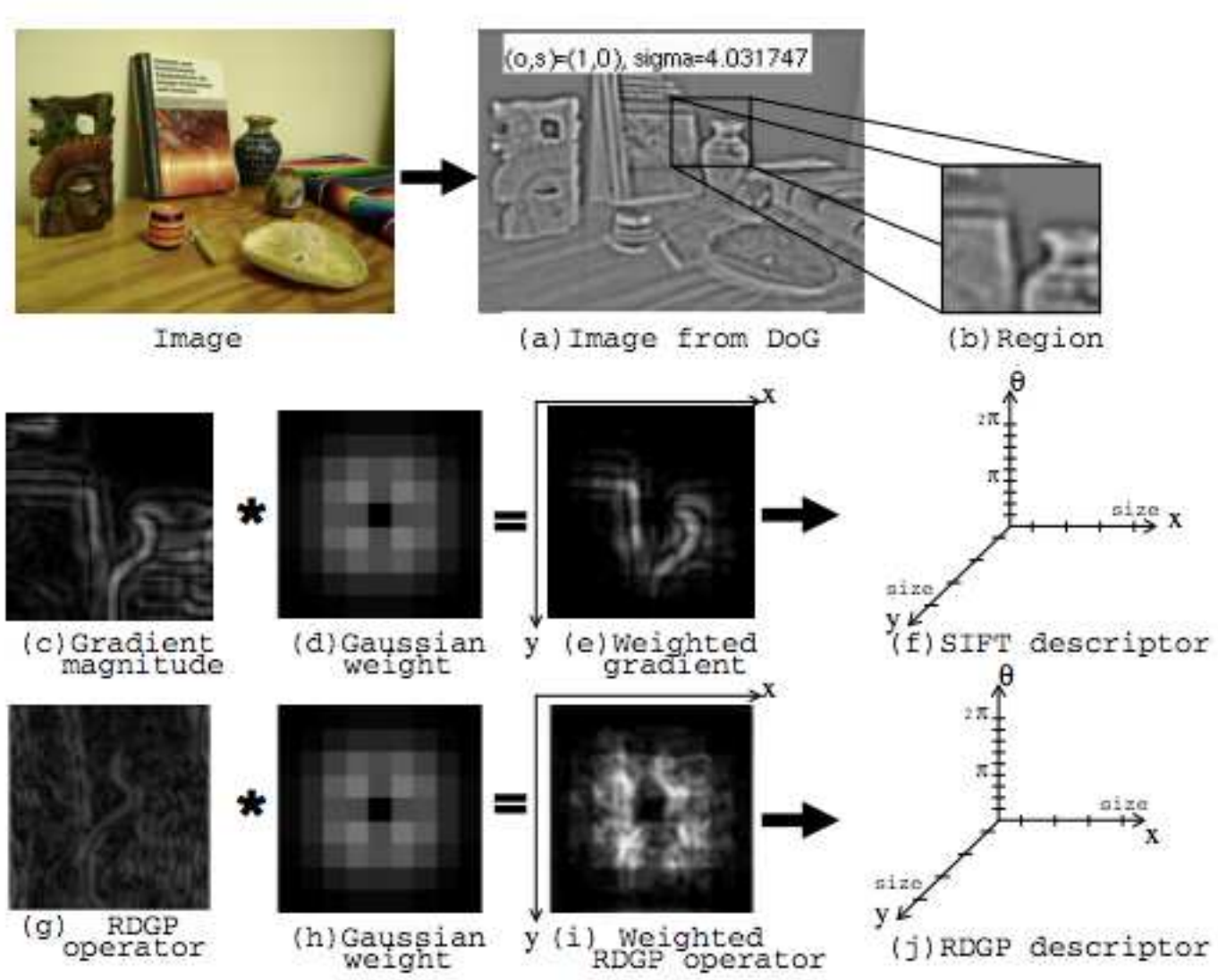
$$Q = \text{argmax}\left\{F(P^x, R^x) = \sum_{i=1}^{n} \frac{2 \cdot (p_i \cdot r_i)}{p_i + r_i}\right\}$$

  - Crossover (90%) & mutation (10%)
  - 50 iterations, population of 50 individuals (24hrs running an experiment)

$$FuncSet = \left\{+, |+|, -, |-|, *, \div, \sqrt{I_t}, \frac{I_t}{2}, log_2(I_t), D_x G_\sigma, D_y G_\sigma, G_\sigma\right\}$$

$$TermSet = \{I, D_x, D_{xx}, D_{yy}, D_{xy}, D_y\}$$

Object recognition with GP



Object recognition with GP

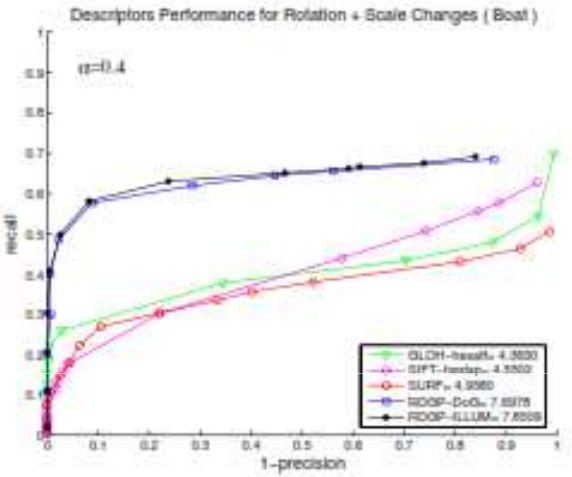
DATASET

Boat image pair

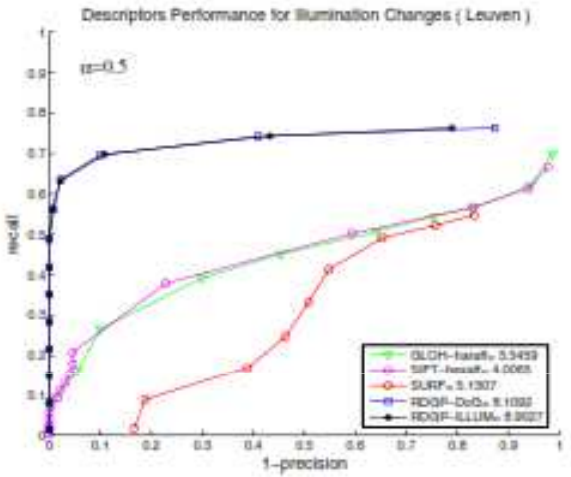
Leuven image pair

Bikes image pair

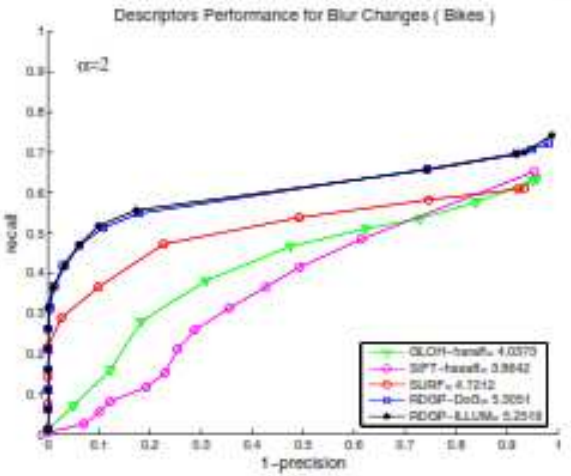
UBC image pair



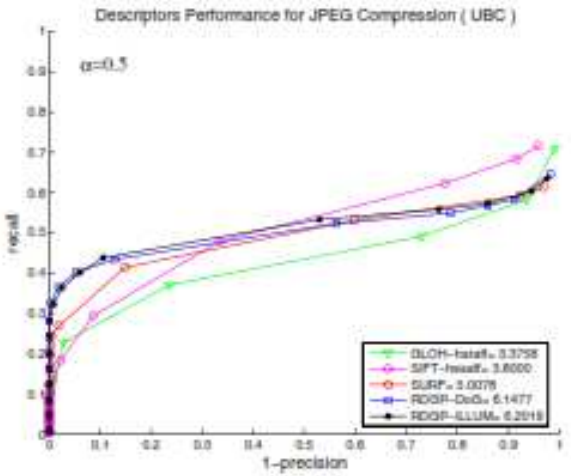
a) Boat image pair



b) Leuven image pair

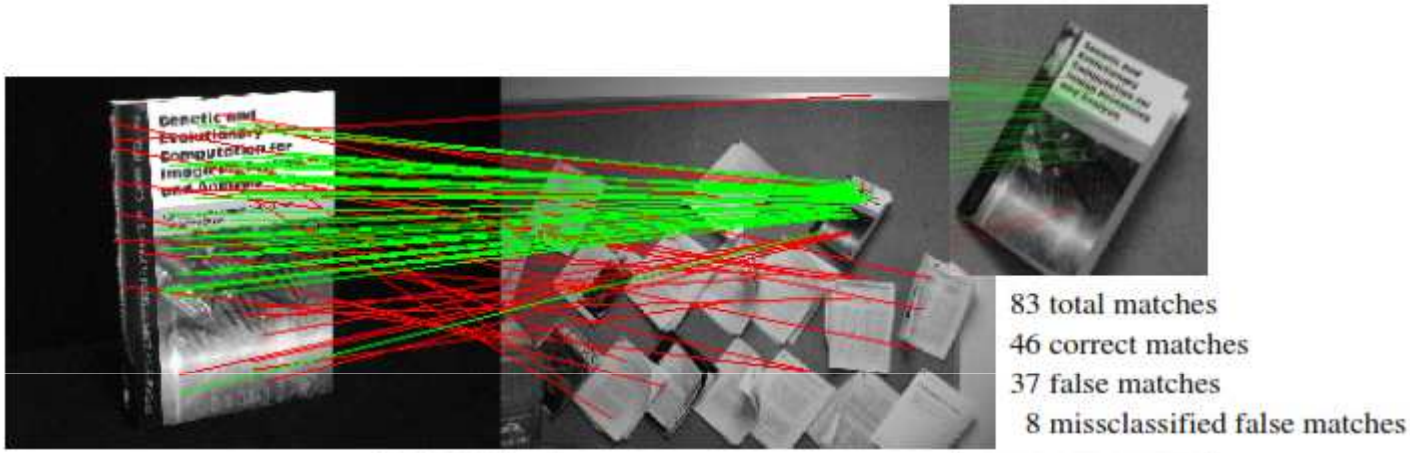


c) Bikes image pair

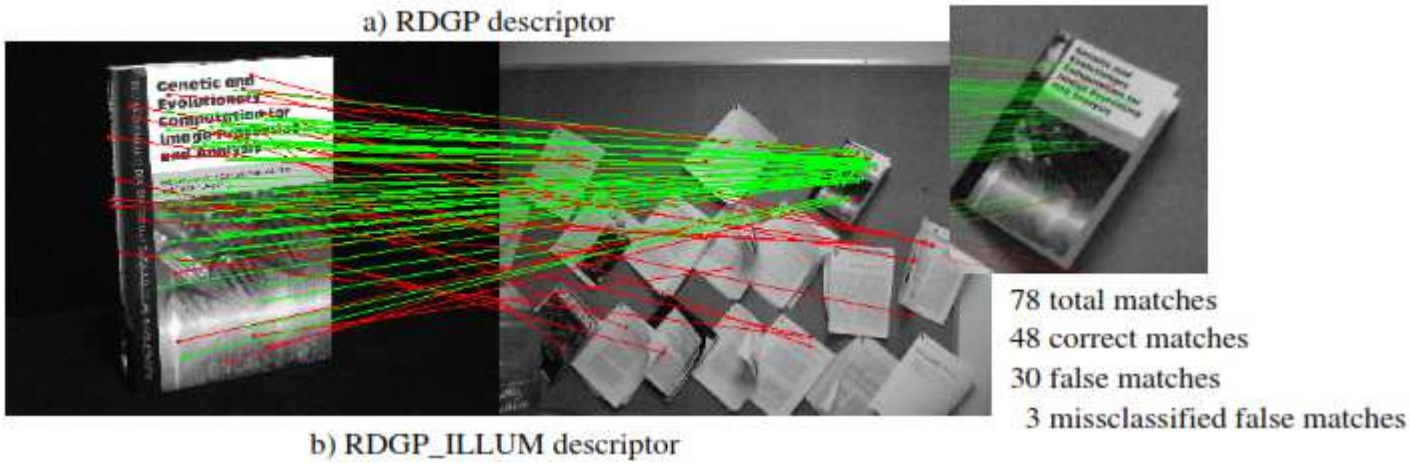


d) UBC image pair

Object recognition with GP



a) RDGP descriptor



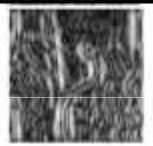
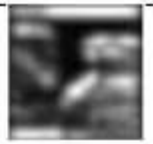
b) RDGP_ILLUM descriptor

Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

Object recognition with GP

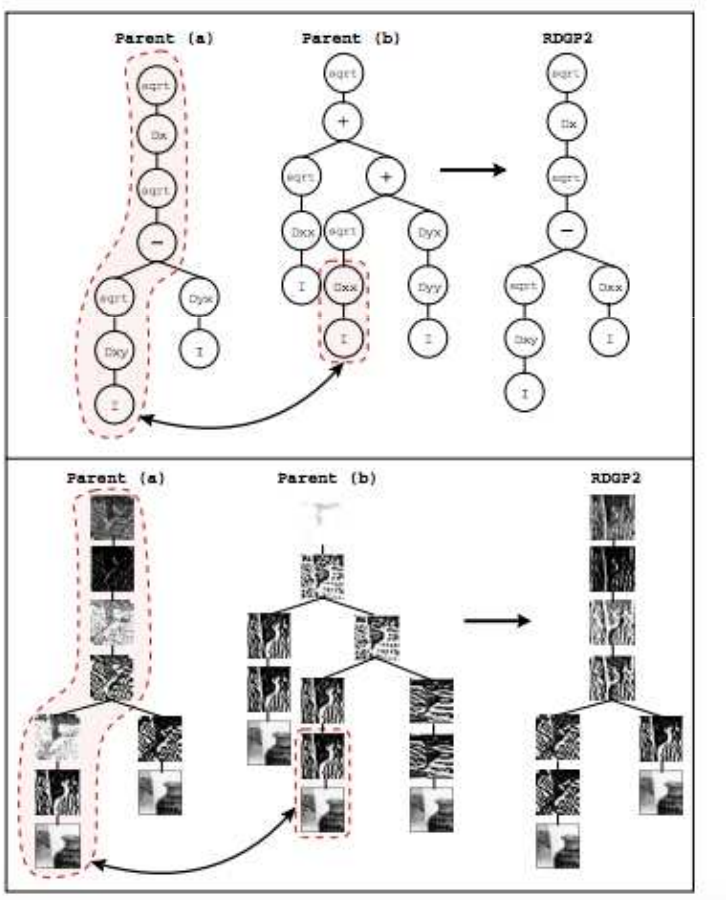
 Image region without any operation

 Image region after SIFT's weighted gradient

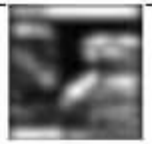
Descriptor	Fitness	Individual's Expression	Mathematical Expression	Image Region
$RDGP_1$	7.4158	$sqrt(sqrt(D_x(sqrt(D_x(sqrt(D_{xy}(image)))))))$	$\sqrt{\sqrt{D_x\left(\sqrt{D_x\left(\sqrt{D_{xy}(I)}\right)}\right)}}$	
$RDGP_2$	7.4859	$sqrt(D_x(sqrt(substract(sqrt(D_{xy}(image)), D_{xx}(image)))))$	$\sqrt{D_x\left(\sqrt{\sqrt{D_{xy}(I)}-D_{xx}(I)}\right)}$	
$RDGP_4$	7.3928	$Gauss2(absdif(Gauss2(absdif(absdif(D_y(image),D_x(D_x(image))),D_y(Logarithm(D_x(D_x(image))))),Half(D_x(D_y(image)))))$	$G_{\sigma=2}\lvert G_{\sigma=2}(\lvert\lvert D_y(I) - D_{xx}(I)\rvert - D_y(\log(D_{xx}(I)))\rvert) - \frac{D_{xx}(I)}{2}\rvert$	
$RDGP_5$	7.4053	$Gauss1(sqrt(Gauss2(sqrt(sqrt(substract(sqrt(Gauss1(D_y(image))),divide(D_{xx}(image),absadd(D_x(image),D_y(image))))))))$	$G_{\sigma=1}\sqrt{G_{\sigma=2}\sqrt{\sqrt{G_{\sigma=1}(D_y(I))}-\frac{D_{xx}(I)}{\lvert D_x(I)+D_y(I)\rvert}}}$	

Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

Crossover

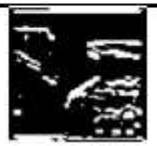
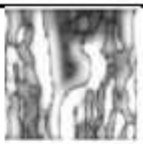


Object recognition with GP

$RDGP_5$	7.4053	$Gauss1(sqrt(Gauss2(sqrt(sqrt($ $substract(sqrt(Gauss1(D_y(image))),$ $divide(D_{xx}(image),absadd(D_x($ $image),D_y(image)))))))$	$G_{\sigma=1}\sqrt{G_{\sigma=2}\sqrt{\sqrt{G_{\sigma=1}(D_y(I))}-\frac{D_{xx}(I)}{\lvert D_x(I)+D_y(I)\rvert}}}$	
$RDGP_{11}$	7.3736	$Half(G_2(G_2(sqrt(D_{xx}($ $Log(D_{xy}(image)))))))))$	$\frac{G_{\sigma=2}(G_{\sigma=2}\sqrt{(D_{xx}(log_2(D_{xy}(I))))}}{2}$	
$HILL_1$	6.2854	$add(D_{xx}(image), sqrt($ $sqrt(D_x(image))))$	$D_{xx}(I)+\sqrt{\sqrt{Dx(I)}}$	
$HILL_2$	6.1779	$add(D_y(image), sqrt($ $Gauss1(sqrt(Gauss2(D_{xx}($ $image))))))$	$D_{yy}(I)+\sqrt{G_{\sigma=1}\sqrt{G_{\sigma=2}(D_{xx}(I))}}$	
$HILL_3$	6.1389	$absdif(D_y(image), subtract($ $D_{xy}(image), Log(Half($ $Gauss1(D_{xx}(image)))))))$	$\left\lvert D_y(I)-\left[D_{xy}(I)-log\left(\frac{G_{\sigma=1}(D_{xx}(I))}{2}\right)\right]\right\rvert$	

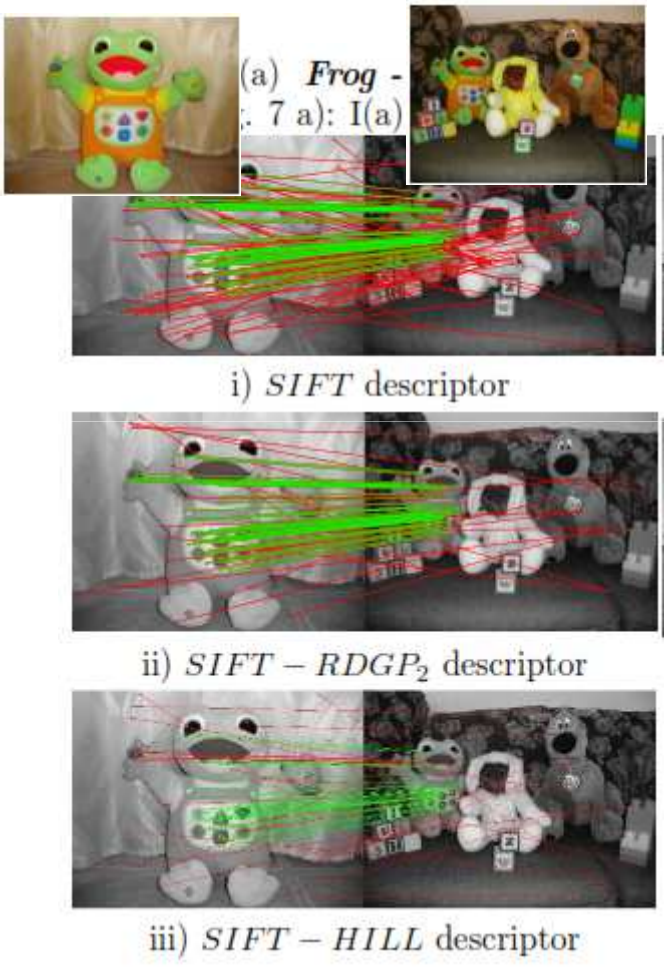
Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

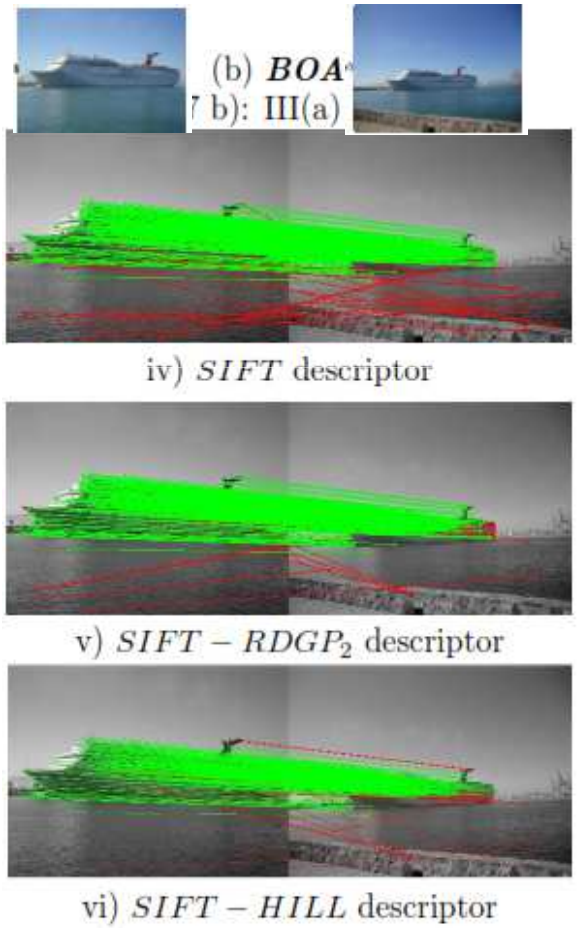
Object recognition with GP

$HILL_2$	6.1779	$add(D_y(image), sqrt($ $Gauss1(sqrt(Gauss2(D_{xx}($ $image))))))$	$D_{yy}(I) + \sqrt{G_{\sigma=1}\sqrt{G_{\sigma=2}(D_{xx}(I))}}$	
$HILL_3$	6.1389	$absdif(D_y(image), subtract($ $D_{xy}(image), Log(Half($ $Gauss1(D_{xx}(image))))))$	$\left\|D_y(I) - \left[D_{xy}(I) - log\left(\frac{G_{\sigma=1}(D_{xx}(I))}{2}\right)\right]\right\|$	
$HILL_4$	6.1345	$sqrt(Half(sqrt(Gauss2($ $D_{xx}(image)))))$	$\sqrt{\frac{\sqrt{G_{\sigma=2}(D_{xx}(I))}}{2}}$	
$HILL_5$	6.1245	$absadd(sqrt(Log(subtract($ $D_{xx}(image), Half(D_y($ $image)))))), D_y(image))$	$\left\|\left[\sqrt{log(D_{xx}(I) - \frac{D_y(I)}{2}}\right] + D_y(I)\right\|$	

Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

Object recognition with GP





Cynthia B. Perez and Gustavo Olague. **Evolving Local Descriptor Operators through Genetic Programming.** In M. Giacobini et al. (Eds.): EvoWorkshops 2009, LNCS 5484, pp. 414–419, 2009.

Object recognition with GP

- GP an effective approach to generate useful descriptors
- No handcraft work is required to obtain competitive descriptors
- Main drawback: interpretation of operators?, generality?

Human action recognition

Object recognition

Content-based image retrieval

APPLICATIONS OF GP 2: COMPUTER VISION

Content-based image retrieval

- Problem: to learn how to combine similarity values obtained from multiple modalities

- There is no single set of features that effectively represent heterogeneous images

- Information fusion (single-modality) effective method to improve retrieval performance

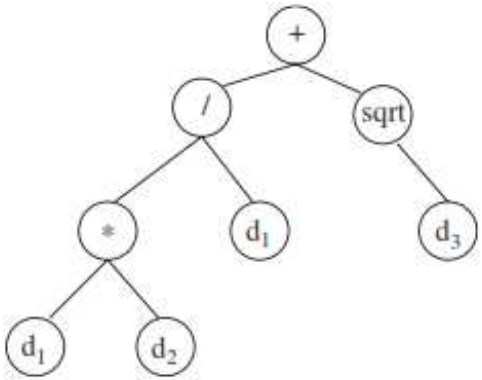
Ricardo da S. Torres, Alexandre X. Falcão, Marcos A. Gonçalves, João P. Pap, Weiguo Fan, Edward A. Foxc, Baoping Zhang, **A genetic programming framework for content-based image retrieval**, Pattern recognition Vol. 42, 283—292, 2009

Content-based image retrieval

- Proposal: Using GP to learn how to combine similarity functions
- Standard GP, using basic arithmetic operations as function set

Terminals

Descriptor	Distance function
BAS40 [4]	OCS [7]
BAS60 [4]	OCS [7]
MS fractal dimension [2]	Euclidean distance
Fourier descriptors [20]	Euclidean distance
Moment invariants [20]	Euclidean distance

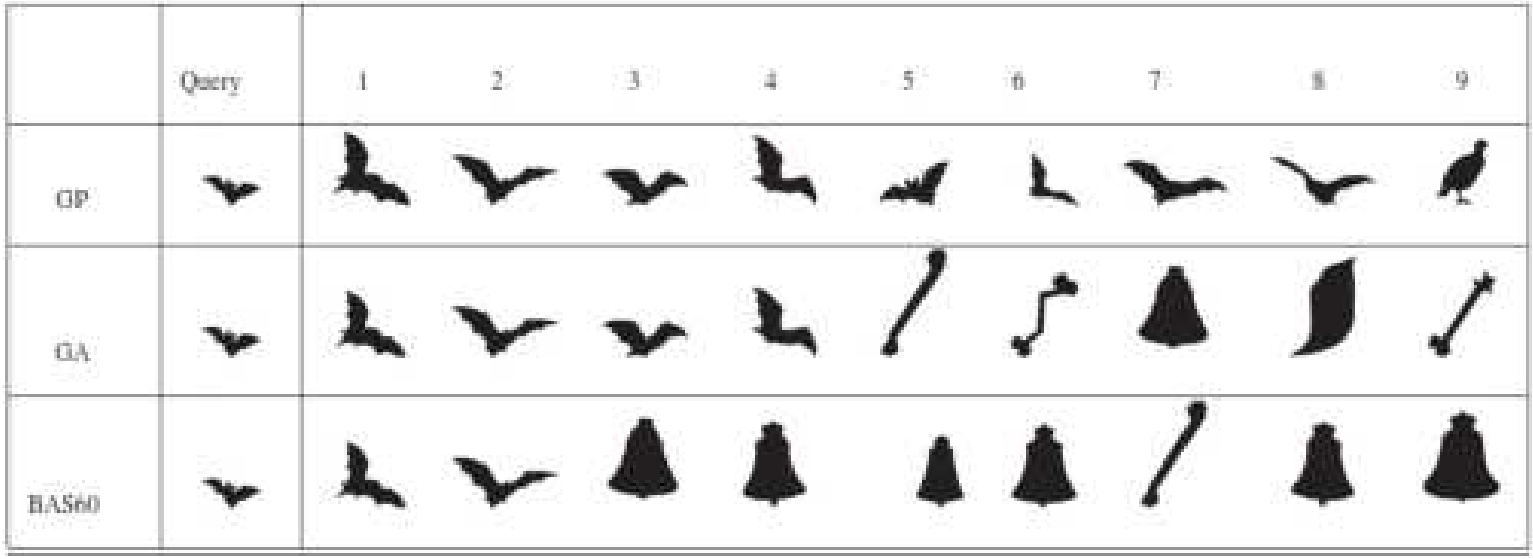


Ricardo da S. Torres, Alexandre X. Falcão, Marcos A. Gonçalves, João P. Pap, Weiguo Fan, Edward A. Foxc, Baoping Zhang, **A genetic programming framework for content-based image retrieval**, Pattern recognition Vol. 42, 283—292, 2009

Content-based image retrieval

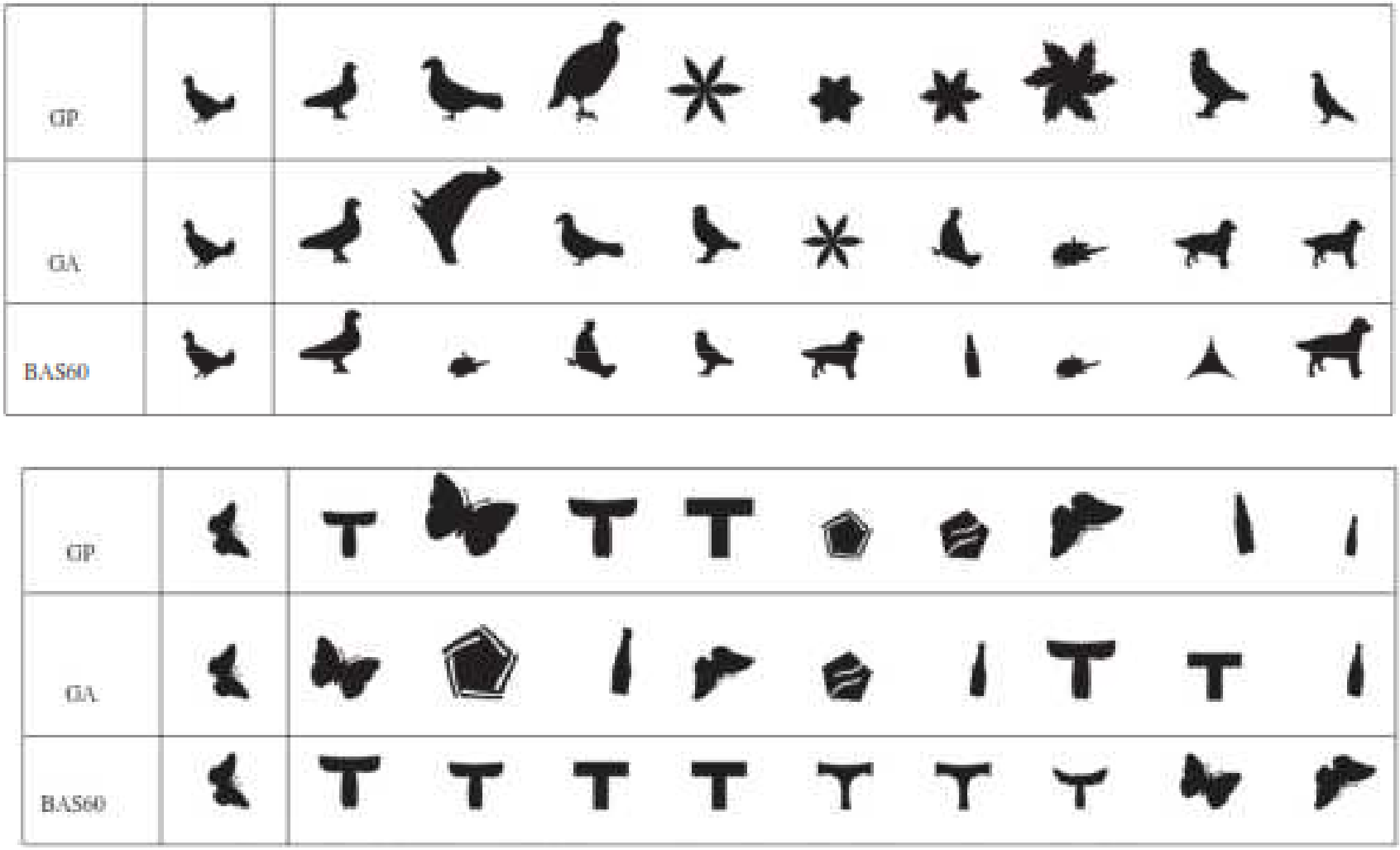
- Results

```
(+ (+ (+ ContourMSFractal 0_0) 1)
  (+ (+ (+ ContourMSFractal 0_0)
      (+ (sqrt 0_5)
        (sqrt (* BAS60 0_3))))
    (* (* BAS60 BAS60)
      (* (* BAS60 BAS60)
        (* (* (* BAS60 BAS60)
          (* (+ (/ BAS60
              (/ (+ ContourMSFractal MomentInvariants)
                (sqrt (+ 0_0 BAS60))))
            (sqrt (+ BAS40 0_0))) BAS40))
        (+ (+ (sqrt (* (* BAS60 BAS60)
            (* (* (* BAS60 BAS60)
              (+ (* BAS60 BAS60)
                (* 1 BAS60))) BAS40)))
          (sqrt 0_0)) BAS60))))))
```



Ricardo da S. Torres, Alexandre X. Falcão, Marcos A. Gonçalves, João P. Pap, Weiguo Fan, Edward A. Foxc, Baoping Zhang, **A genetic programming framework for content-based image retrieval**, Pattern recognition Vol. 42, 283—292, 2009

Content-based image retrieval



Ricardo da S. Torres, Alexandre X. Falcão, Marcos A. Gonçalves, João P. Pap, Weiguo Fan, Edward A. Foxc, Baoping Zhang, **A genetic programming framework for content-based image retrieval**, Pattern recognition Vol. 42, 283—292, 2009

Content-based image retrieval

- GP an effective approach to generate useful methods for information fusion
- No handcraft work is required to obtain competitive fusion strategies
- Main drawback: interpretation of operators?, generality?

Same conclusions as before

Questions?