# MACHINE INTELLIGENCE

## Genetic Algorithms and Computational Learning Theory

**K.S.Srinivas**
Department of Computer Science and Engineering

# MACHINE INTELLIGENCE

## Genetic Algorithms – Applications

**K.S.Srinivas**

Department of Computer Science and Engineering

# Genetic Algorithms : Applications

- **Optimization** − Genetic Algorithms are most commonly used in optimization problems wherein we have to maximize or minimize a given objective function value under a given set of constraints.

- **Neural Networks** − GAs are also used to train neural networks, particularly recurrent neural networks.

- **Parallelization** − GAs also have very good parallel capabilities, and prove to be very effective means in solving certain problems, and also provide a good area for research.

# Genetic Algorithms : Applications

- **Image Processing** – GAs are used for various digital image processing (DIP) tasks as well like dense pixel matching.

- **Vehicle routing problems** – With multiple soft time windows, multiple depots and a heterogeneous fleet.

- **Scheduling applications** – GAs are used to solve various scheduling problems as well, particularly the time tabling problem.

# Genetic Algorithms : Application Areas

**Machine Learning** –genetics based machine learning (GBML) is an area in machine learning.

**Robot Trajectory Generation** – GAs have been used to plan the path which a robot arm takes by moving from one point to another.

**DNA Analysis** – GAs have been used to determine the structure of DNA using spectrometric data about the sample.

# Genetic Algorithms :GA for Clustering

# Genetic Algorithms :GA for Clustering

- Choose random centroids population
- Assign instances to centroids
- Compute new centroids for each chromosome

51.6 72.3 18.3 15.7 29.1 32.2

- Select parents with min square error
- Cross over at length l say 2 means 29.1 and 32.2 are the tail
- Mutate by using $\delta$ from a normal distribution between 0 and 1
- At any position m ( randomly chosen) change the value v to

$$v \pm 2 * \delta * v, \quad v \neq 0,$$

$$v \pm 2 * \delta, \quad v = 0.$$

- Go to step 2 – terminate at convergence

# Genetic Algorithms :

## APPLICATIONS OF GENETIC  ALGORITHMS

# Genetic Algorithms : GA for Neural Networks

- 2 Things can be learnt
  Weights
  Hyper parameters

# Genetic Algorithms : GA for weight training

General framework of GAs for neural network training.

(i) Decode each individual in the current population into a set of connection weights and construct a corresponding ANN with the weights.

(ii) Evaluate the ANN by computing its total mean square error between actual and target outputs.

(iii) Determine fitness of individual as inverse of error. The higher is the error, the lower is the fitness.

(iv) Store the weights for mating pool formation.

(v) Implement search operators such as cross-over/mutation to parents to generate offsprings.

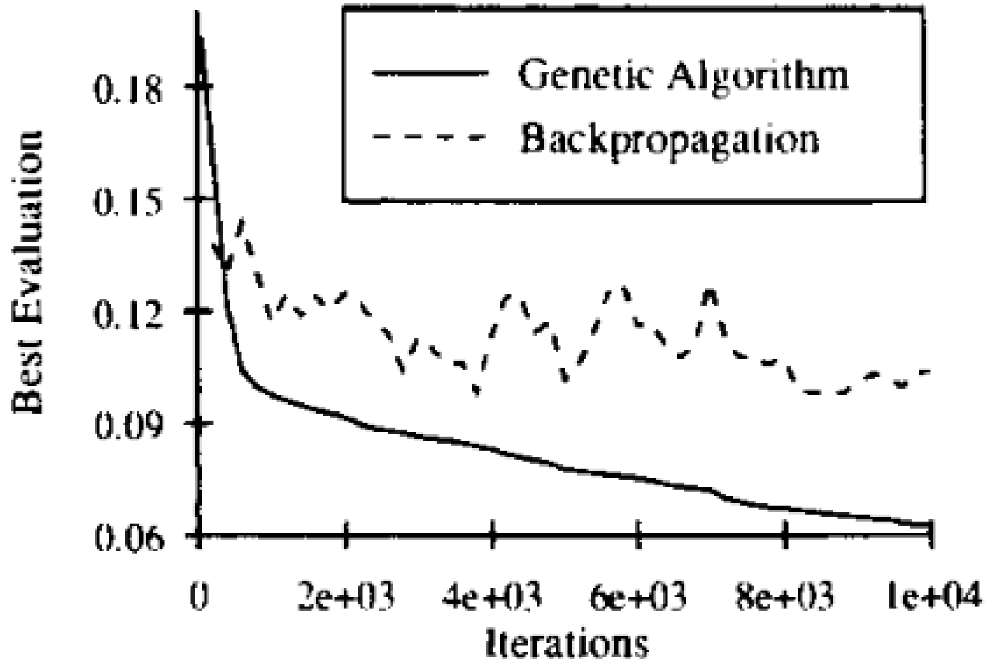
(vi) Calculate fitness for new population.

(vii) Repeat steps (iii) to (vi) until the solution converge.

(viii) Extract optimized weights.

# Genetic Algorithms : GA for Neural Networks

## Convergence - Rapid



# Genetic Algorithms : GA for ANN

# Genetic Algorithms : GA for ANN

# Genetic Algorithms : GA for ANN

## Fitness function

- The fitness is calculated as the accuracy of the ANN model.

$$Accuracy = \frac{NumCorrectClassify}{TotalNumSamples}$$

- Fitness = acc. Higher the accuracy, higher the fitness score.
- The result is that we never use back-propagation as our method for calculating the weights.

# Genetic Algorithms : GA for ANN

## Fitness

So vanishing or exploding gradients are no longer a problem.

The only operation required is “forward” propagation to measure the output and compare it with the target value. We could use mse or cross-entropy as the error measure.

# Genetic Algorithms : GA for ANN

## Disadvantages

If the number of parameters to be learnt are too many, then this may require a large initial population and also too many generations. High computing power will be needed.

A combination of GA and Gradient descent may help faster convergence.(?? Needs to be experimentally verified)

# Genetic Algorithms : GA for ANN

## GA for learning ANN hyper parameters

- Genetic Algorithms can also be used to learnt the network hyper parameters like – number of layers, number of neurons in each layer, the dropout rate, learning rate, activation units to be used etc.
- Each chromosome can encode all these as its genes.
- We can start with a set of chrosomes.

# Genetic Algorithms : GA for ANN

# Genetic Algorithms : GA for ANN

## GA can also be used for hyper-parameters

- Psss – network optimizers are weights, learning rates etc
- Each of the candidate activations can also be encoded in another vector w
- The chromosome is concatenation of uvw

- Number of layers (or the network depth)
- Neurons per layer (or the network width)
- Dense layer activation function
- Network optimizer

$U_i$: Binary variable, which take the value 1 if the $i^{th}$ hidden layer is used, and zero otherwise. where $i = 1, \dots, N$ , $j = 1, \dots, n_i$.

$$U = (U_1, \dots, U_N) \ , V = (V_{ij})$$

$V_{ij}$: Binary variable, which take the value 1 if the $i^{th}$ neuron of the $i^{th}$ hidden layer is used, and zero otherwise where $j = 1, \dots, n_i$. $i = 1, \dots, N$

# Genetic Algorithms : Algorithm

- Train all the NNs simultaneously or one by one.
- After all of them are done training, calculate their training cost.
- Calculate the “fitness”(how well it did in that iteration) of each NN based on its cost.
- Find the maximum fitness in the population(required for step 5, can be disregarded depending on the implementation of step 5).
- Pick 2 NNs based on a probability system regarding their fitness.
- Crossover the genes of the 2 NNs. This will create a “child” NN.
- Mutate the genes of the child. Mutating is required to maintain some amount of randomness in the GA.

# Genetic Algorithms : GA for ANN

## GA for learning hyper parameters of ANN

- The fitness function is the accuracy of the network.
- Unlike the case where we learnt the weights of the network, this time the architecture keeps changing and we get different fitness values for different architectures.
- We pick hyper parameter sets with best fitness.

$$Accuracy = \frac{NumCorrectClassify}{TotalNumSamples}$$

# Genetic Algorithms : GA in Decision Trees

- Decision Trees are usually constructed using the ID3 or the C4.5 Algorithms.
- After the tree is built, we usually resort to the Reduced Error Pruning or the Rules-post-pruning to increase testing accuracy.
- But we can use GA too.

# Genetic Algorithms : Application of GA in DT

Generation of Decision Tree

Rules generated from the decision tree can be optimized using genetic algorithm

This is not dealt exhaustively (meaning NOT in syllabus ) but I urge you to refer to paper which focuses on optimizing rules of decision – very briefly dealt

A Modified Decision Tree Algorithm Based on Genetic Algorithm for Mobile User Classification Problem

Dong-shengLiu1,2 andShu-jiangFan1 tree

http://dx.doi.org/10.1155/2014/468324

# Genetic Algorithms : GA in Decision Trees

PES UNIVERSITY ONLINE

# Genetic Algorithms : GA in Decision Trees

# Genetic Algorithms : GA in Decision Trees

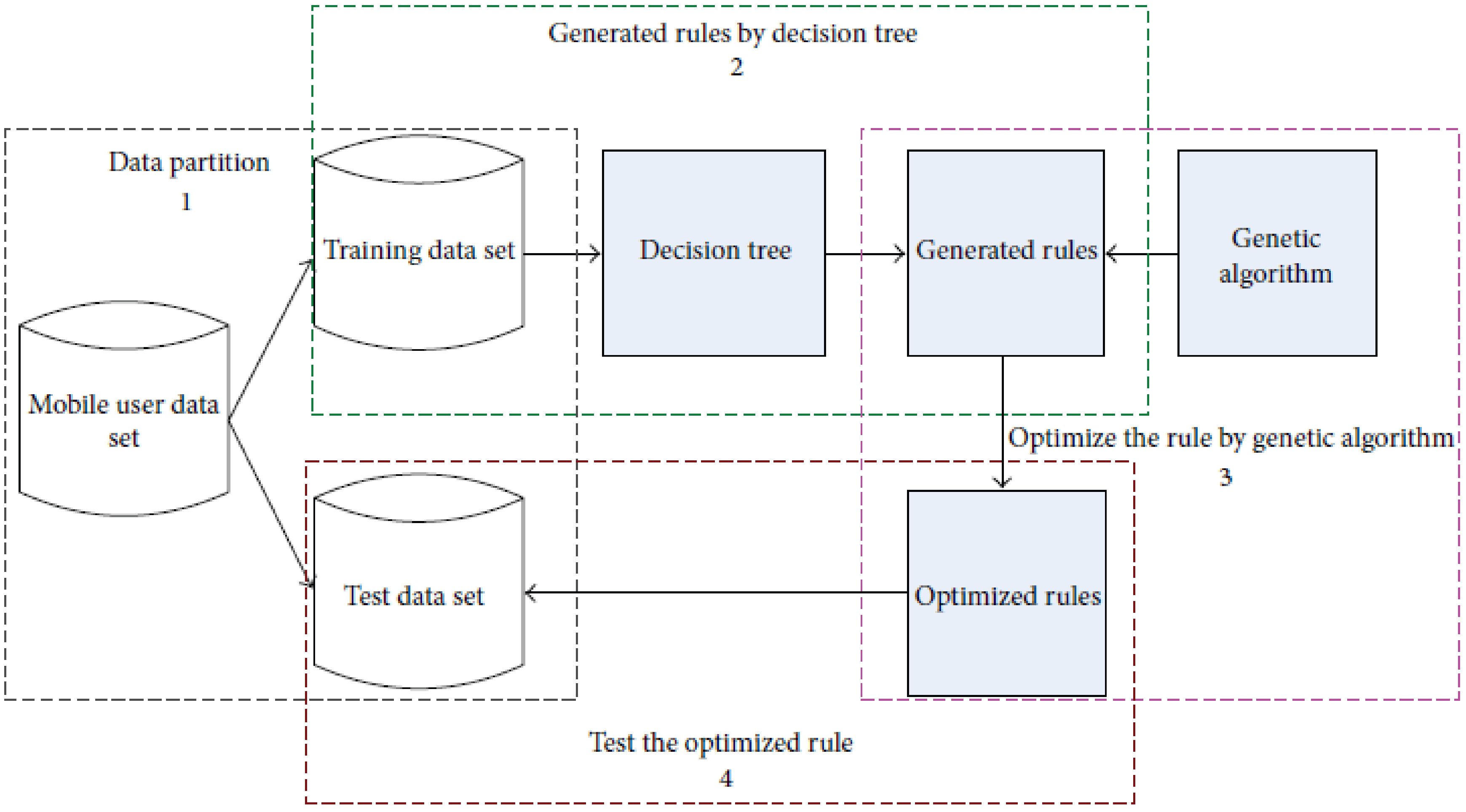



Figure 1: Framework of the proposed model.

# Genetic Algorithms : GA in Decision Trees

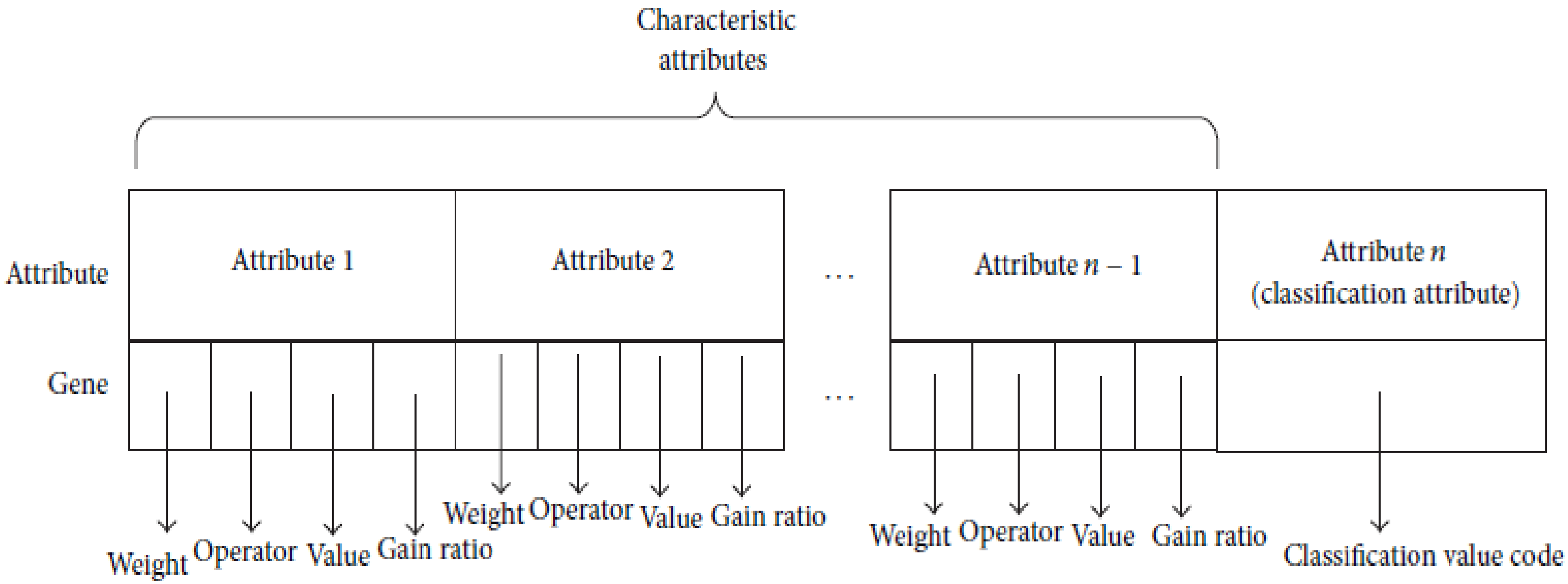



FIGURE 4: Chromosome construction.

# Genetic Algorithms : GA in Decision Trees

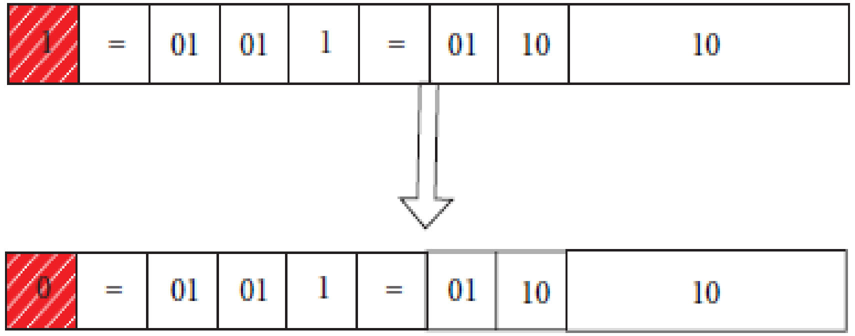



FIGURE 6: An example for weight mutation.

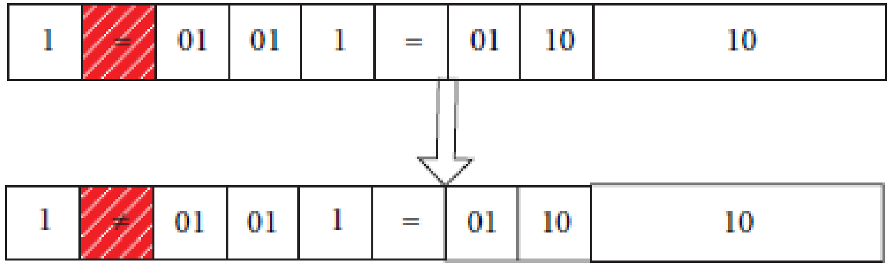



FIGURE 7: An example for operator mutation.

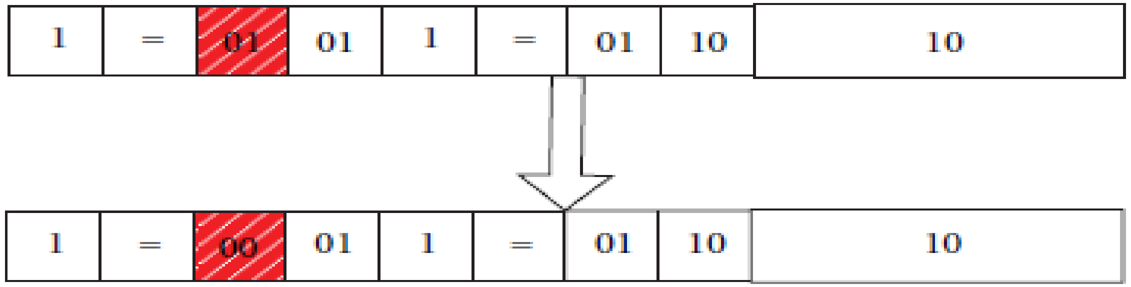



FIGURE 8: An example for value mutation.

# Genetic Algorithms : GA in Decision Trees

# Genetic Algorithms : GA in Decision Trees

Table 5: Comparison on accuracy.

| | Training data | Test data |
|---|---|---|
| DT-GA | 73.82% | 72.20% |
| C4.5 | 68.20% | 67.90% |
| SVM | 72.50% | 70.10% |

Table 7: Accuracy on Iris and Breast-cancer data.

| | Iris test data | Breast-cancer test data |
|---|---|---|
| DT-GA | 72.20% | 76.40% |
| C4.5 | 67.90% | 68.50% |
| SVM | 70.10% | 69.10% |

# Genetic Algorithms : GA in Decision Trees

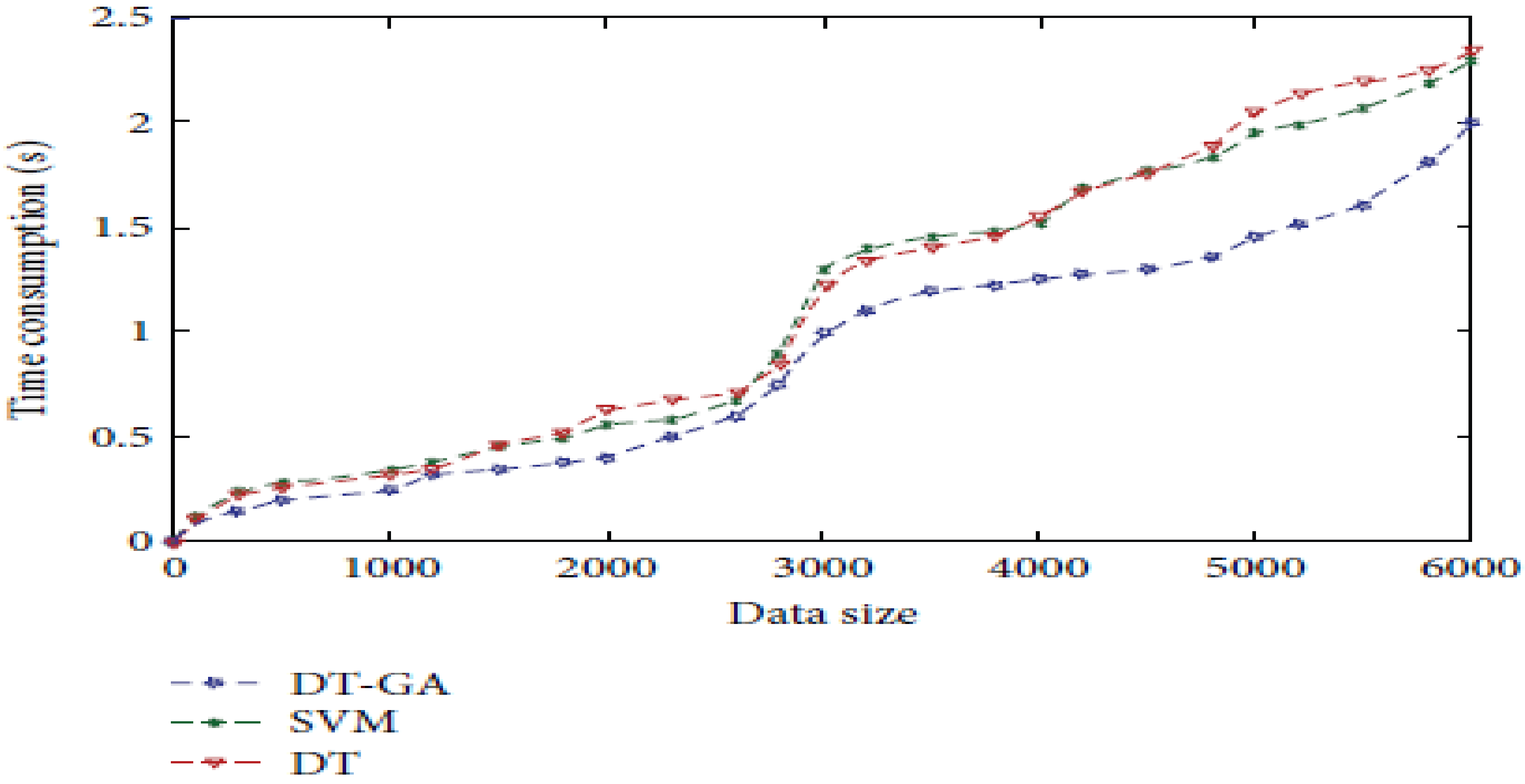



FIGURE 10: Time consumption comparison.

# THANK YOU

**K.S.Srinivas**

Department of Computer Science