How Many Parachutists will be Needed to Find a Needle in a Pastoral?

Akira Imada

Brest State Technical University
Belarus

Intrusion Detection by Soft-computing

CONTENTS:

- Supervised Learning
  - ⋆ Samples for *Training* and *Testing*
    - · *Iris-flower* & *KDD-cup-99* dataset
    - · A challenge using Iris-flower
- Conjecture 1

  "Intrusion is like a needle in hay not like an iris family"
  - ⋆ How to search for needles in hay? – 4 experiments

CONTENTS: (cont'd)

- Conjecture 2

  "No efficient such an algorithm"

  ⋆ Hoping to be a good debate

  ⋆ Yet other two challanges

- Concluding Remarks

How to design an Intrusion Detection System intelligently, if any?

- Two categories of detection
  - ⋆ Detection of *known attack*.
  - ⋆ Detection of *unknown no-normal*.

Detecting known-attacks meaningful?

⇓

A lookup-table enough?

⇓

Our interst is in detecting *unknown no-normal*.

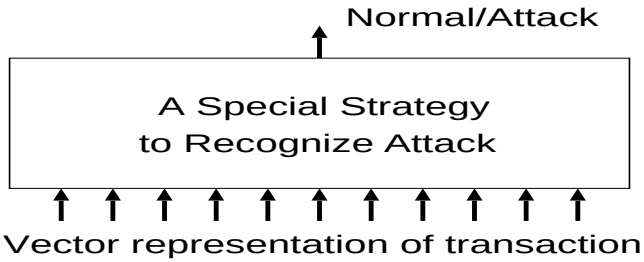
General assumption:

- TCP/IP connection to a network

  $\Rightarrow$ can be represented by a n-dimensional vector

Our Goal

- Is a vector input to our system a *Normal* or an *Attack*?



- A successful result in the literaturs seems to be

  $Accurate\ Detection\ Rate > 90\%$

  $False\ Alarm\ Rate < 10\ \%$

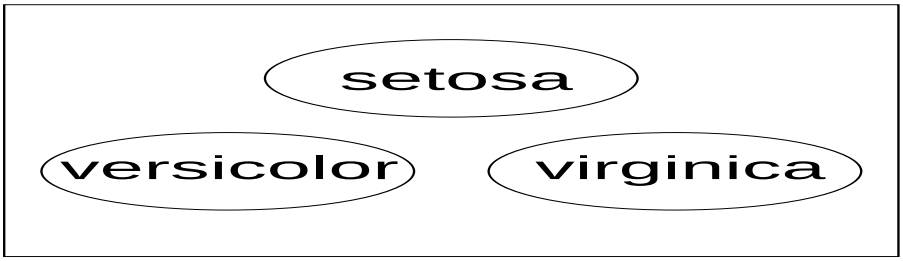
Training & Testing

We need a testset when we want a supervised methods

- Frequantly used two public domain datasets.
  - ⋆ Iris-flower dataset.
  - ⋆ KDD-cup-1999 dataset

What is iris flower dataset

- 3 families of iris flower ⇒ 150 samples in total;
- Each family ⇒ 50 samples; Each sample ⇒ 4 features.



IRIS flower

A reported success

Castellano et al. (2000)
assumed one to be normal while the other two abnormal.

$\Downarrow$

- abnormal detection rate = 96%
- false alarm rate = 0.6%

$\Downarrow$

But can we be so optimistic?
When a family of iris is normal then are others abnormal?

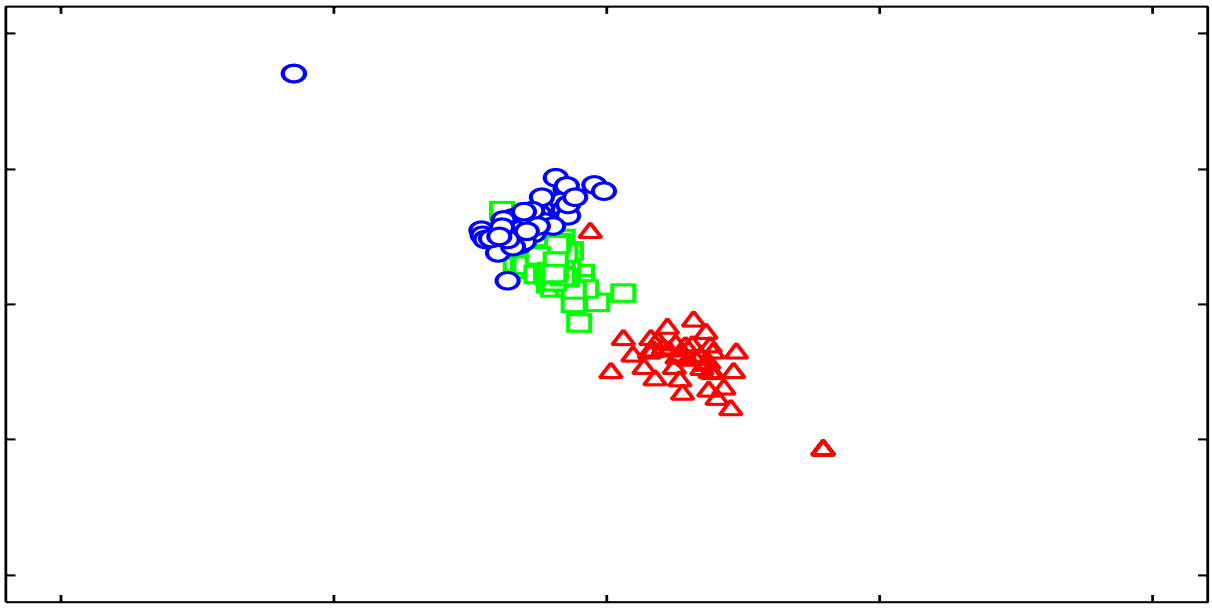
Let's visualize iris families — Sammon mapping

Sammon mapping maps points in a high-D space to 2-D

by

keeping distance relation preserved as much as possible
or
distances in the n-D space are approximated by distances in 2-D space with a minimal error.

A Sammon mapping of iris families



Where do outliers lie?

Not in the domain for the other families!!

$\Downarrow$

Not at the point at random, either.

$\Downarrow$

But outlier usually hides behind Normal.

Attacks by Mutant

□ Charrange 1

(1) Assume one family as Normal

(2) Mutate the Normal samples and take them as Attack.

(3) Train your system with half of the {Normal + Mutant}

(4) Test the system with remaining {Normal + Mutant}

⇓

Will a successful result be possible?

What is the KDD-cup-1999 Dataset?

- *Attack* $\Rightarrow$ 32 attack types of the 4 categories:
  - $\star$ *Probing*

    by proving a vulnerability of the network;
  - $\star$ *Denial-of-Service (DoS)*

    by denying legitimate requests to a system;
  - $\star$ *User-to-Root (U2R)*

    an unauthorized access to local super-user or root;
  - $\star$ *Remote-to-Local (R2L)*

    an unauthorized local access from a remote machine.
- *Normal*

How big is the KDD-cup-1999 dataset?

Labeled 4,898,430 records $\Rightarrow$ *Training*
Un-labeled 311,029 records $\Rightarrow$ *Testing*
and
Each record is 41-dimensional vector

$\Downarrow$

Sammon Mapping wouldn't work any more!

Can dimension be reduced?

- Kuchimanchi et al. (2004)
  - ⋆ 41-D to 19-D (by Principal Component Analysis)
    - · (detection accuracy, false positive) = (99.92%, 0.26%)
  - ⋆ while the original being
    - · (detection accuracy, false positive) = (99.94%, 0.23%)

    (both using Decision Tree)
- Joshi et al. (2005)
  - ⋆ 41-D to 5-D (taking the first five from above)
    - · (detection accuracy, false positive) = (79%, 21%)

    (using Hidden Markov Process)

It might be interesting here to see KDD-cup-99 winner's result

Detection rate for 4 attack types

Probe	*DoS*	*U2R*	*R2L*
83.3%	*97.1%*	*13.2%*	*8.4%*

Detection rate by Sabhnani et al. (2003)

	Probe	*DoS*	*U2R*	*R2L*
Multi-layer Perceptron	*88.7*	*97.2*	*13.2*	*5.6*
Gaussian Classifier	*90.2*	*82.4*	*22.8*	*9.6*
K-mean Clustering	*87.6*	*97.3*	*29.8*	*6.4*
Nearest Cluster Algorithm	*88.8*	*97.1*	*2.2*	*3.4*
Radial Basis Function	*93.2*	*73.0*	*6.1*	*5.9*
Leader Algorithm	*83.8*	*97.2*	*6.6*	*1.0*
Hypersphere Algorithm	*84.8*	*97.2*	*8.3*	*1.0*
Fuzzy Art Map	*77.2*	*97.0*	*6.1*	*3.7*
C4.5 Decision Tree	*80.8*	*97.0*	*1.8*	*4.6*

We doubt the success by Kuchimanchi et al.

⋆ 41-D to 19-D (by PCA)

  · (detection accuracy, false positive) = (99.92%, 0.26%)

⋆ while the original being

  · (detection accuracy, false positive) = (99.94%, 0.23%)

⇓

We won't go into detail here, but

Why U2R and R2L attacks resist to be detected?

Let's try a thought experiment!

Ratios in the samples for testing

Normal	*Probe*	*DoS*	*U2R*	*R2L*
19.5%	*1.3%*	*73.9%*	*5.2%*	*0.1%*

$\Downarrow$

Always-return-U2R-strategy would result in

Probe	*DoS*	*U2R*	*R2L*
0.0%	*0.0%*	*5.2%*	*0.0%*

$\Rightarrow$ Better than C4.5 in Sabhnani et al. (2003)

Or

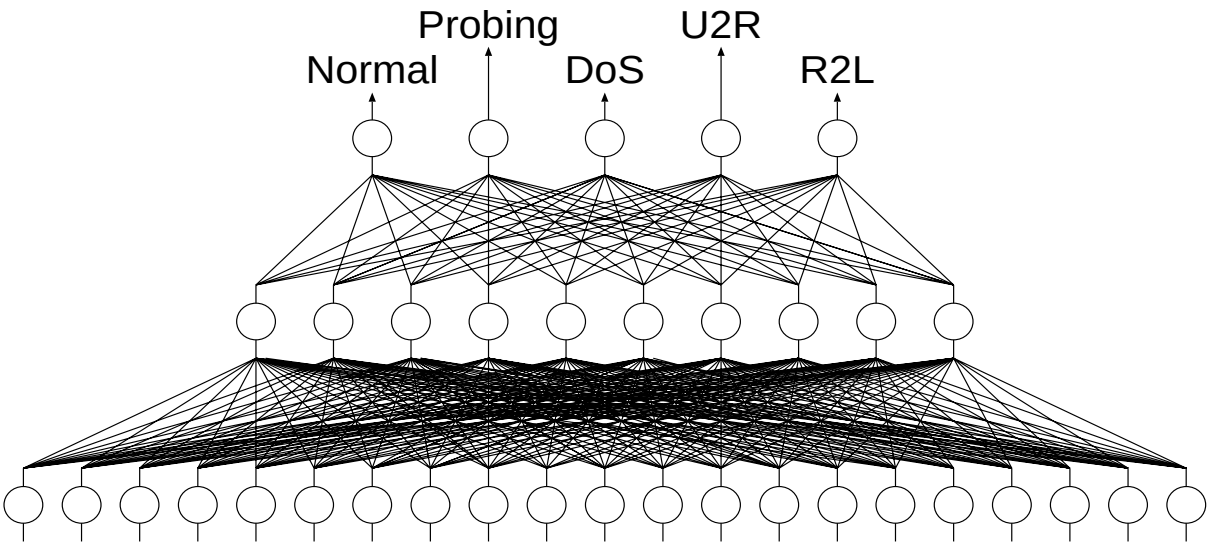
always-return-randomly-strategy

which returns either Normal, Probe, DoS, U2R, or R2L at random regardless of the input.

$\Rightarrow$ a high score to detect DoS attacks, like

Normal	*Probe*	*DoS*	*U2R*	*R2L*
19.5%	*1.3%*	*73.9%*	*5.2%*	*0.1%*

What we want?

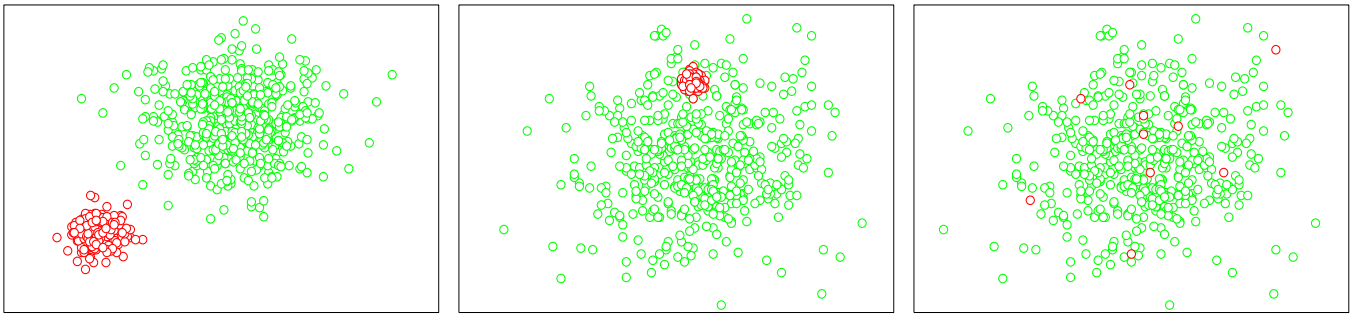


pre-proccessed 41 attributes in KDD cup 1999 data set

A-tiny-set-of-R2L vs. a-huge-set-of-Normal

	Total	Normal	Probe	DoS	U2R	R2L
Labeled for Training	4,898,430	19.9%	0.8%	79.3%	$\approx$ 0.0%	$\approx$0.0%
Non-labeled for Testing	311,029	19.5%	1.3%	73.9%	5.2%	0.1%

Fictitious but possible distributions



□ Conjecture 1

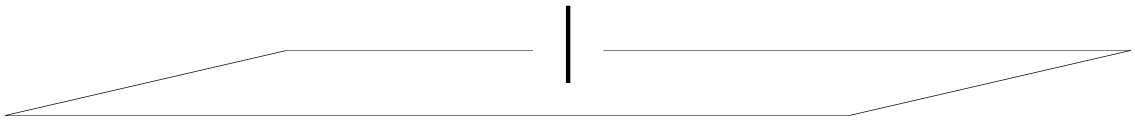
- *U2R, R2L and Real Attacks are like needles in a haystack.*

What does a needle look like in a haystack?

The original Hinton & Nowlan's Needle:

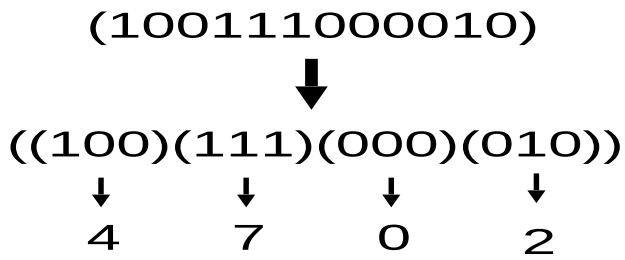
- A-needle ⇒ Only one configuration of 20-bit binary string.
  like (11000 11010 11101 0100)
- Haystack ⇒ $2^{20} - 1$ search points

A fictitious needle in a haystack in 2D.



Simple Gene

genotype & phoenotype



1. Random Fall of Parachutists

□ Algorithm 1

(1) Create a p-bit octal PIN at random.

(2) Create randomly one 3p-bit of binary string.

(3) Translate the string into p-bit octal code.

(4) Check if the translated code matches the PIN.

(5) If matches, end the run. Otherwise go back to 2.

$$\Downarrow$$

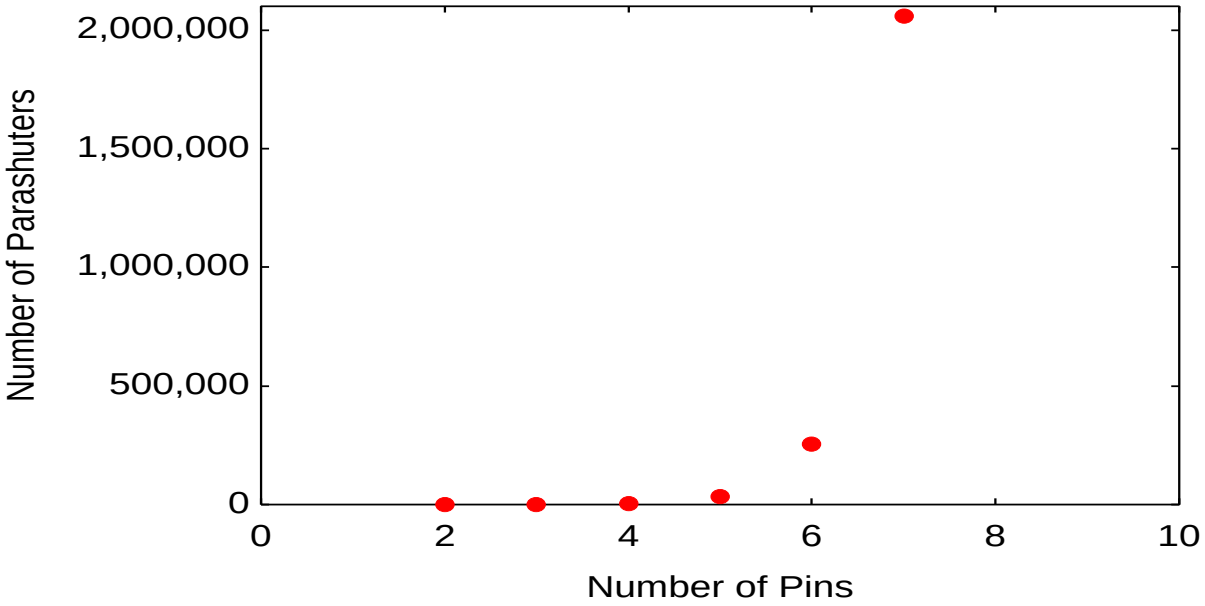
How many random tries will be needed to find the needle?

The reason of our title

$\Downarrow$

How many parachutists will be needed to find
a needle in a pastoral?

Random Fall – a critarion of comparison



7 digig octal $\Rightarrow$ 21-bit binary was a limit

2. What if they are allowed to walk after fall?

□ Algorithm 2

. . .

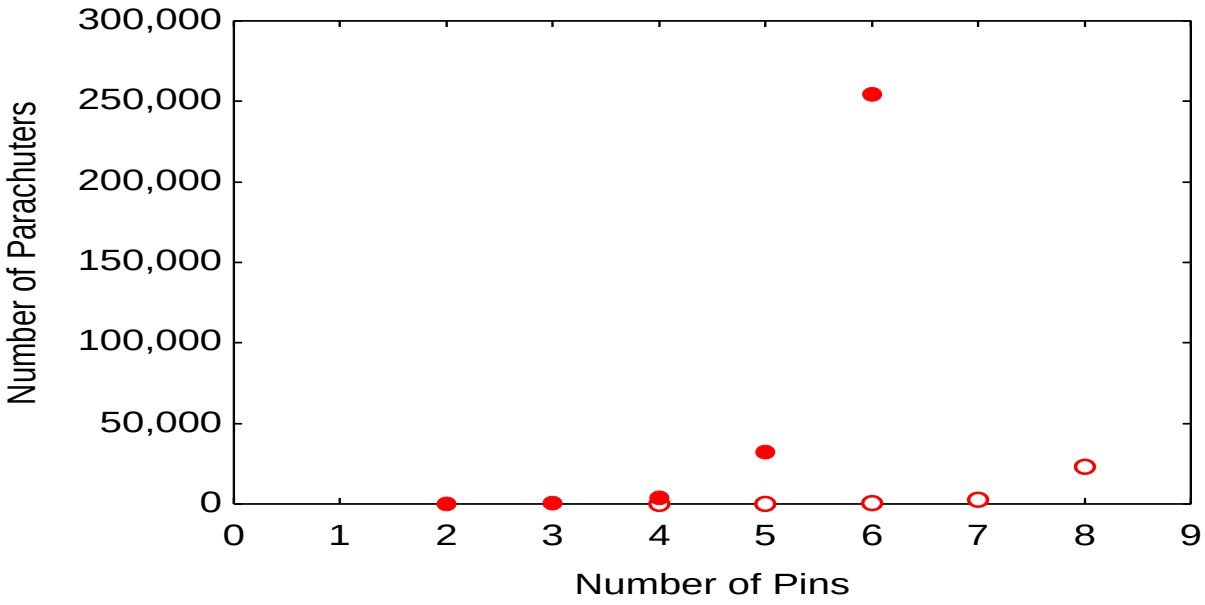
(5) If matches, end the run.

If not matches,

- *give a mutation (by flipping a bit chosen at random) with a probability of* $1/3p$

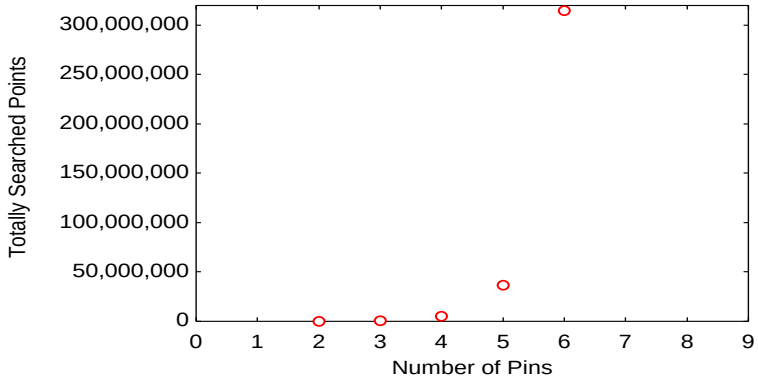
until the translation matches the PIN,
or number of steps exceeds 1000.

Random Fall & Walks after Fall



9 digig octal $\Rightarrow$ 27-bit never made a run stop

But really that efficient?



⇓

Alas, the actuall number of points searched for is much more than our random search.

3. Neutral Mutation

Neutral genotype-phenotype Mappings
— by Shipman et al. (2000)

What is neutral mutation?

One-to-many genotype-phenotype mapping
⇓
A mutation on a genotype without affecting phenotype

Walk by Neutral Mutation

□ Algorithm 3

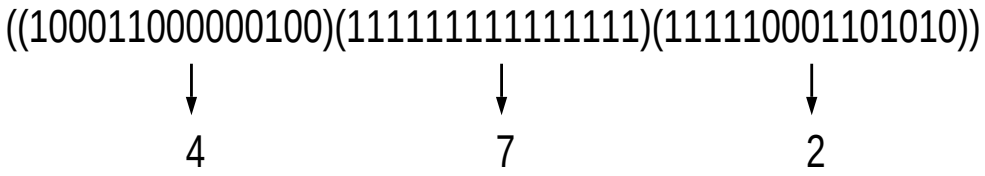
(3) Try point-wise mutation on the genotype such that the result maps into the same phenotype as the one before the mutation.

(4) Assess all possible single-mutation-neighbors of the new genotype to determine whether any new phenotype is discovered.

(5) Step 3 to 4 are repeated untill the phenotype matches the PIN, or untill a pre-fixed number of steps is reached.

To implement a neutrality

the-number-of-1 (mod 8) $\Rightarrow$ phenotype



$\Downarrow$

Again, our result was worse than our random search.

4. Mutation on Intron

What is mutation on intron?

Assume some of the genes do not affect phenotype.

$\Downarrow$

A mutation on the gene will not affect phenotype, either.

$\Downarrow$

neutral mutation

Does Neutral Mutation on Intron Enhance Efficiency of Search?

Yu & Miller (2002)

"Finding needles in haystack is not hard with neutrality"

vs.

Collin's (2005)

"Finding needles in haystack is harder with neutrality"

Even-Parity Problem

A strange fitness landscape

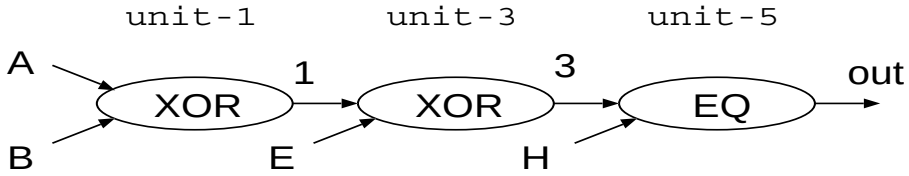
$\Downarrow$

Output $\Rightarrow$ all-correct/half-correct/not-correct-at-all

(1, 0.5 or 0 $\cdots$ no intermediate fitness value)

$$((XOR, A, B)(EQ, C, D)(XOR, 1, E)(EQ, F, G)(EQ, 3, H))$$

$\Downarrow$



Mutation on Intron

□ **Algorithm 4** – by Yu & Miller (2002)

(2) Create randomly an initial individual which is considered to be the winner to the next generation.

(3) Carry out point-wise mutation on the winning parent to generate 4 offspring.

(4) Construct a new generation with the winner and its offspring.

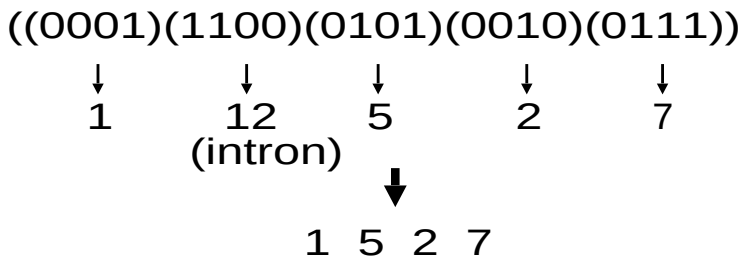
(5) Select a winner from the current population using the following rules.

(i) If any offspring has a better fitness than the parent, the one with highest fitness becomes the winner.

(ii) If fitness of all offspring have the same fitness as the parent, one offspring is randomly selected, and if the parent-offspring pair has a Hamming distance within the permitted range, the offspring becomes the winner, otherwise the parent remains as the winner.

(5) Back to step 2 unless the maximum number of generations reaches, or a solution is found.

Intron



$\Downarrow$

Almost similar result as our random search.

□ Conjecture 2

- *No such effective algorithm to look for a needle*

"Artificial immune system detects an attack!"

How fantastic it sounds!

A landmark

Forrest, Perelson et al. (1994)

"Self Nonself Discrimination in a Computer"

But two decades has passed since then and still an open issue.

⇓

We hope not, but isn't it just a fantasy?

Yet another problem: Training only by Normal

Can a Sommelier be trained without bootlegs?

Gomez et al. (2003)
"A set of fuzzy rules characterized abnormal space using only normal samples."
by
10% dataset of KDD-cup-1999 dataset
⇓
"It detects attacks with the detection rate 98.30% and false alarm rate 2.0%."
⇓
Really satisfactory, if it's really true.

Attack by Dummy

□ Charrange 2

(1) Prepare two sub-datasets from KDD-cup-1999 dataset. One is picked up from normal samples and call it D_{normal}*. The other is from attack samples and call it* D_{attack}*.*

(2) Furthermore, randomly create an attack dataset – dummy attacks, and call it D_{dummy}*.*

(3) Train your intrusion detection system only with D_{normal}*.*

(4) Then, try two tests, one with only D_{attack}*, and the other with only* D_{dummy}*, avoiding any a priori prediction.*

Placebo Experiment

Compare your system with a *random-replyer*

□ Charrange 3

(1) Create a simple device which randomly returns either one of Normal, Prove, DoS, U2R, or R2L for any input.

(2) Prepare a test dataset including enough amount of records uniformly from Normal, Prove, DoS, U2R, and R2L.

(3) Compare the performances of the detector you designed with the random-reply-machine created in step 1, feeding the same dataset prepared in step 2.

Conclusion

a hacker is a person who is extremely good at finding a pattern which is very close to the normal traffic

Sorry for this negatively sound conclusion
Needless to say, however, this article is not to negate the possibility, but we hope this will be a serious challenge to intrusion detection community to emerge real innovative ideas.