Use of K-Nearest Neighbor classifier for intrusion detection Yihua Liao, V. Rao Vemuri

Mingxing Gong

CISC850
Cyber Analytics

Outline

- Introduction
- Methodology
- Experiments
- Discussion & Conclusion

Outline

- Introduction
- Methodology
- Experiments
- Discussion & Conclusion

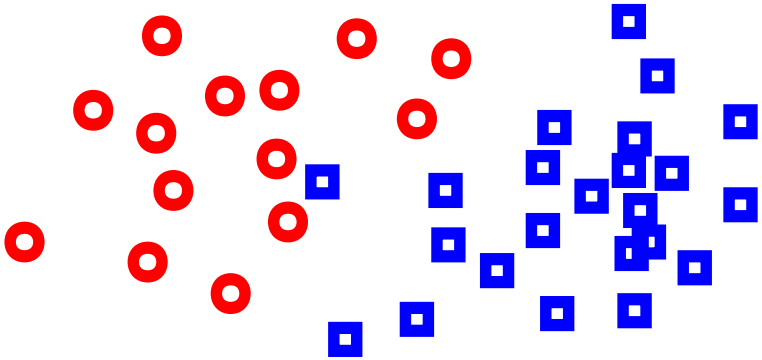
Introduction

- High false alarm probability or low attack detection accuracy

- Two general approaches:
  - Misuse detection
  - Anomaly detection

- Local ordering vs. frequency of system calls

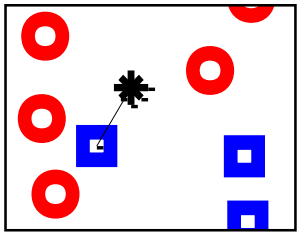
Nearest Neighbour Rule



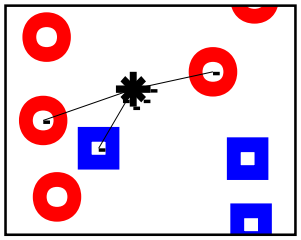
Consider a two class problem where each sample consists of two measurements (*x*,*y*).

Compute the *k* nearest neighbours and assign the class by majority vote.

k = 1



k = 3



Reference: www.robots.ox.ac.uk/~dclaus/cameraloc/samples/nearestneighbour.ppt

Outline

- Introduction
- Methodology
- Experiments
- Discussion & Conclusion

Methodology

- Apply text categorization methods to intrusion detection

Table 1: Analogy between text categorization and intrusion detection when applying the kNN classifier.

Terms	Text categorization	Intrusion Detection
N	total number of documents	total number of processes
M	total number of distinct words	total number of distinct system calls
n_i	number of times ith word occurs	number of times ith system call was issued
f_{ij}	frequency of ith word in document j	frequency of ith system call in process j
D_j	jth training document	jth training process
X	test document	test process

Methodology

- Each document is represented by a vector of words

$$a_{ij} = f_{ij}$$

- Weighting approach tf·idf (term frequency – inverse document frequency)

$$a_{ij} = \frac{f_{ij}}{\sqrt{\sum_{l=1}^{M} f_{lj}^2}} \times log\left(\frac{N}{n_i}\right)$$

- The cosine similarity is defined as follows:

$$sim(X, D_j) = \frac{\sum_{t_i \in (X \cap D_j)} x_i \times d_{ij}}{\|X\|_2 \times \|D_j\|_2}$$

Outline

- Introduction
- Methodology
- Experiments
- Discussion & Conclusion

Experiments

- DARPA data
- Cross validation and 50 distinct system calls

Table 2: List of 50 distinct system calls that appear in the training data set.

access	audit	auditon	chdir	chmod	chown	close	creat
execve	exit	fchdir	fchown	fcntl	fork	fork1	getaudit
getmsg	ioctl	kill	link	login	logout	lstat	memcntl
mkdir	mmap	munmap	nice	open	pathdonf	pipe	putmsg
readlink	rename	rmdir	setaudit	setegid	seteuid	setgid	setgroups
setpgrp	setrlimit	setuid	stat	statvfs	su	sysinfo	unlink
utime	vfork						

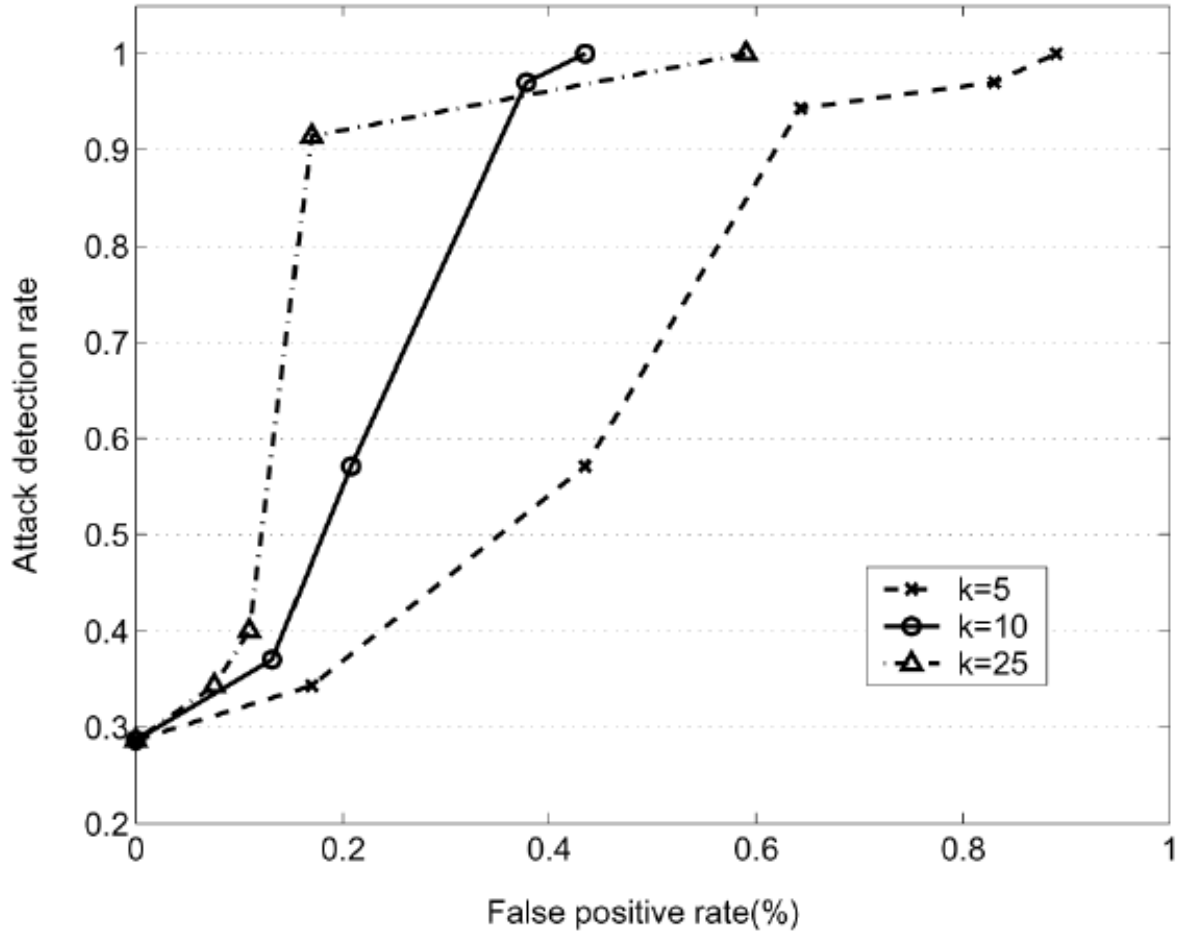
KNN classifier algorithm for anomaly detection

Figure 1: Pseudo code for the kNN classifier algorithm for anomaly detection.

build the training normal data set D;
for each process X in the test data **do**
  if X has an unknown system call **then**
    X is abnormal;
  else then
    for each process D_j in training data **do**
      calculate $sim(X, D_j)$;
      if $sim(X, D_j)$ equals 1.0 **then**
        X is normal; exit;
    find k biggest scores of $sim(X, D)$;
    calculate sim_avg for k-nearest neighbors;
    if sim_avg is greater than $threshold$ **then**
      X is normal;
    else then
      X is abnormal;

KNN classifier performance

Figure 2: Performance of the kNN classifier method expressed in ROC curves for the tf•idf weighting method. False positive rate vs attack detection rate for k=5, 10 and 25.



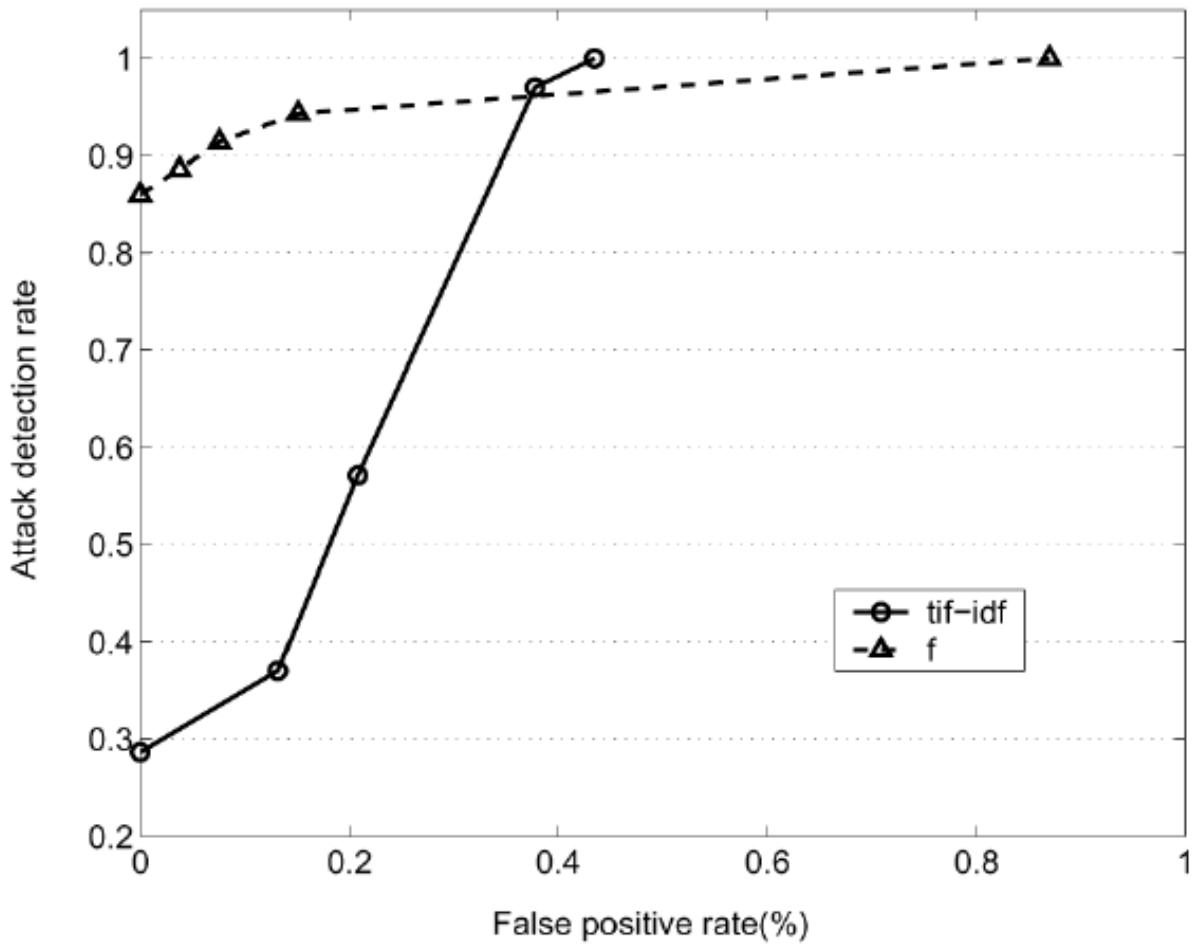
Anomaly Detection

- The overall running time of the kNN method is *O(N)*

- Integrate with signature verification

Frequency Weighting vs. tf·idf Weighting

Figure 3: ROC curves for tf•idf weighting (k=10) and frequency weighting (k=15).



Frequency Weighting vs. tf·idf Weighting

Table 3: Attack detection rate for DARPA testing data when anomaly detection is combined with signature verification.

Attack	Instances	Detected	Detection rate
Known attacks	16	16	100%
Novel attacks	8	6	75%
Total	24	22	91.7%

Outline

- Introduction
- Methodology
- Experiments
- Discussion & Conclusion

Discussion

- kNN Classifier advantages
- Compared tf·idf weighting with the frequency weighting
- Classification cost can be further reduced by only using most influential system calls

Conclusion

- kNN Classifier is able to effectively detect intrusive program behavior with low false positive rate

- Further research is in process to investigate the reliability and scaling properties of the kNN classifier method

Reference

[1] www.robots.ox.ac.uk/~dclaus/cameraloc/samples/nearestneighbour.ppt

[2] Yihua Liao, V. Rao Vemuri, 'Use of K-Nearest Neighbor classifier for intrusion detection', *Computers & Security, Volume 21, Issue 5*, 1 October 2002, Pages 439-448