Binary and multi-class classification in network anomaly detection using deep neural networks

Michele Pagano

Joint work with Christian Callegari and Stefano Giordano

Department of Information Engineering
University of Pisa

Information Technologies and Mathematical Modelling
2-5 December 2020, Tomsk

Outline

Outline

Why an intrusion detection system?

- Network security mainly means *prevention*
  - Physical protection for hardware
  - Passwords, access tokens, etc. for *authentication*
  - *Access control list for* authorization
  - Cryptography for *secrecy*
  - *Backups and redundancy for* authenticity

BUT . . .

. . . Absolute security cannot be guaranteed!

$\Rightarrow$ Need for a system which acts when prevention fails

Intrusion Detection System

An intrusion detection system or IDS is a software/hardware tool used to detect unauthorized access to a computer system or a network

IDS Taxonomy: Host based vs. Network based

Host based IDS

- Aimed at detecting attacks related to a specific host
- Architecture/Operating system dependent
- Processing of high level information (e.g. system calls)
- Effective in detecting insider misuse

Network based IDS

- Aimed at detecting attacks towards hosts connected to a LAN
- Architecture/Operating system independent
- Processing data at lower level of granularity (packets)
- Effective in detecting attacks from the “outside”

IDS Taxonomy: Misuse based vs. Anomaly based

Misuse based (Signature based, Rule based) IDS

- Identifies intrusion by looking for patterns of traffic or of application data presumed to be malicious
- Pattern of misuses are stored in a database
- Effective in detecting only “known” attacks
- Encrypted traffic ???

Anomaly based IDS

- Identifies intrusions by classifying activity as either anomalous or normal
- Need a training phase to recognize normal activity
- Able to detect “new” attacks
- Generates more false alarms than a misuse based IDS

IDS Taxonomy: Stateless vs. Stateful

Stateless IDS

- Treats each event independently of others
- Simple system design
- High processing speed

Stateful IDS

- Maintains information about past events
- The effect of a certain event depends on its position in the events stream
- More complex system design
- More effective in detecting distributed attacks

IDS Taxonomy: Centralized vs. Distributed

Centralized IDS

- All the operations are performed by the same machine
- More simple to realize
- Only one point of failure

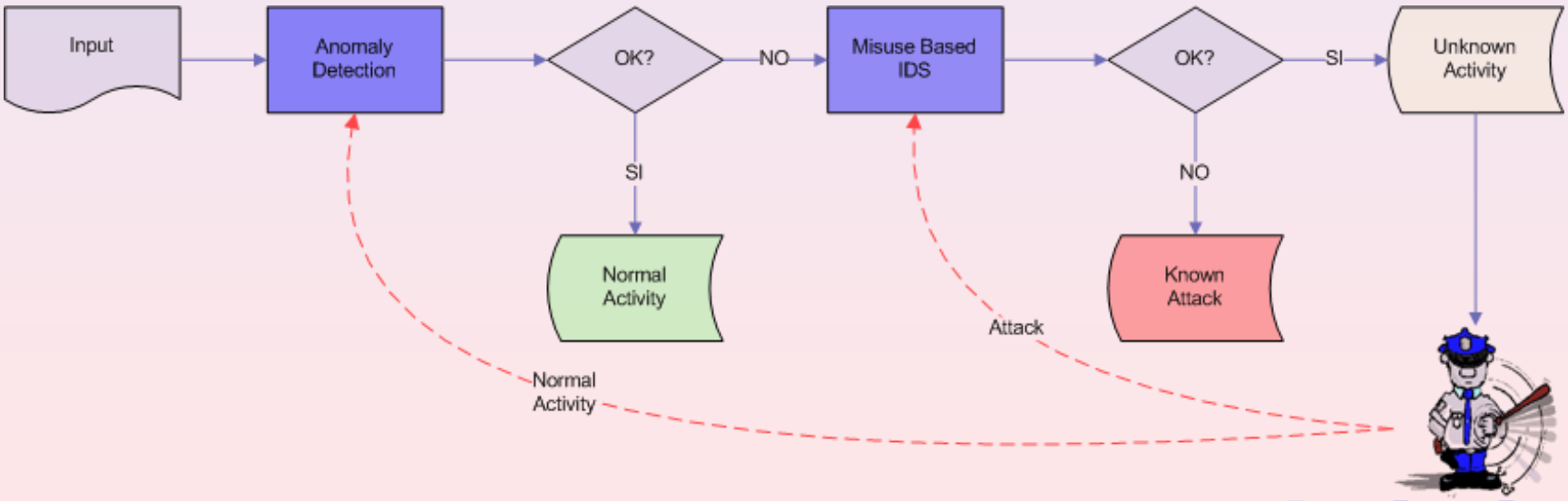
Distributed IDS

- Composed of several components
  - **Sensors** which generate security events
  - **Console** to monitor events and alerts and control the sensors
  - Central **Engine** that records events and generate alarms
- May need to deal with different data formats
- Need of a secure communication protocol (IPFIX)

The best choice?

Combined use

- HIDS (for insider attacks) & NIDS (for outsider attacks)
- Stateless IDS (fast data process) & Stateful IDS (for “complex” attacks)
- Misuse IDS (low False Alarm rate) & Anomaly IDS (for “new” attacks)



Anomaly detection:Metrics and Models

Metrics

- *Event counter*
- *Interval timer*
- *Resource measure*

Statistical models

- *Operational model*: abnormality is decided by comparing observations x_n of a RV x (representing a quantitative measure accumulated over a period) with a fixed threshold
- *Mean and standard deviation model*: abnormality is decided by checking if x_n falls inside the confidence interval
- *Multivariate model*: based on the correlations between two or more metrics
- *Markov process model*: based on the transition probabilities
- *Time series model*: takes into account order and inter-arrival time of the observations

Network anomaly detection: Traffic descriptors

- Traffic parameters, which vary significantly from the normal behavior to the anomalous one
- The number of potential traffic descriptors is huge (some papers identify up to 200 descriptors)

Some examples

- Packet length
- Inter-arrival time
- Flow size
- Number of packets per flow

For each descriptor we can consider

- Mean Value
- Variance and higher order moments
- Distribution function
- Quantiles

Basic definitions

- A = alarm
- $-A$ = not an alarm
- I = attack (intrusion)
- $-I$ = not an attack

False Positive (FP)

The error of rejecting a null hypothesis when it is actually true $\Rightarrow$ creation of an alarm in correspondence of normal activities

$$P(A|-I) = \text{False positive (alarm) probability}$$

False Negative (FN)

The error of failing to reject a null hypothesis when it is in fact not true $\Rightarrow$ missed detection

$$P(-A|I) = \text{False negative probability}$$

Performance indexes

- Detection Rate (DR)

$$DR = \frac{TP}{TP + FN}$$

- False Alarm Rate (FAR)

$$FAR = \frac{FP}{FP + TN}$$

- Accuracy (ACC)

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- ROC (Receiver Operating Characteristics) curve: Detection Rate vs. False Alarm Rate
- AuC (Area under the Curve)

MAWI-Lab Traffic Traces

- MAWI (Measurement and Analysis on the WIDE Internet) archive (sample-points B and F)
  - One raw traffic trace per day, containing 15 minutes of traffic (pcap file)
  - Real traffic from a transpacific backbone link connecting Japan and the United States

MAWILab Project

Anomaly labelling is obtained combining the output of four anomaly detectors

- Hough transform
- Gamma distribution modelling
- Kullback-Leibler divergence
- Principal Component Analysis

Why MAWI repository?

- Unlike the widely-used DARPA98-99 dataset, the network is not emulated and the traffic mixture is representative of the current mixtures of network services and applications
- Unlike KDD (derived from DARPA 1998), the dataset contain raw traffic traces
- Unlike other datasets (i.e., CIDDS-001 and CIDDS-002, UNSW-NB 15, CICIDS2017, and the Kyoto honeypot) the traffic is given by real traffic captured over a real backbone network and offers a good variety of both normal and attack flows

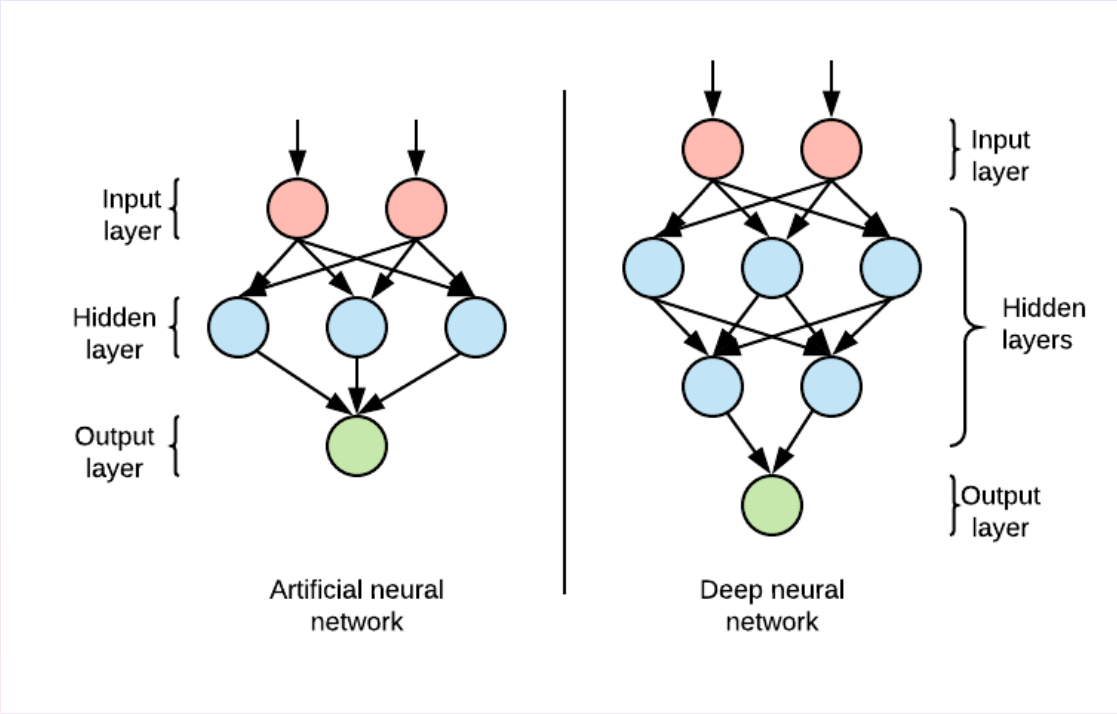
MAWILab : Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking

R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, ACM CoNEXT 2010

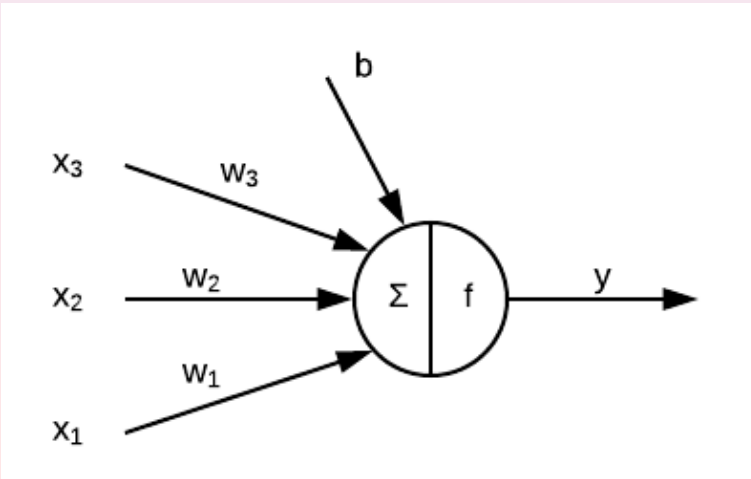
Outline

Artificial Neural Networks

- An Artificial Neural Network (ANN) is composed of artificial neurons
- The neurons are organized into three kinds of layers
  - input layer
  - hidden layer
  - output layer

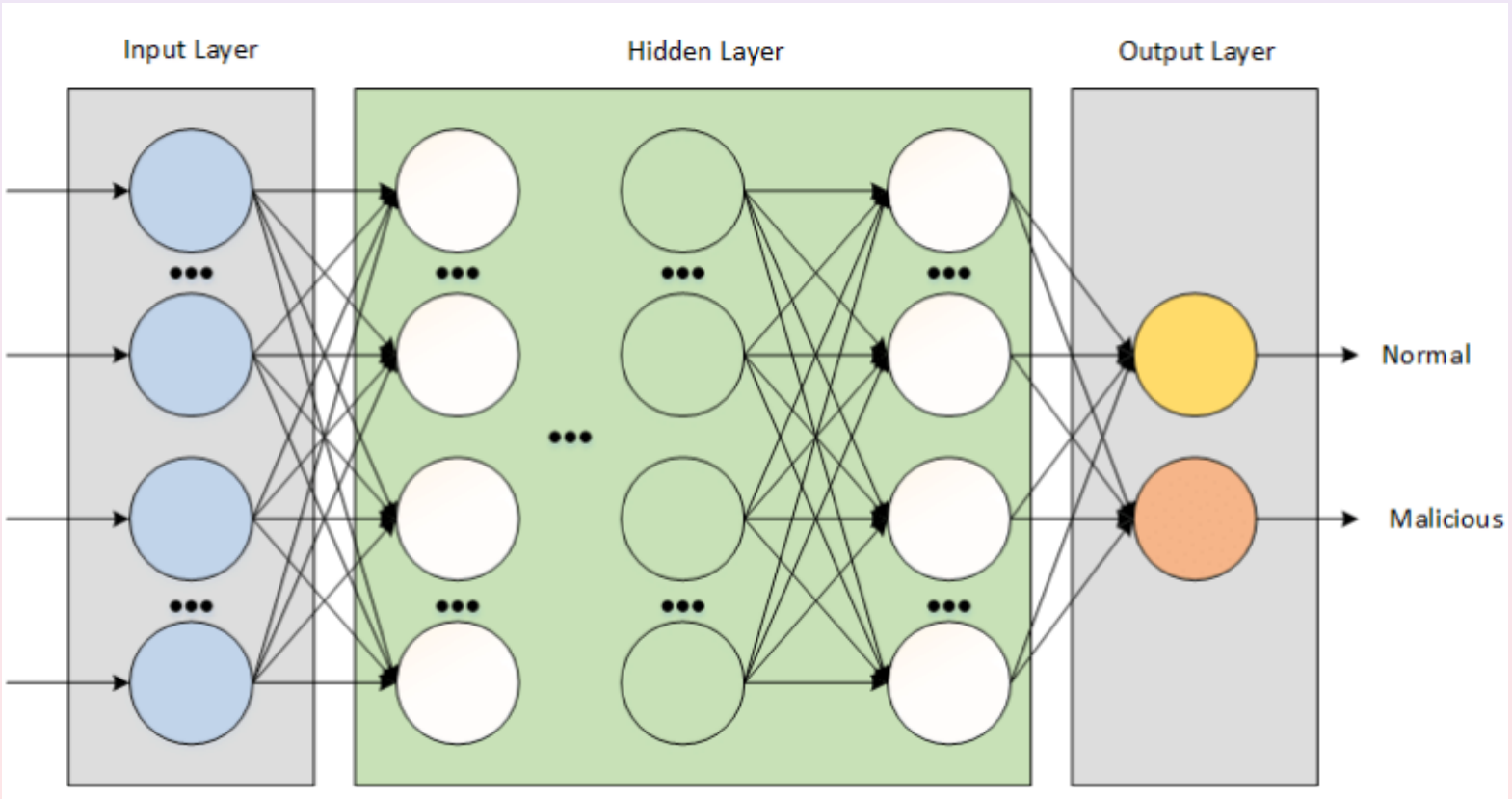


- Each neuron calculates the weighted sum of the inputs and applies an activation function to compute the output



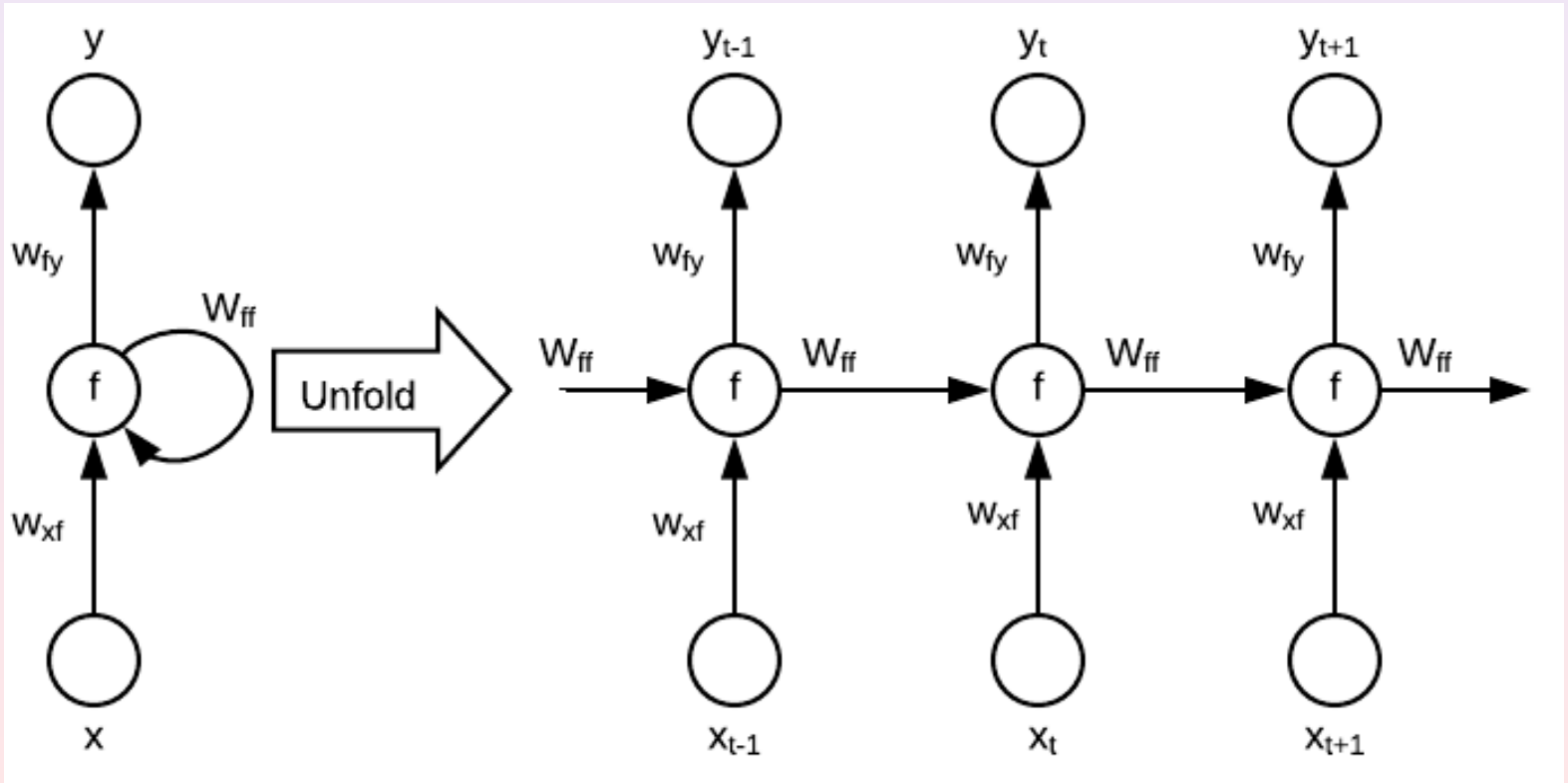
Multi–Layer Perceptron

- The multilayer perceptron (MLP) is the most basic kind of feedforward neural network
- It is a fully connected network, where each neuron is connected to all of the neurons in previous layer



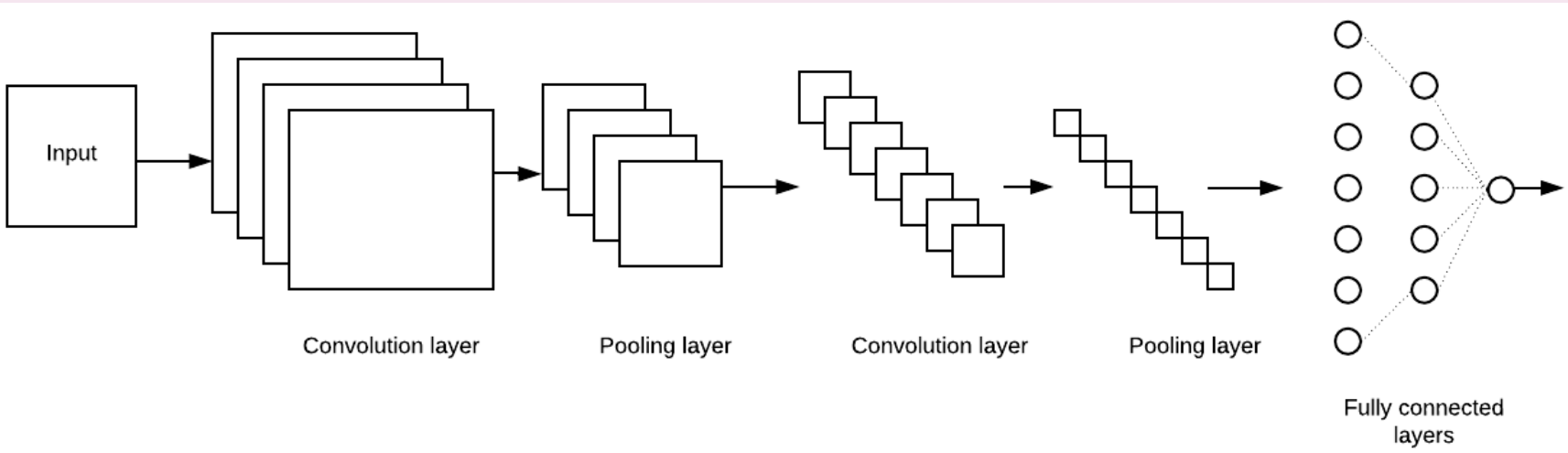
Recurrent Neural Network

- A Recurrent Neural Network (RNN) is composed of recurrent neurons
- A recurrent neuron is connected to all nodes in the previous layer but it has also a recurrent connection that connects the output back to the node itself



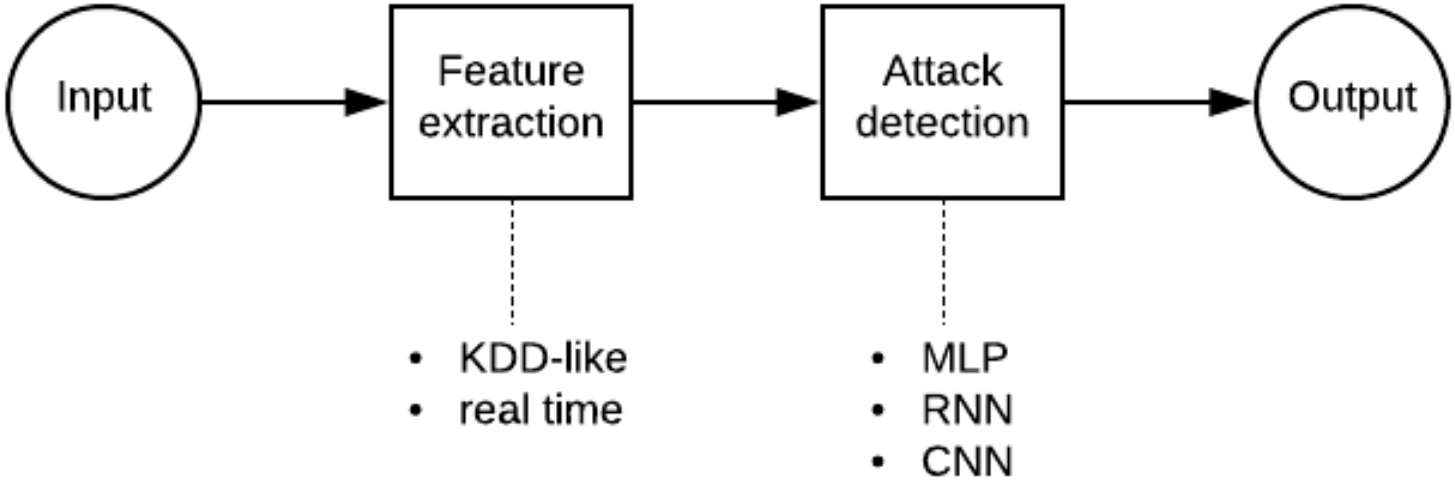
Convolutional Neural Network

- Convolutional neural network (CNN) is a particolar feed-forward neural network used for finding patterns in images
- CNNs are composed of an input layer, one or more convolutional layers followed by a pooling layer, one or more fully connected layers (FCLs) and one output layer
- CNNs have a lot of hyperparameter, such as the number of convolutional layers, the number of filters, the filter size, and the number of FCLs



Outline

System Architecture and implementation



Implemented using Python3

- Keras: high-level deep learning library
- TensorFlow: machine learning library (as backend)
- CPU versions of the libraries (not optimized for GPU)

Feature Extraction

The feature extraction module generates 17 flow features in real time from the traffic data using a combination of hash tables

Basic features	SPORT	Source port of the connection
	DPORT	Destination port of the connection
	PROT	Protocol
	NPKTS	Number of packets
	NBYTES	Number of bytes
	NSYN	Number of packets with the SYN flag set
	. . .	ACK, RST, FIN, PUSH, URG, ECN
	DUR	Duration of the connection
Aggregated features	SAMEAP1	number of connections to same SOURCEADDR and SPORT of the current flow
	SAMEAP2	number of connections to same DESTADDR and DPORT of the current flow
	SAMEA1	number of connections to the same SOURCEADDR of the current flow
	SAMEA2	number of connections to the same DESTADDR of the current flow

Feature Extraction

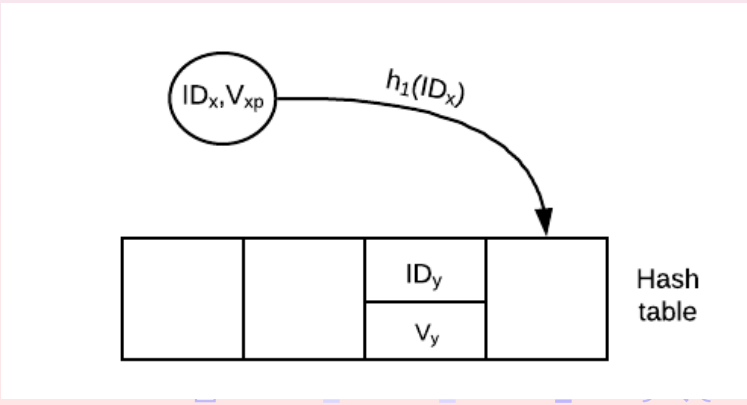
- A flow is identified by the tuple

(SIP, DIP, SPORT, DPORT, PROT)

- A packet is analyzed as an array $[ID_x, v_x^1, ..., v_x^n]$ where ID_x is the flow identifier and v_x^i is the value of the i^{th} feature
- Each entry of the hash table is an array containing the flow ID and the features values
- For each packet the corresponding flow entry is computed applying the hash function to the flow identifier

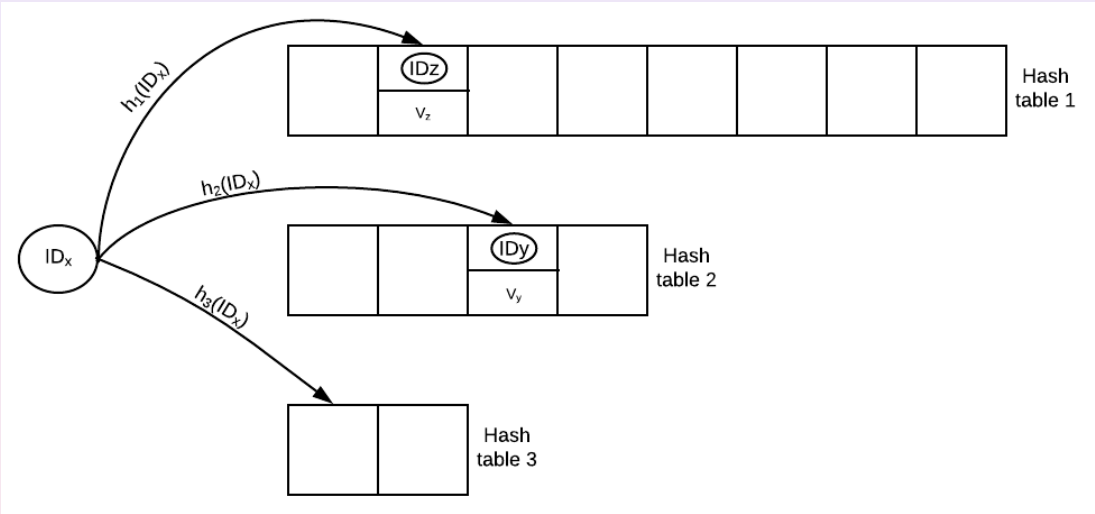
- If the flow identifier matches the flow identifier stored in the entry the feature value is updated:

$$S^i[h(ID_x)] += v_x^i$$



More on hash tables

- Three distinct levels (each with an independent hash function) to solve the problem of collisions (due to the use of hash functions)



- DUR is the only non-additive feature
  $\Rightarrow$ Store the time-stamps of the first and last observed packets of the flow
- Different IDs for aggregated features
  $\Rightarrow$ Two additional hash structures

Attack Detection

- Features are fed to the attack detection block, where a DNN with 17 input nodes is used for classification
  - Multi-Layer Perceptron
  - Recurrent Neural Network
  - Convolutional Neural Network
- Dataset for performance evaluation
  - Five consecutive days (900 s) – July 2018 data in MAWI
  - Training: first 400 s of the first two days
  - Testing: all remaining traffic $\approx 116.8 \cdot 10^6$ flows
- Architecture parameter tuning on a reduced dataset
  - Single day of traffic
  - The first 400 s for training and the remaining for testing
- The block can perform
  - Binary classification: 2 output nodes
  - Multiclass classification: 12 output nodes

Performance indexes

- Detection Rate (DR)

$$DR = \frac{TP}{TP + FN}$$

- False Alarm Rate (FAR)

$$FAR = \frac{FP}{FP + TN}$$

- Accuracy (ACC)

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- Execution time

  Inference time for a 2 s time-bin $\approx$ 65000 flows

Binary classification: Tuning

CNN tuning: number of convolutional layers

Layers	Time(s)	DR	ACC	FAR
1	0.90	0.537	0.311	0.707
2	1.20	0.906	0.888	0.113
3	1.39	0.854	0.809	0.194
4	1.52	0.964	0.786	0.228
5	1.57	0.980	0.126	0.943

CNN tuning: number of filters

Filters	Time(s)	DR	ACC	FAR
16	0.65	0.639	0.910	0.068
32	0.78	0.923	0.825	0.183
64	1.03	0.879	0.422	0.615
128	1.25	0.956	0.868	0.139
16,32	0.65	0.788	0.883	0.109
32,64	0.90	0.555	0.242	0.783
64,128	1.26	0.964	0.852	0.157

Binary classification: Tuning

CNN tuning: filter dimension

Filter size	Time(s)	DR	ACC	FAR
2	1.16	0.846	0.885	0.112
3	1.19	0.923	0.869	0.135
4	1.23	0.965	0.778	0.237
5	1.28	0.889	0.833	0.172
6	1.23	0.982	0.169	0.897

CNN tuning: number of FCLs and neurons

FCL	neurons	Time(s)	DR	ACC	FAR
1	16	1.08	0.775	0.876	0.116
1	32	1.10	0.937	0.866	0.140
1	64	1.13	0.941	0.845	0.162
2	32,16	1.98	0.926	0.889	0.114
2	64,32	2.47	0.824	0.921	0.072

Binary classification: *Optimal configurations*

Parameter	MLP	RNN
Number of hidden layers	5	3
Number of neurons	17	10
Activation function	ReLU	Tanh
Kernel initializer	normal	normal
Loss	BC	MSE
Learning rate	0.001	0.001
Epochs	1	5
Batch size	32	32

Parameter	CNN
Number of CNN layers	2
Number of filters	64,128
Size of filters	3
Strides	1
Padding	same
Number of FCL layers	1
Number of neuron FCL	32
Activation function	ReLU
Kernel initializer	normal
Loss	BC
Learning rate	0.001
Epochs	2
Batch size	32

Binary classification: Results

- Tests executed on a general purpose PC
  - AMD Ryzen 9 3900x at 4.2 Ghz
  - 32GB of RAM

DNN	Time(s)	DR	ACC	FAR
MLP	0.38	0.698	0.907	0.076
RNN	0.99	0.735	0.804	0.190
CNN	1.10	0.926	0.899	0.103

- These results are far from being as good as those in the literature when testing the system over KDD
- Performance are sligtly better than in the *few works with NNs* that test the system over the MAWILab dataset, where accuracy is always lower than 0.85

Multi-class classification: Overall results

Actual low classification

Class	Type of Attack	Flow number
0	Normal	107931015
1	Unknown	2605
2	Other	7029
3	HTTP	3570496
4	Multi. Points	3670526
5	Alpha Flow	125365
6	IPv6 Tunneling	86980
7	Port Scan	1020
8	NET Scan ICMP	5192
9	NET. Scan TCP	1182666
10	NET. Scan UDP	215890
11	DOS	17182

Performance comparison

DNN	Time(s)	DR	ACC	FAR
MLP	0.35	0.780	0.914	0.075
RNN	0.79	0.642	0.918	0.059
CNN	2.00	0.859	0.897	0.099

Multi-class classification: *Optimal configurations*

Parameter	MLP	RNN
Number of hidden layers	3	2
Number of neurons	30	30
Activation function	ReLU	Tanh
Kernel initializer	normal	normal
Loss	BC	CE
Learning rate	0.001	0.001
Epochs	5	5
Batch size	32	32

Parameter	CNN
Number of CNN layers	2
Number of filters	64,64
Size of filters	3
Strides	1
Padding	same
Number of FCL layers	1
Number of neuron FCL	32
Activation function	ReLU
Kernel initializer	normal
Loss	CE
Learning rate	0.001
Epochs	2
Batch size	32

Multi-class classification: Confusion matrix

CNN with *optimal* configuration

A/P	0	1	2	3	4	5	6	7	8	9	10	11
0	98410410	3	0	6032714	3324365	1975	0	0	0	59386	102162	0
1	1137	0	0	1411	57	0	0	0	0	0	0	0
2	2920	0	0	2133	1976	0	0	0	0	0	0	0
3	542895	4	0	2576228	447983	76	0	0	0	2889	421	0
4	1288878	9	4	810112	1494278	378	0	0	0	7083	69776	8
5	45348	12	1	5510	74067	380	0	0	0	26	20	1
6	78427	0	0	2657	4987	0	0	0	0	909	0	0
7	21	0	0	876	123	0	0	0	0	0	0	0
8	2455	0	0	1653	1084	0	0	0	0	0	0	0
9	500477	0	0	509665	126294	0	0	0	0	25472	20758	0
10	42653	0	0	51759	27238	0	0	0	0	0	94240	0
11	12452	0	0	2408	2301	0	0	0	0	21	0	0

Outline

Conclusions

- We have proposed an anomaly based IDS based on Deep learning able to work in real time
- Comparison of different DNNs
  - Multi-Layer Perceptron
  - Recurrent Neural Network
  - Convolutional Neural Network
- Best performance offered by CNNs
  - Better DR/FAR
  - Robustness wrt tuning parameters and training dataset
  - Misclassification of *rare* classes
- Future Work
  - Different DNNs: Short-Long Term Memory (SLTM), Gated Recurrent Unit (GRU)
  - More datasets
  - Deep Reinforcement Learning


СПАСИБО ЗА
ВНИМАНИЕ
И ПОПЫТКУ
ПОНИМАНИЯ