ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/328345684

Blockchain Security Architecture

Preprint · October 2018

DOI: 10.13140/RG.2.2.26604.41607

CITATIONS
0

READS
30

1 author:

Konstantinos Demertzis
Democritus University of Thrace
97 PUBLICATIONS **885** CITATIONS

SEE PROFILE

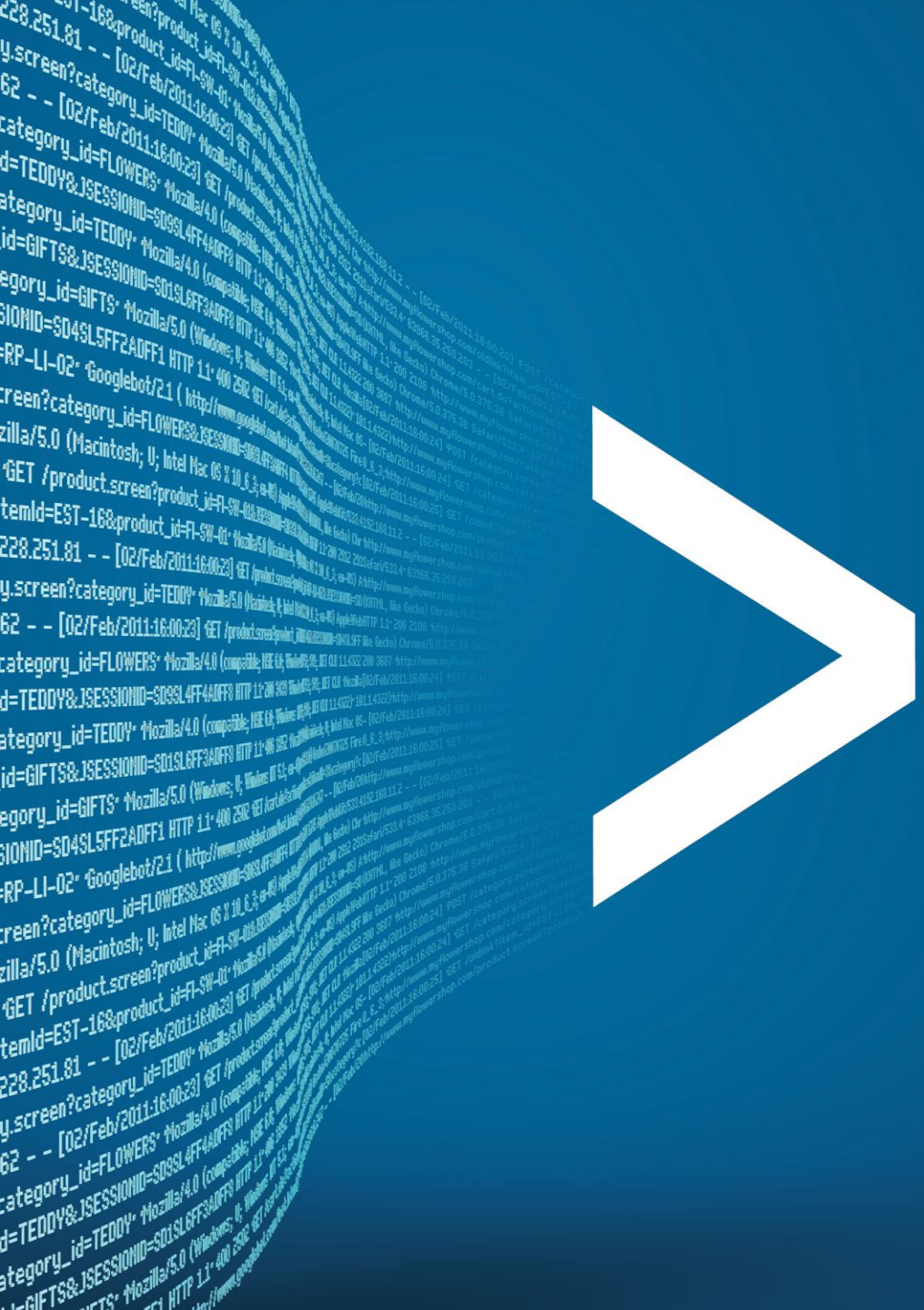
Some of the authors of this publication are also working on these related projects:

Smart Energy Grids View project

Forest Fires View project

All content following this page was uploaded by Konstantinos Demertzis on 17 October 2018.

The user has requested enhancement of the downloaded file.



An IIoT Blockchain Security Architecture Based on Deep Learning Smart Contracts

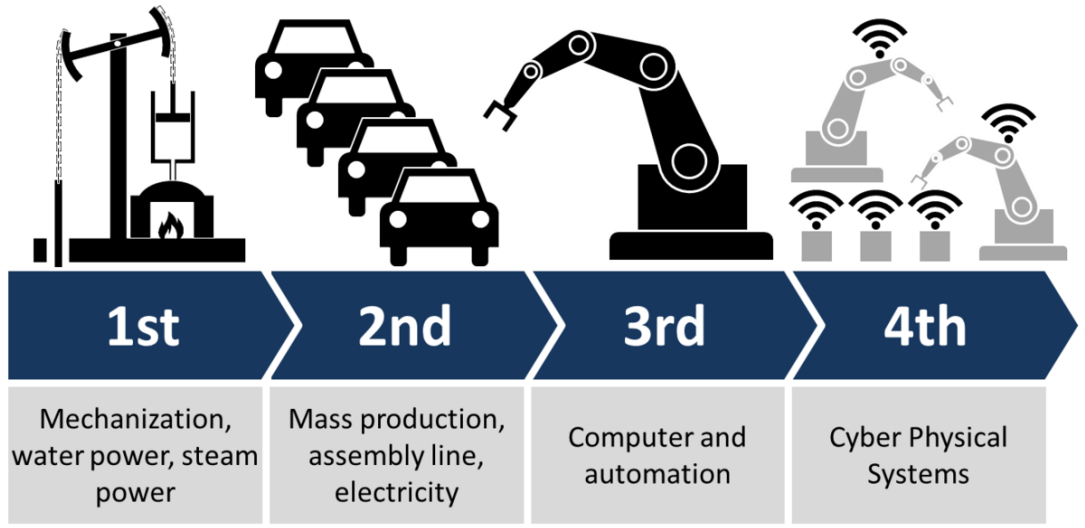
Konstantinos Demertzis

No system is safe!!!

INTRODUCTION

Outline

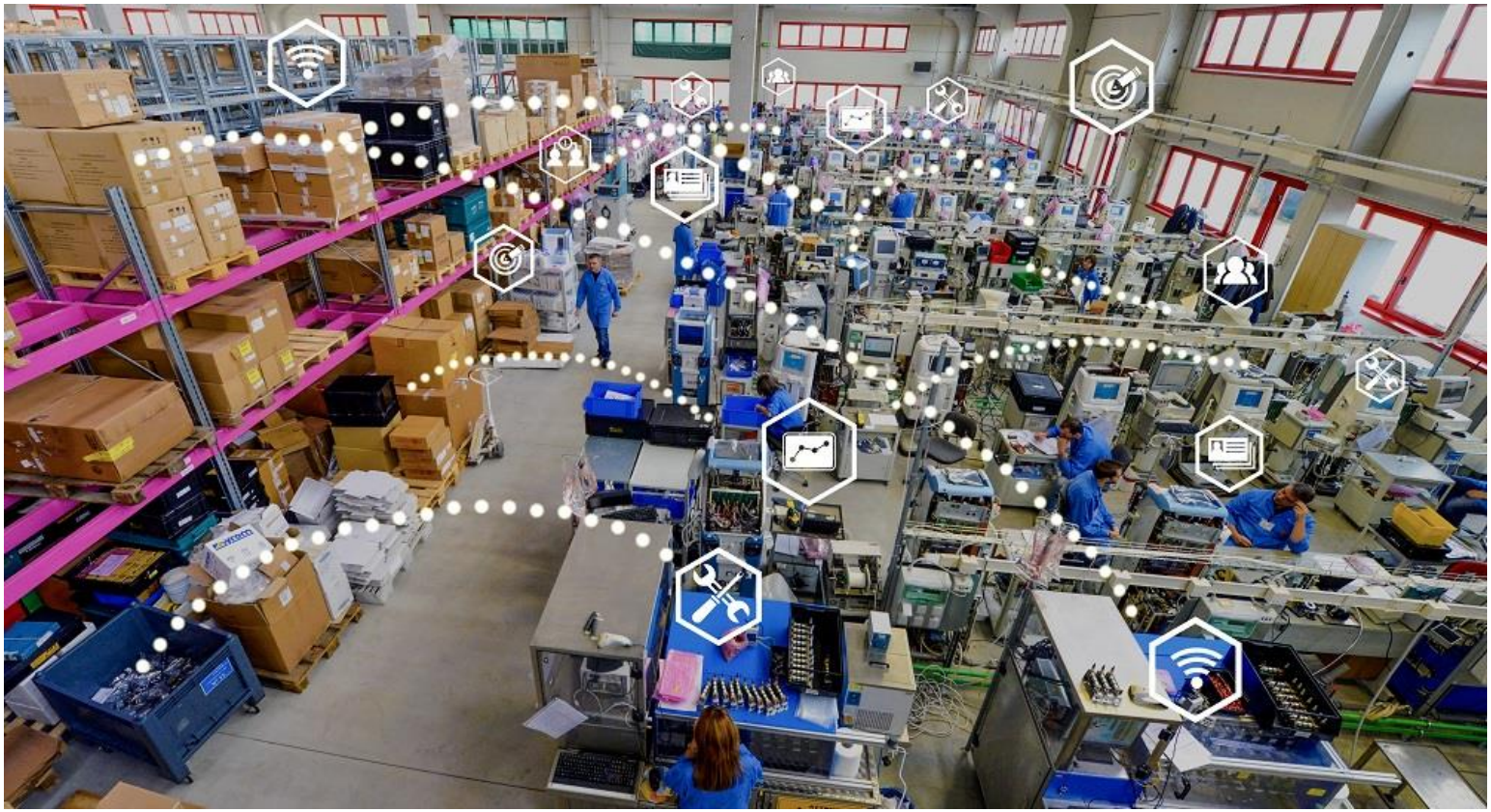
- Industrial Internet of Things (IIoT) Cyber Security
- Deep Learning and Anomalies Detection
- IIoT and Blockchain – Proposed Architecture
- Conceptual Model
  - Requirements – Challenges – Goals
- Approach and Methodology
  - The proposed IIoT Blockchain Network
  - The Deep Learning Smart Contracts
  - An example of Deep Learning Smart Contract
- Data
  - Dataset – Features Extraction – Data pre-Processing and Threshold Criteria
- Results
- Applications Of Proposed Architecture
- Discussion – Innovation – Future Directions



Plans are nothing, planning is everything

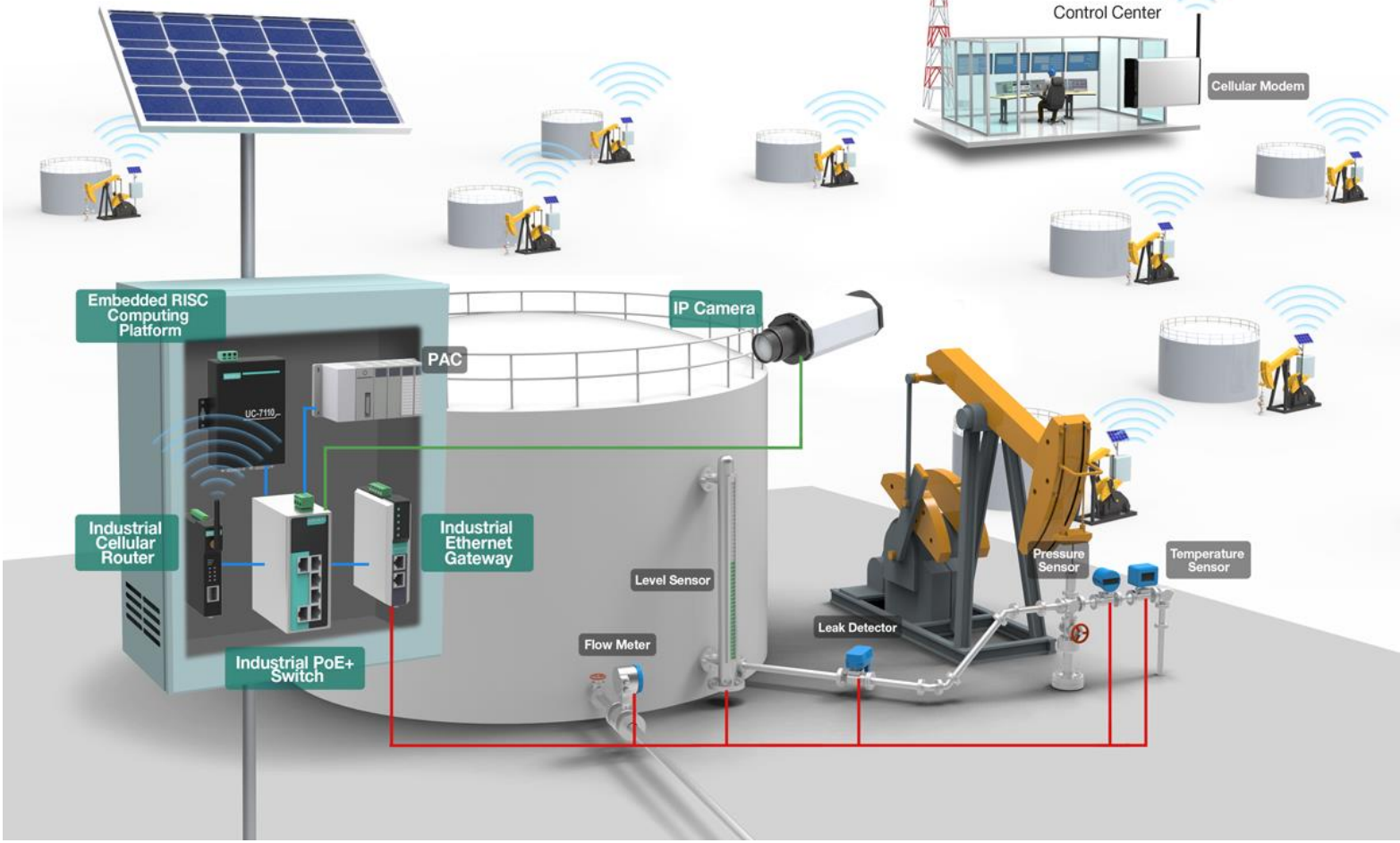
INDUSTRIAL IOT CYBER SECURITY

What is IIoT?



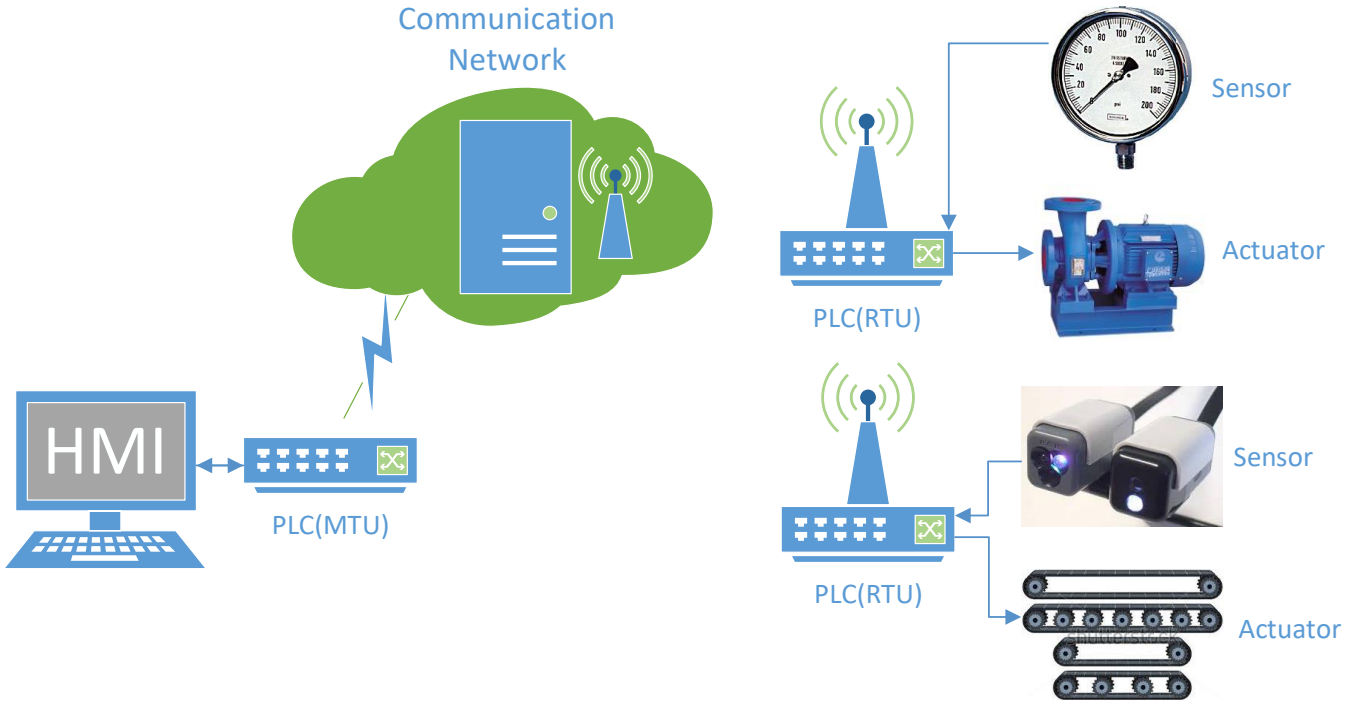
listen to your data®

IIoT Cyber Security



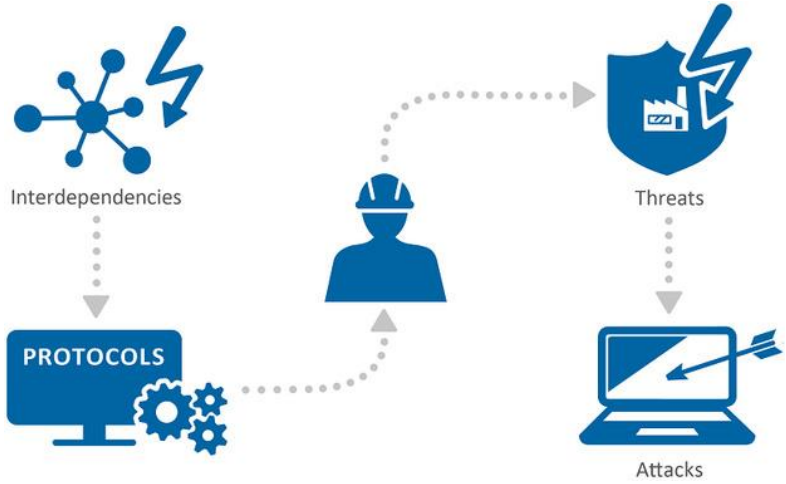
Typical SCADA topology

- They are generally made up of four major components.
  - Sensing and Actuation
  - Programmable Logic Controllers (PLCs)
  - Communication Network
  - Human Machine Interface (HMI)



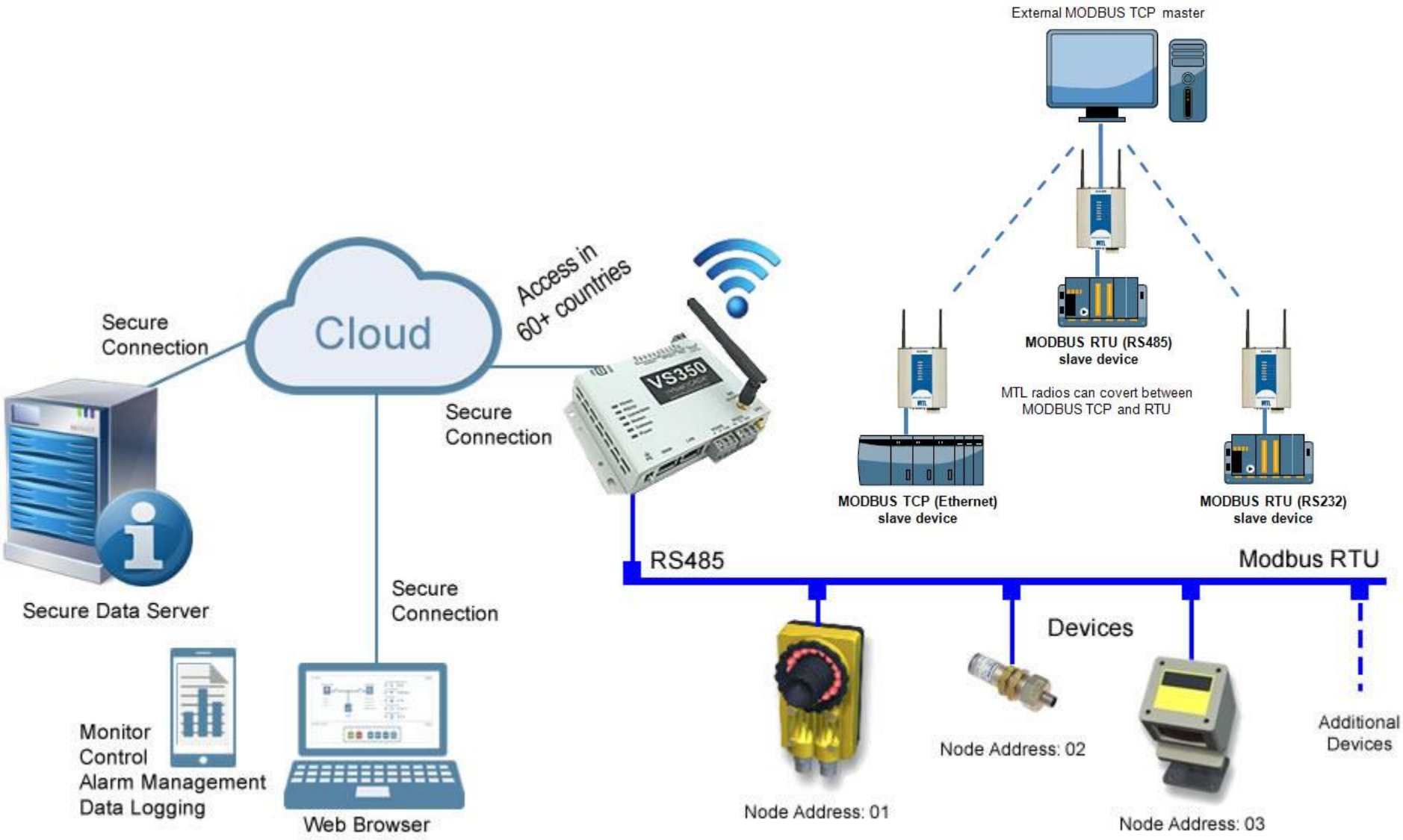
Security Issues with SCADA systems

- Lack of authentication in communication protocols
  - Spoofing
- Security through obscurity
  - Systems are specialized for certain application and cannot be understood unless in knowledge group
- System is isolated physically
  - Since the systems are physically secure with locks and keys they cannot be tampered with



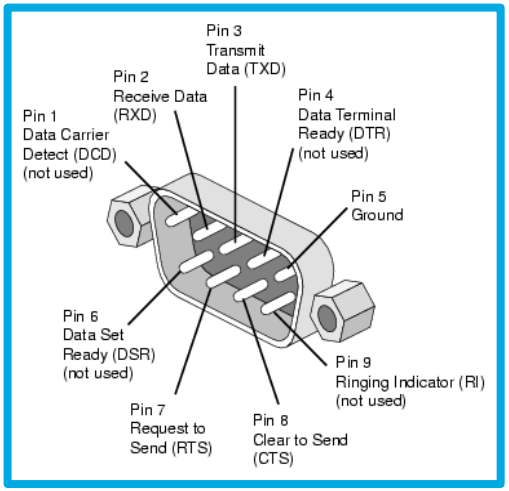
- SCADA systems are also being interconnected with the Internet to allow for increased control and cost savings.

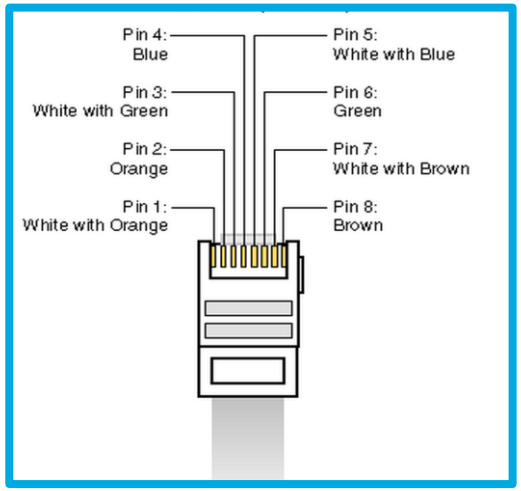
What is SCADA MODBUS?



What is SCADA MODBUS?

- This protocol is used on many Industrial Control Systems (ICSs), specifically SCADA systems.
- Master/Slave Configuration
  - Similar to client/server except slave does not request data, it only receives commands from master.
- Transmitted over serial lines (Modbus RTU/ASCII) or over Ethernet (Modbus TCP).





62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
d=TEDDY&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Windows NT 5.1; SV1; .NET CLR 1.1.4322"
ategory_id=TEDDY" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
listen to your data®

Motivation

Advanced Persistent Threat

- An Advanced Persistent Threat (APT) is a set of stealthy and continuous computer hacking processes in which an unauthorized person gains access to a network and stays there undetected for a long period of time.
- APT attacks target organizations in sectors with high-value information, such as military networks, national defense, manufacturing, financial industry and critical infrastructures.
- **Advanced** – zero days malware.
- **Persistent** – low-and-slow
- **Threat** – the criminal operators have a specific objective and are skilled, motivated, organized and well funded.

Motivation

#RSAC

The Phases of an APT Attack

1. Reconnaissance
Attacker leverages information from a variety of sources to understand their target.

2. Incursion
Attackers break into network by using social engineering to deliver targeted malware to vulnerable systems or by attacking public facing infrastructure.

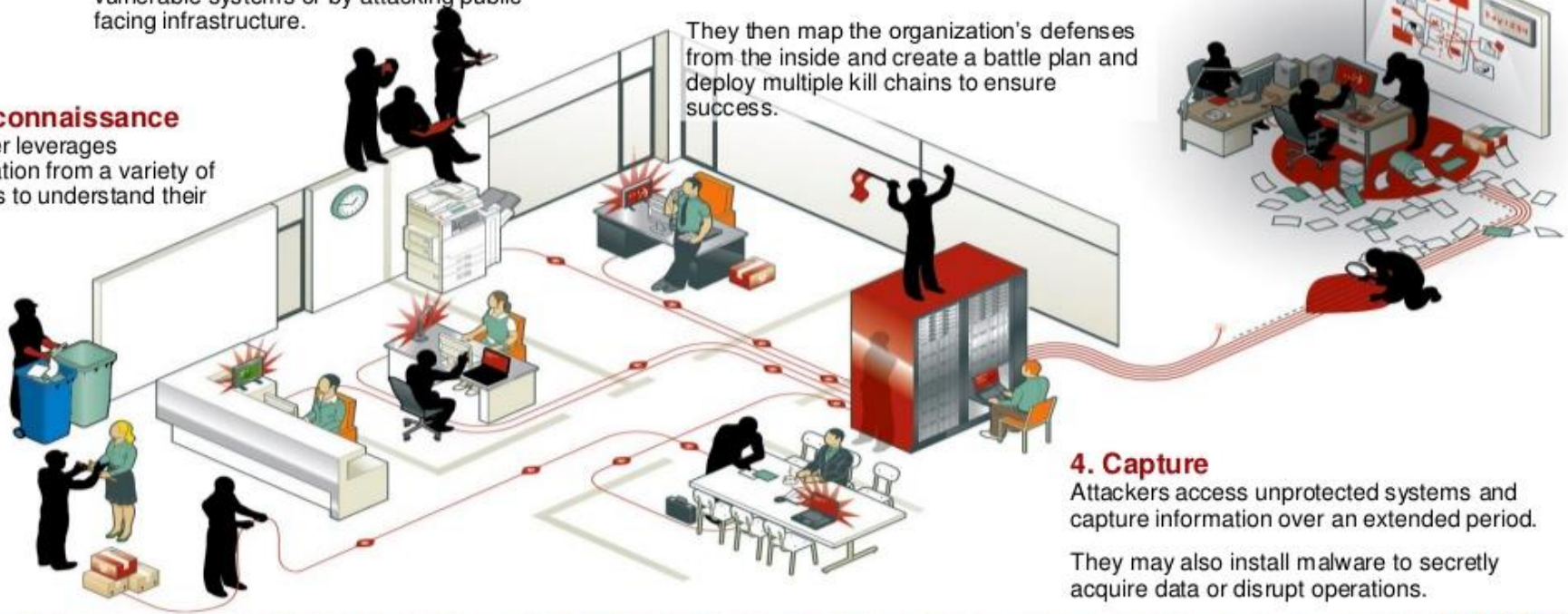
3. Discovery
Once in, the attackers stay "low and slow" to avoid detection.

They then map the organization's defenses from the inside and create a battle plan and deploy multiple kill chains to ensure success.

4. Capture
Attackers access unprotected systems and capture information over an extended period.

They may also install malware to secretly acquire data or disrupt operations.

5. Exfiltration
Captured information is sent back to attack team's home base for analysis and further exploitation.



Symantec

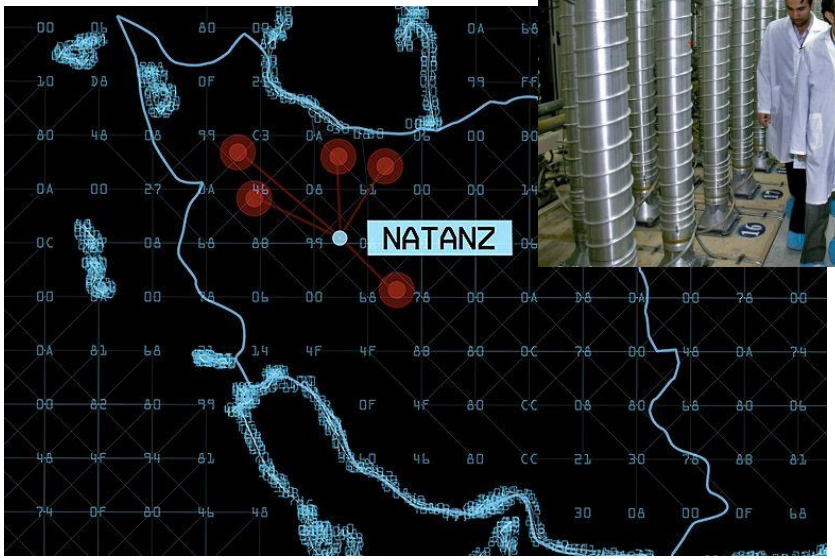
RSAConference2015

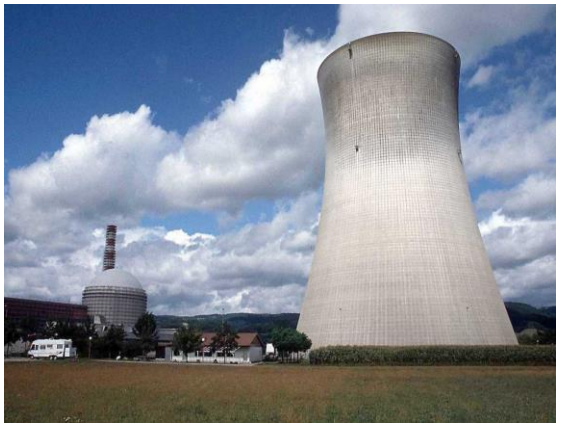
62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"

listen to your data®

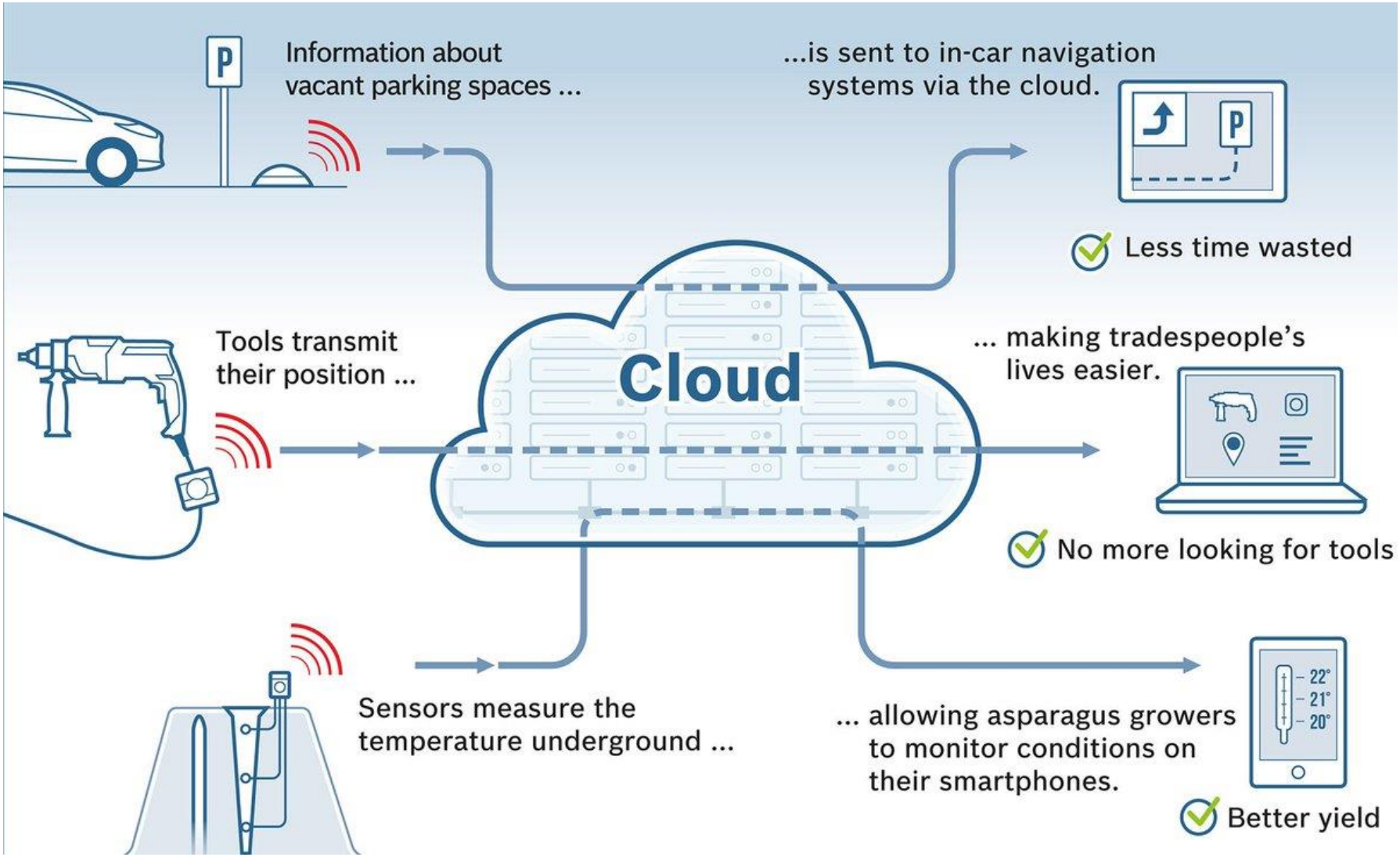
Motivation

- Recent attacks on SCADA systems
  - Stuxnet
  - Davis-Besse Nuclear Plant
  - Maroochy, Australia
  - Flame
  - Aurora

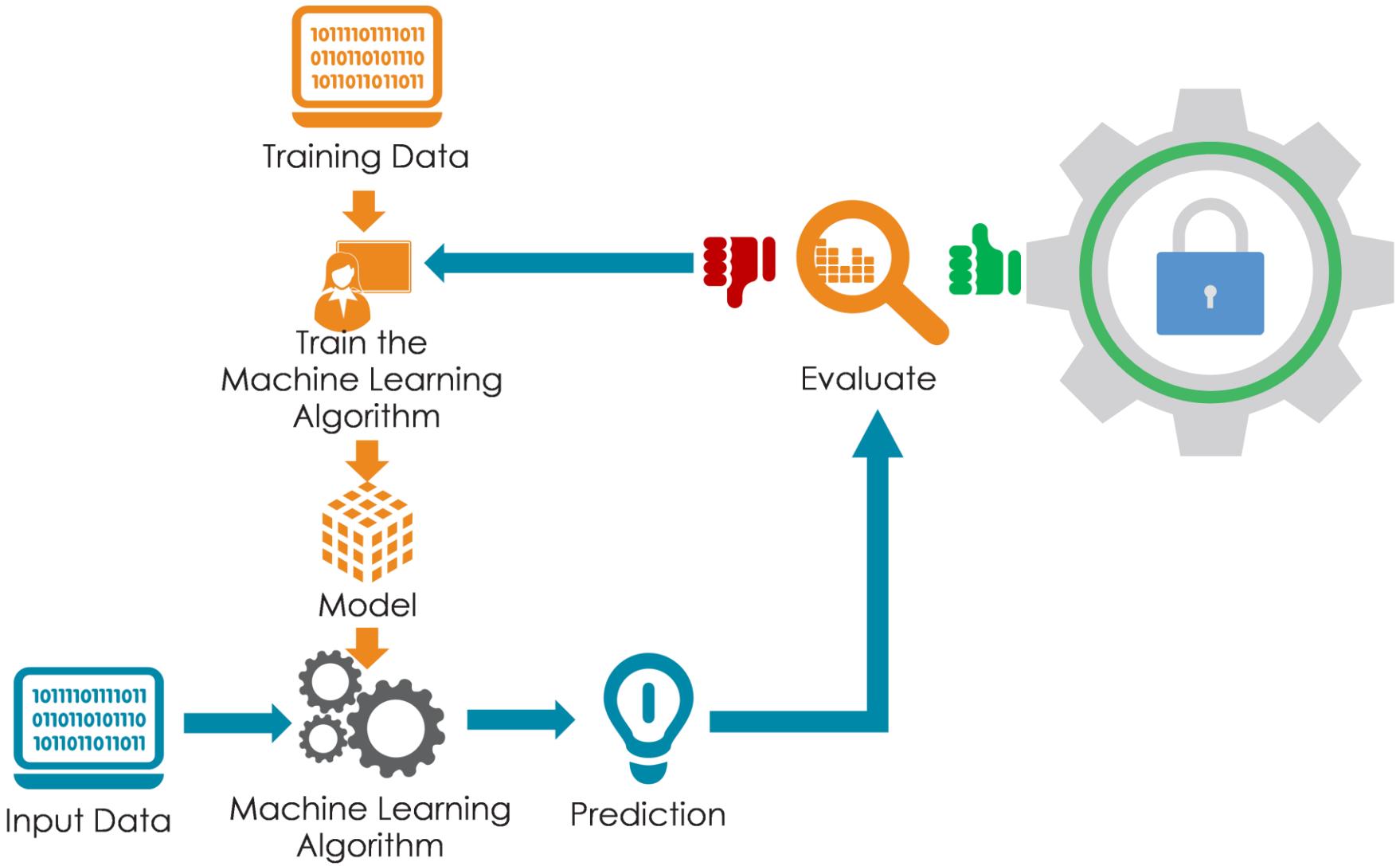




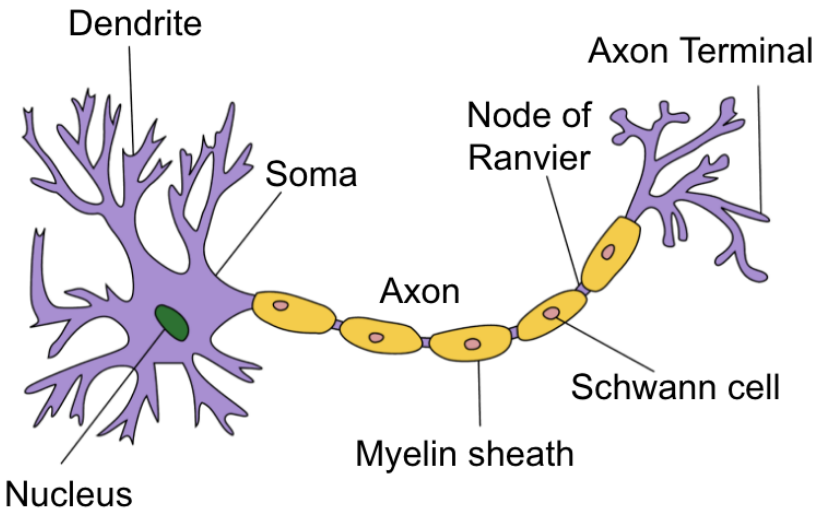
IIoT and Blockchain



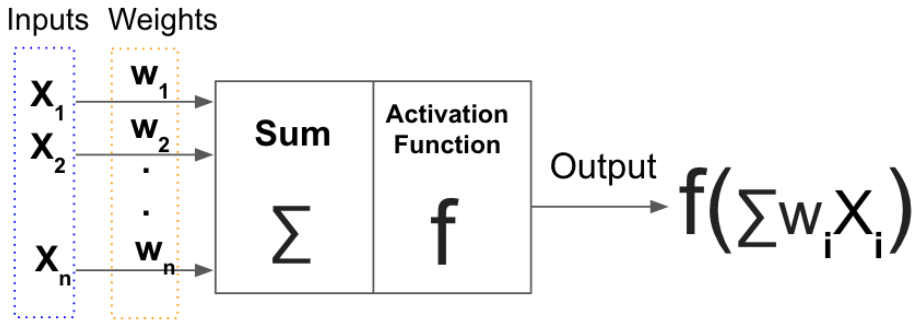
What is Machine Learning?



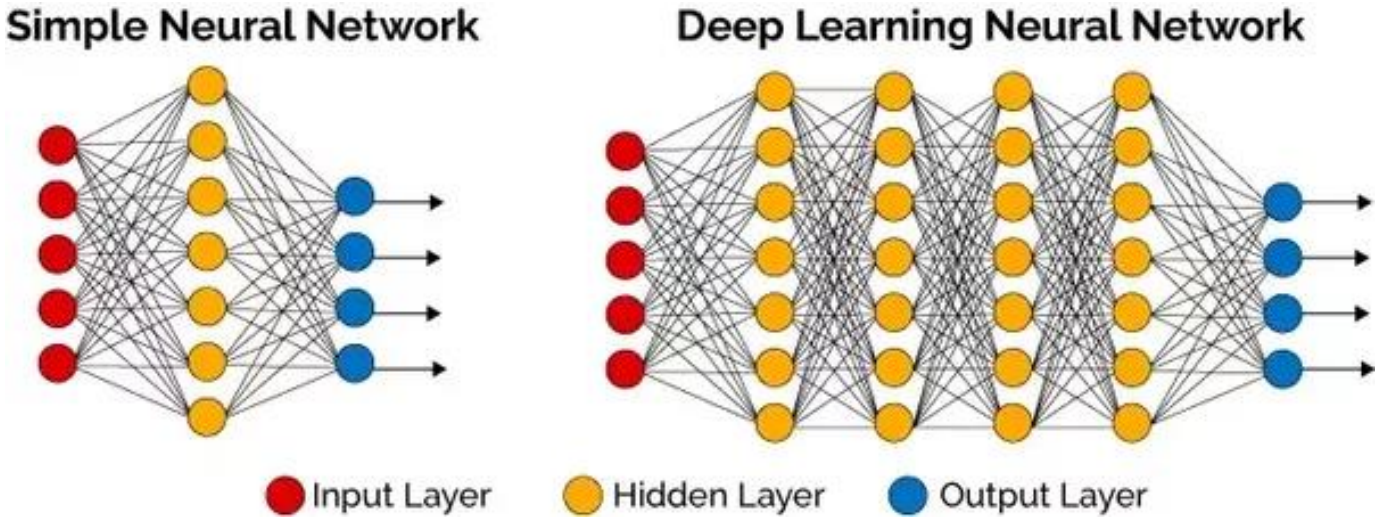
What is Deep Learning?



Structure of a typical neuron



Structure of artificial neuron



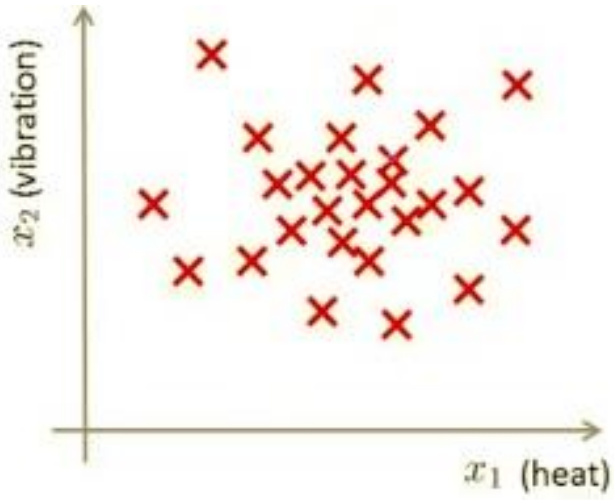
What is Anomaly Detection?

listen to your data®

Example of Anomaly Detection

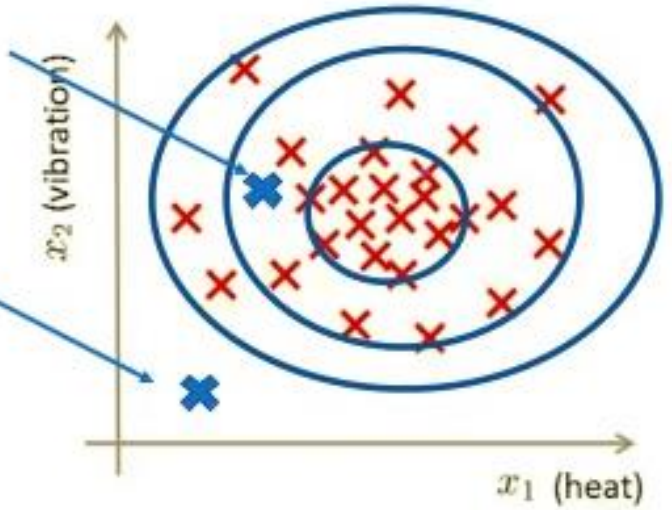
- Density estimation
  - Dataset: { $x^{(1)}$, $x^{(2)}$, $x^{(3)}$, ,,, , $x^{(m)}$}
  - Is "New engine: x_{test}" anomalous?

Model p(x) 에 대하여.

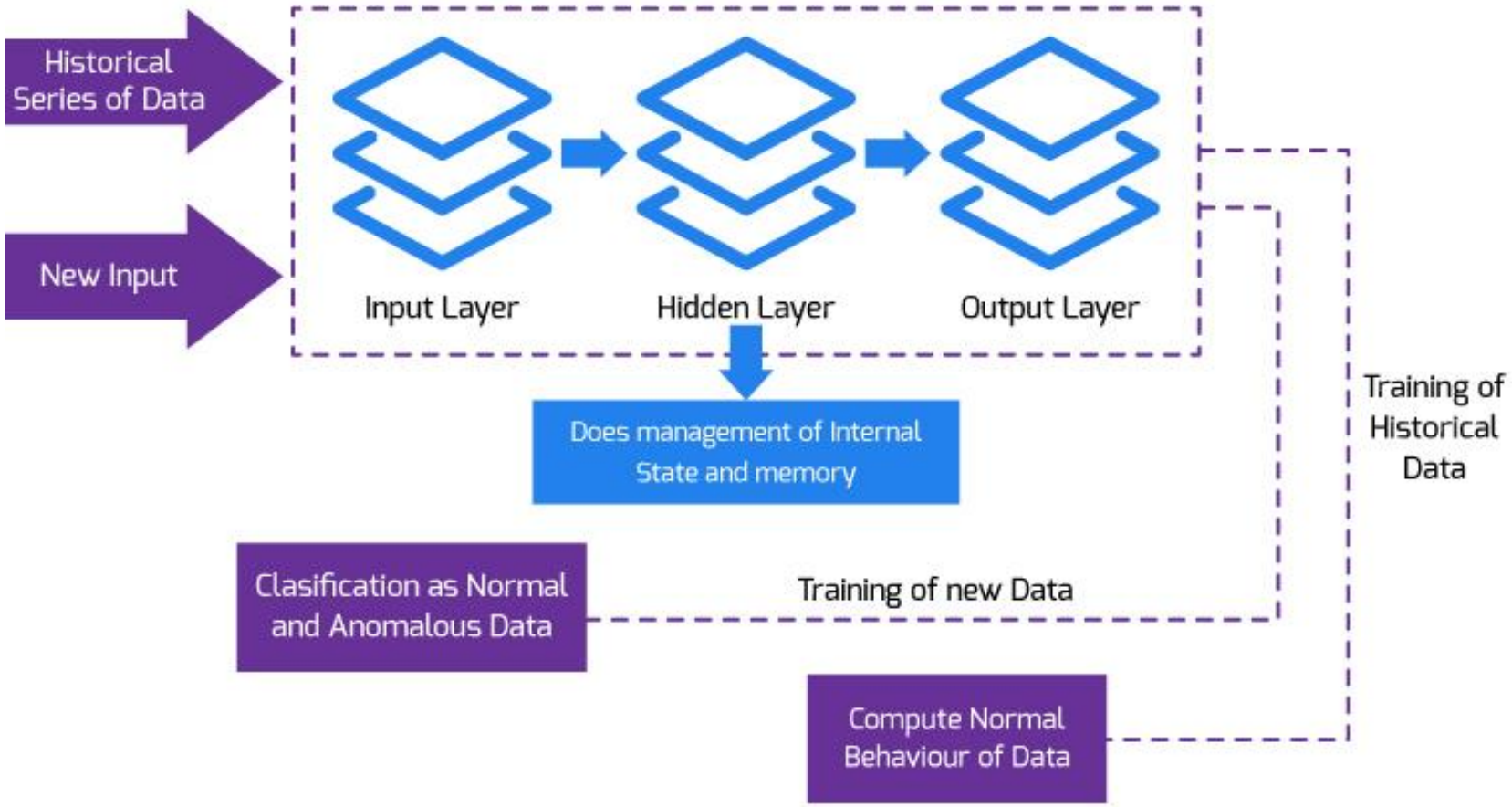


$P(x_{test}) >= E \rightarrow$ not anomaly, normal

$P(x_{test}) < E \rightarrow$ flag anomaly



Anomaly Detection using Deep Learning



Proposed Architecture

chain of things

listen to your data®

Use the right tools for the job

CONCEPTUAL MODEL

Proposed Architecture

Requirements:

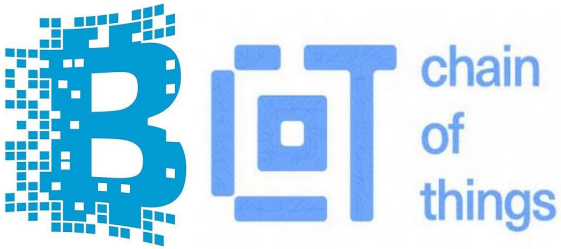
- Security
- Fault tolerance
- Durability
- Public access possibility
- The possibility of consensus

Challenges

- Principle of least privilege
- Transparent Interaction
- Integrated dynamic policy management methodology
- Limited resources

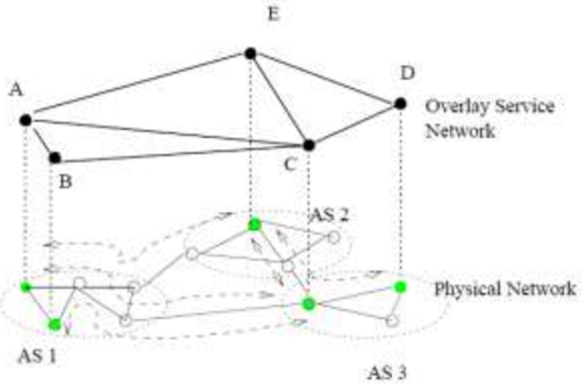
Goals

- Architecture
- Security
- Privacy



Proposed Architecture

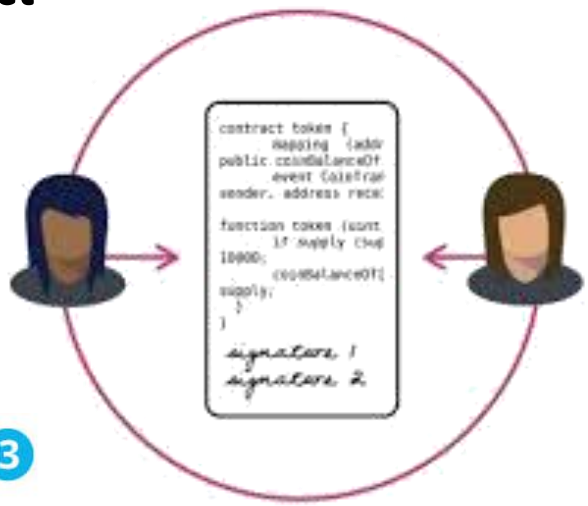
Architecture



- Authorization layer
  - ✓ entities,
  - ✓ namespaces,
  - ✓ resources
  - ✓ Delegations of Trust (DoTs)
    - example: DoT=$\langle A_{pk}^{from}, A_{pk}^{to}, URI_{rsrc}, Permissions, Metadata \rangle$
- Syndication layer
  - ✓ publish permission → “stats” (example: *temp, hum*, etc)
  - ✓ subscribe permission → “cmd” (example: *reset, restart*, etc)
- Overlay layer
  - ✓ nodes in the overlay network can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying network.

Proposed Architecture

The Deep Learning Smart Contract



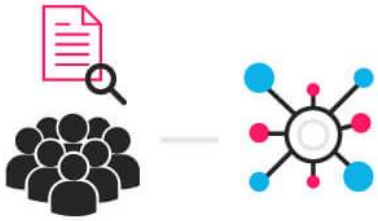
1

An option contact between parties is written as code into the blockchain. The individuals involved are anonymous, but the contact is the public ledger.

2

A triggering event like an expiration date and strike price is hit and the contract executes itself according to the coded terms.

3



Regulators can use the blockchain to understand the activity in the market while maintaining the privacy of individual actors' positions

Autonomy

You're the one making the agreement; there's no need to rely on a broker or lawyer

1

2

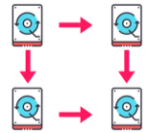
Trust

Your documents are encrypted on a shared ledger

Backup

On the blockchain,Your documents are duplicated many times over

3



Savings

Smart contracts save you money since they knock out the presence of an intermediary

4

Accuracy

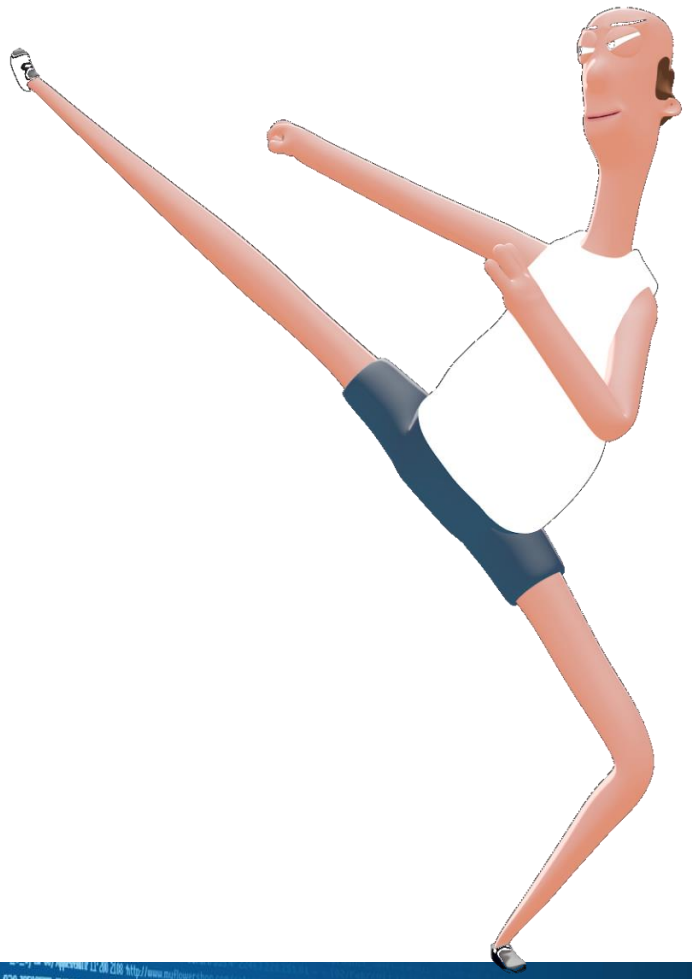
Smart contracts are not only faster and cheaper but also avoid the errors that come from manually filling out heaps of forms.

5

www.Blockgeeks.com

Blockgeeks

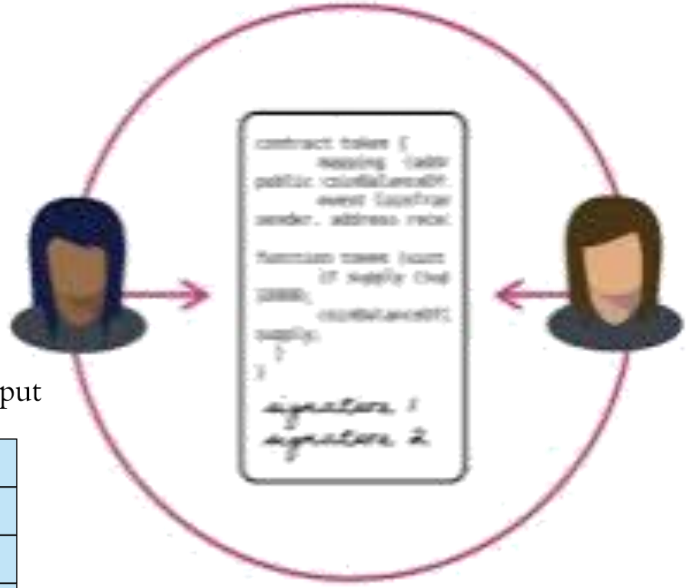
Modeling Methodology

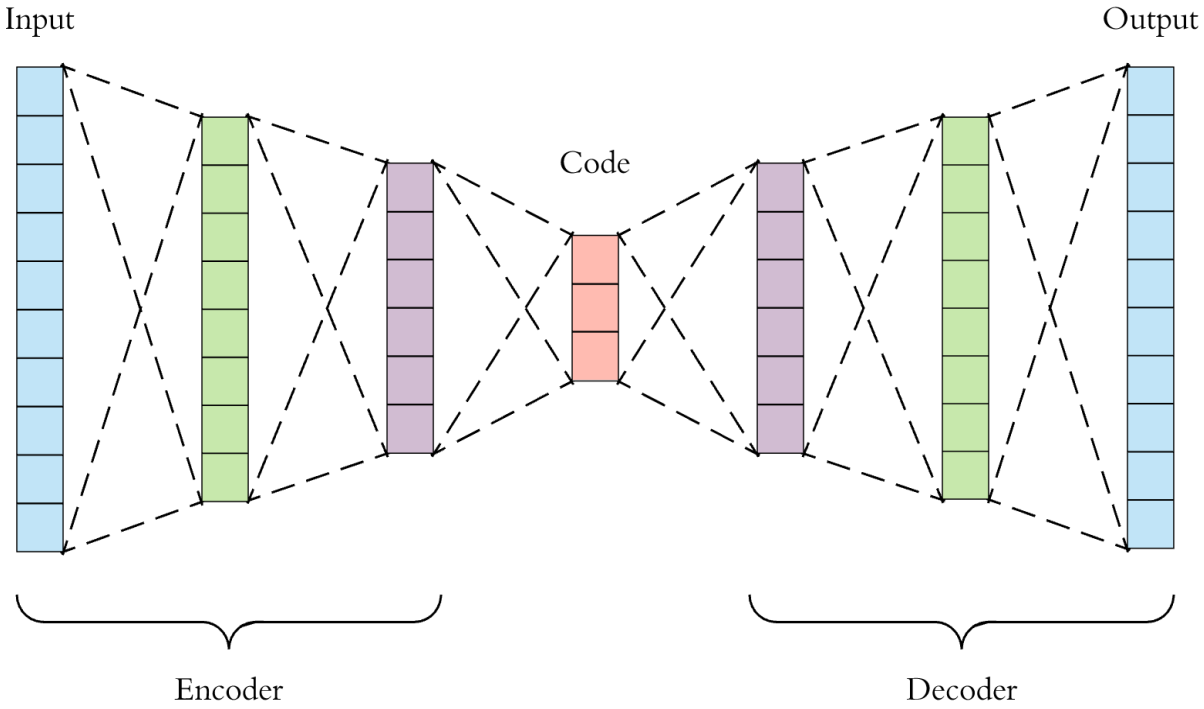


62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "http://www.myflowershop.com/category.screen?category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
d=TEDDY&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Windows NT 5.1; SV1; .NET CLR 1.1.4322)
ategory_id=TEDDY" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)
listen to your data®

Modeling Methodology

The Deep Learning Smart Contract





chain of things

Modeling Methodology

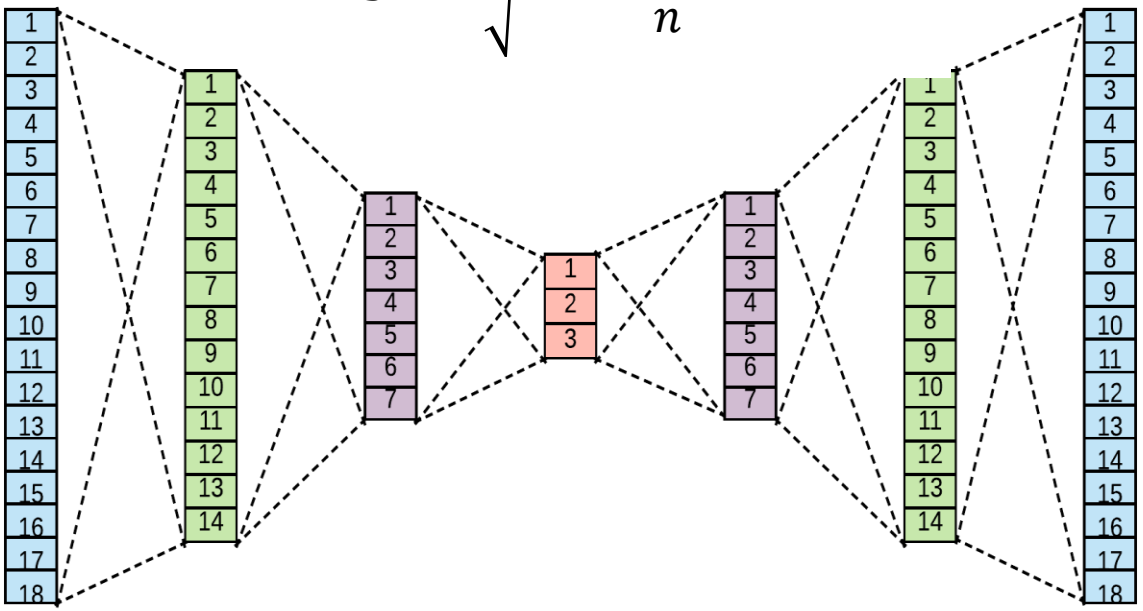
The Deep Learning Smart Contract

- Hyperparameters of Autoencoder
  - ✓ Input – Output size
  - ✓ Code size
  - ✓ Number of layers
  - ✓ Number of nodes per layer
  - ✓ Loss function

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}$$

Lasso Regression or L1 regularization (Least Absolute Shrinkage and Selection Operator) will be used during training:

$$\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|$$



Smart Contract

#MachineAccount

IIoT_thing_thermostat;

#MachineAddress

FE80:0000:0000:0000:0202:B3FF:FE1E:8329;

#MachineInternals

Energy (GeV) = 1.320 | Lifetime (hours) = 9658.150 | Current (mA) = 4.210
Last Refill = 23 May 22:11 | Mode = standby | Feedback Status = On

#MachineStatus

Publisher; Sender;

#DataStream

Size = 125 kb; TimeStamp = 201802305221100;

#DataStreamID

ID = 101;

 # FeaturesOfTrafficFlow

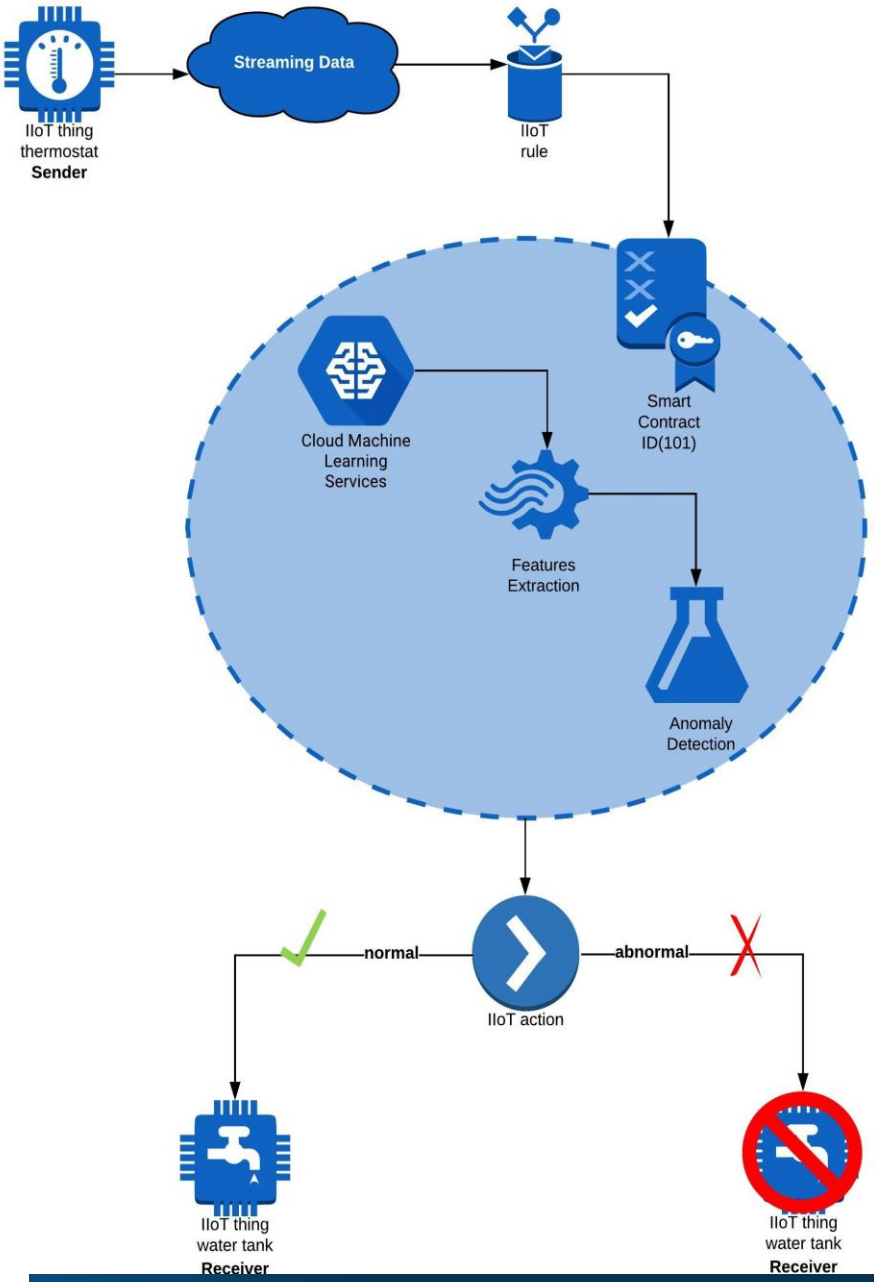
command_address; response_address; command_memory;
response_memory; command_memory_count; response_memory_count;
comm_read_function; comm_write_fun; resp_read_fun; resp_write_fun;
sub_function; command_length; resp_length; control_mode; control_scheme;
pump; crc_rate; measurement;

#Functions

function DataSender (MachineAccount, MachineAddress, MachineStatus);
function DataReceiver (MachineAccount, MachineAddress, MachineStatus);
function DataTransaction (DataStream, DataStreamID);
function DataFlow (TrafficFlow, TrafficFlowID);
function TransactionSession (Session, SessionID);
function *FeaturesOfTrafficFlow;*
function *AnomalyDetection*;

#Loop

start
if *DataSender to DataReceiver a DataTransaction*
then DataFlow process *to FeaturesOfTrafficFlow*
and FeaturesOfTrafficFlow to AnomalyDetection
else if *AnomalyDetection* normal *then TransactionSession*
else send alert to SOC include *DataSender, DataTransaction, DataFlow, FeaturesOfTrafficFlow, AnomalyDetection*
end



“It's not failure, it's data…”

DATA

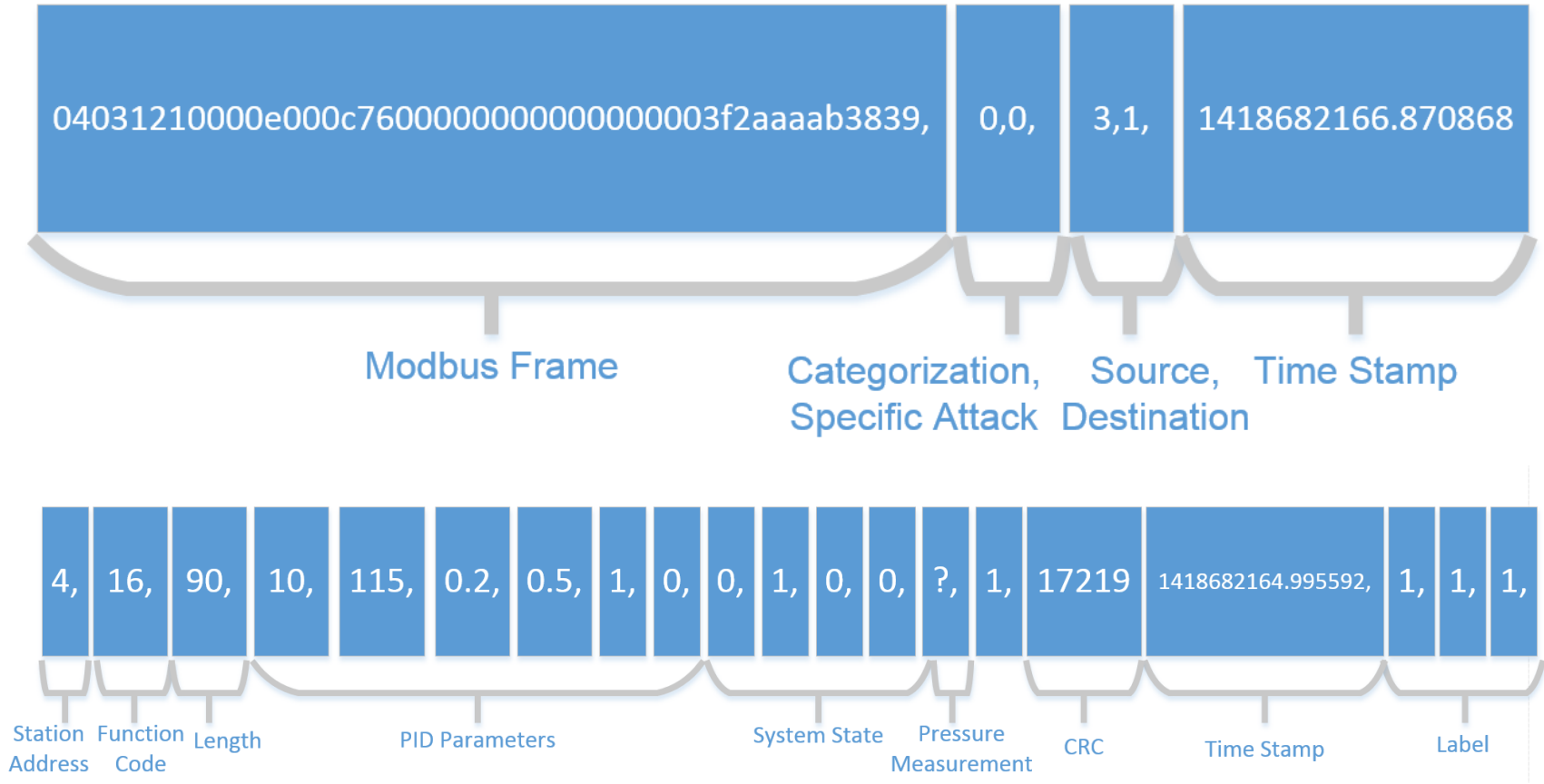
Data

Dataset

- The dataset captured using a network data logger, which monitored and stored MODBUS traffic from a RS-232 connection.
- A bump-in-the-wire was used to capture data logs and to inject attacks.
- The device was implemented via a C program running on a Vmware VM.
- The virtual machine included two RS-232 serial ports connected to a USB-to-serial converter.
- The C program monitored each serial port for traffic.
- Detected traffic was timestamped and recorded in a log file.
- To facilitate attacks, the C program incorporated hooks to inject, delay, drop and alter network traffic.

Data

Dataset



Data

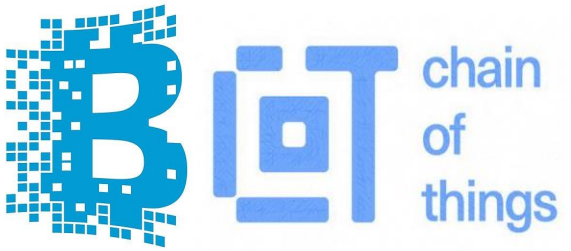
Dataset

- The datasets contain network transactions captured over a serial line
  - Network information
    - Time stamp, Station address, CRC, etc.
  - Payload information
    - System control and state information
  - Label
    - Binary / Category / Specific mode identifier
- Deep packet inspection provides system state information
    - Pressure measurements, pump state, solenoid state, etc.

Feature	Type
address	Network
function	Command Payload
length	Network
setpoint	Command Payload
gain	Command Payload
reset rate	Command Payload
deadband	Command Payload
cycle time	Command Payload
rate	Command Payload
system mode	Command Payload
control scheme	Command Payload
pump	Command Payload
solenoid	Command Payload
pressure measment	Response Payload
crc rate	Network
command response	Network
time	Network
attack	Label

Data

Dataset



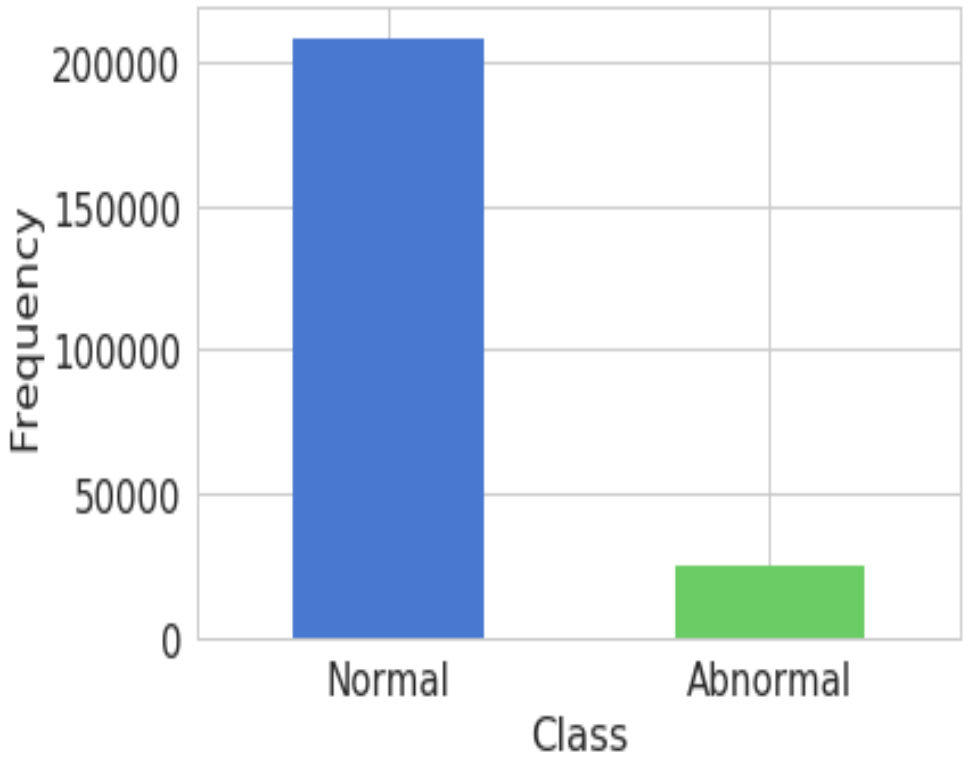
Type of Attacks	Abbreviation
Normal	Normal(0)
Naïve Malicious Response Injection	NMRI(1)
Complex Malicious Response Injection	CMRI(2)
Malicious State Command Injection	MSCI(3)
Malicious Parameter Command Injection	MPCI(4)
Malicious Function Code Injection	MFCI(5)
Denial of Service	DOS(6)
Reconnaissance	Recon(7)

Feature	Type
address	Network
function	Command Payload
length	Network
setpoint	Command Payload
gain	Command Payload
reset rate	Command Payload
deadband	Command Payload
cycle time	Command Payload
rate	Command Payload
system mode	Command Payload
control scheme	Command Payload
pump	Command Payload
solenoid	Command Payload
pressure measment	Response Payload
crc rate	Network
command response	Network
time	Network
attack	Label

Data

Dataset

- The water_dataset includes 18 independent parameters and 233,295 instances, from which 206,761 normal and 26,534 abnormal.



- The dataset is imbalanced.
- A dataset is imbalanced if the distribution of classes is not uniform.
- The problem arising from imbalance is that accuracy can be high for a classifier although it does perform rather bad.
- Resampling Techniques (SMOTE, etc) and Ensemble Techniques (Bagging, Boosting, Adaptive Boosting, etc)

Data

One-class classification by Deep Autoencoder

- The OCC method, tries to find objects of a particular class among all objects, by learning from a training set containing only objects of this class.
- Typically, these algorithms aim to implement classification models in which the negative class is absent.
- This mode of operation in which classifiers are required to determine effectively and reliably the boundaries of the class separation only based on the knowledge of the positive class, is a particularly complex problem of ML. When only data from the target class is available, the classifier is trained to receive target objects and to reject the ones that deviate significantly.
- The basic concept in OCC problem solving is the reverse of the generalization that is being pursued in other ML problems.
- Particularly, it is intended that the parameter setting is fully defined, even if this exponentially increases the complexity of the classifier.

Data

Data pre-Processing

- Duplicate records and records with missing values were removed.
- The datasets were determined and normalized to the interval [-1,1] to phase the problem of prevalence of features with wider range over the ones with a narrower range, without being more important.

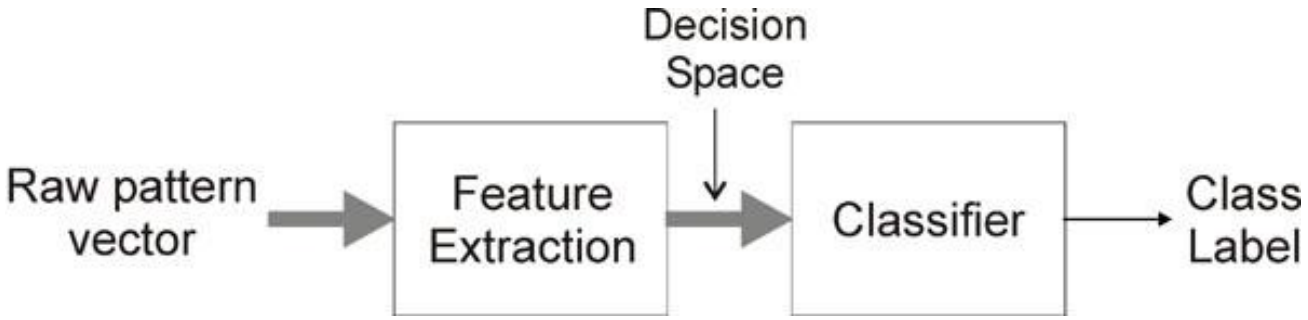
Dataset Threshold Criteria

- Optimal Dataset Threshold

  1: Calculate the error using Euclidean distance between actual and predicted on each training data;

  2: Arrange the error in decreasing order;

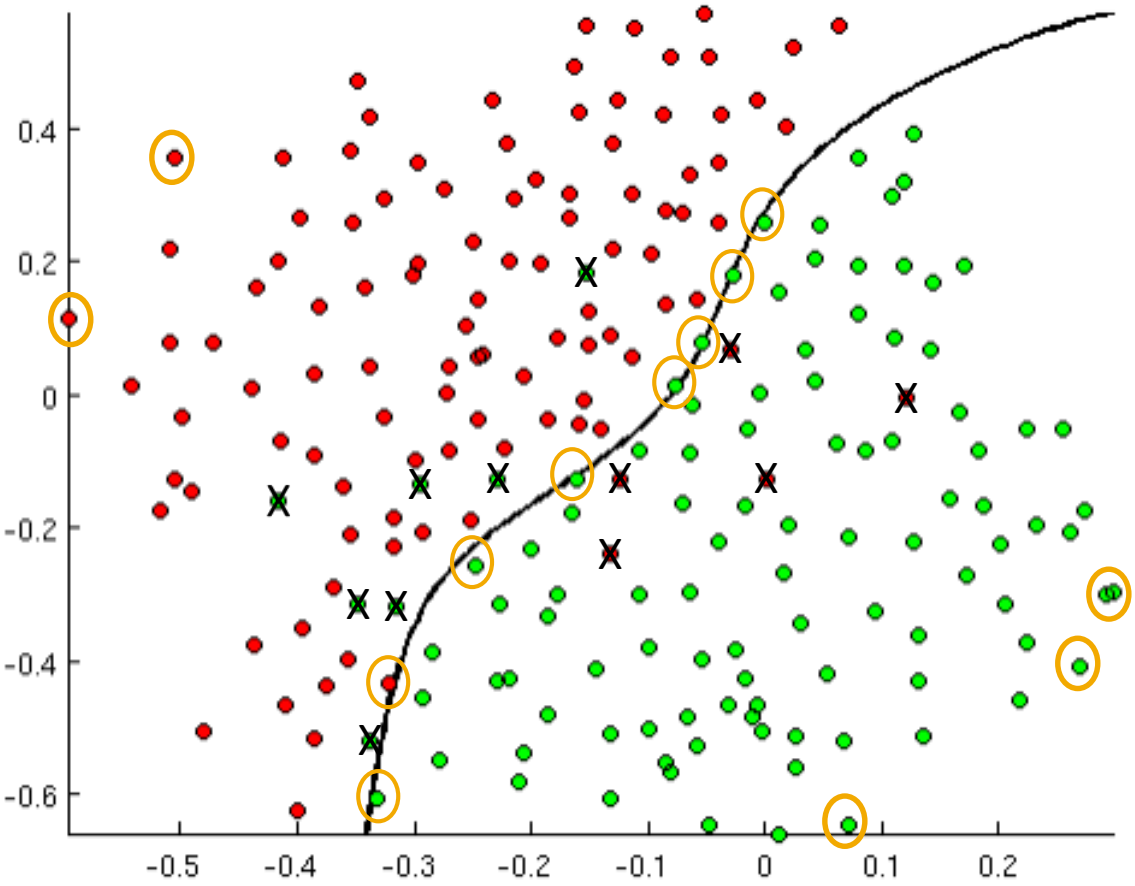
  3: Set the threshold at rejection of 10% most erroneous data (false negative rate at the rate of 10%);



Data

Dataset Threshold Criteria

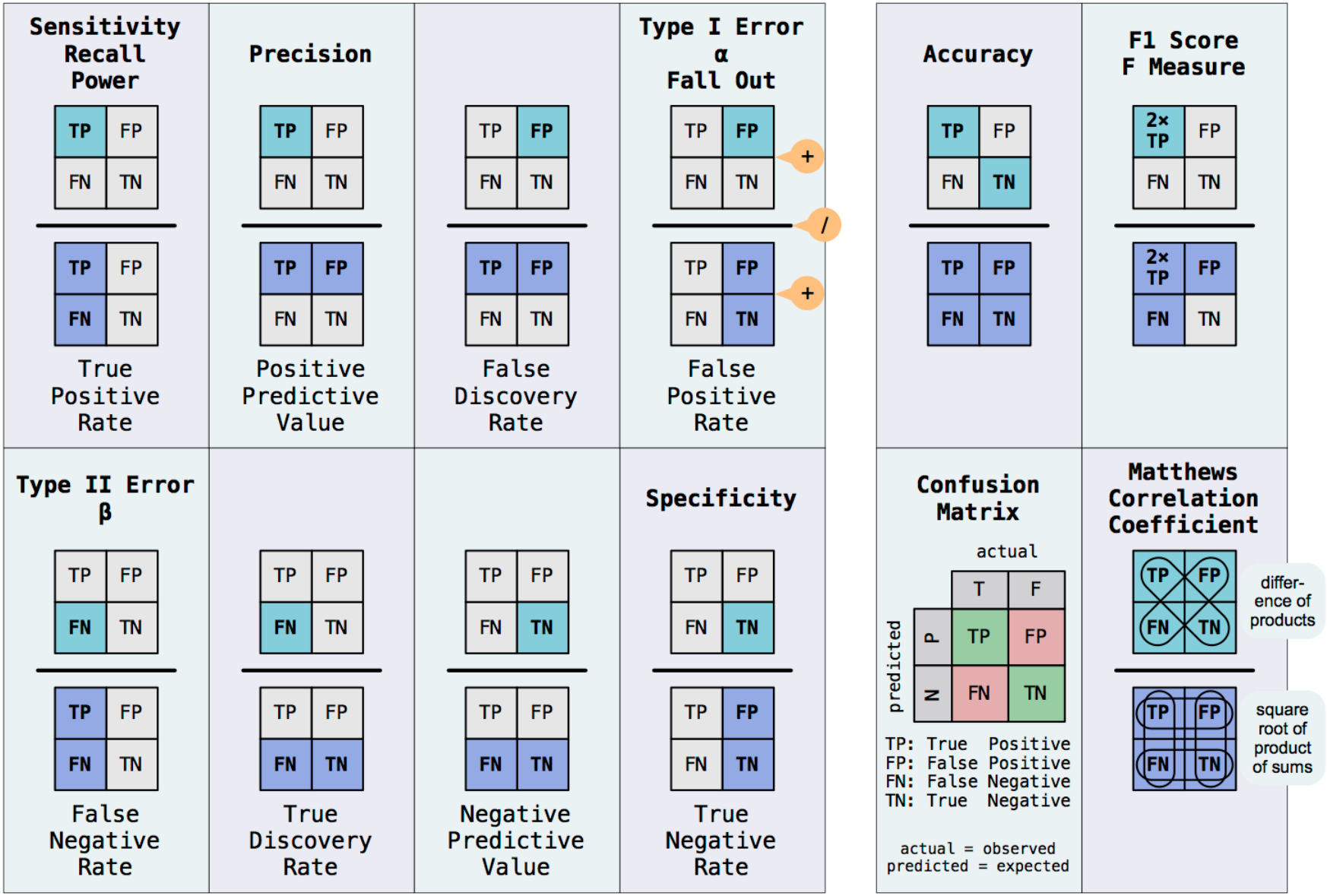
- Optimal Dataset Threshold
- The final water_dataset includes 18 independent parameters and 212,091 instances, from which 185,557 normal and 26,534 abnormal.



“If you know the enemy and know yourself you need not fear the results of a hundred battles…”

RESULTS

Statistical Classification Metrics



62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
d=TEDDY&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"
ategory_id=TEDDY" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)"

Deep Autoencoder

Classification Metrics -> Confusion Matrix

		"Golden Standard" (Real Truth Values)		
		Positive	Negative	
Observed	Predicted positive	True Positive	False Positive (Type 1 error)	Precision
	Predicted Negative	False Negative (Type 2 error)	True Negative	
		Recall/ Sensitivity	(Specificity)	

$$\text{Accuracy} = \frac{TP + TN}{TN + FP + FN + TP}$$

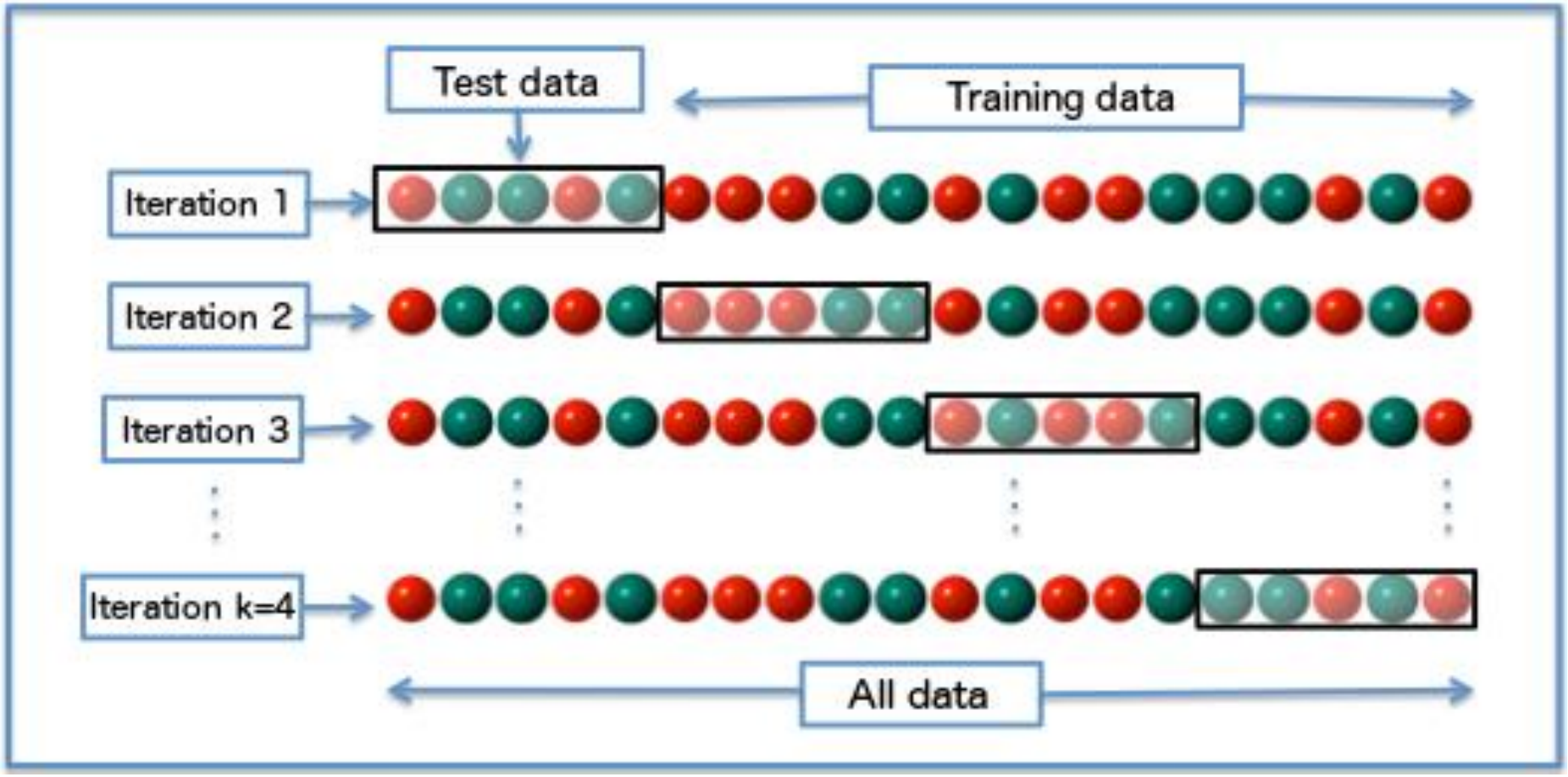
$$\text{Precision} = \frac{TP}{FP + TP}$$

$$\text{Recall} = \frac{TP}{FN + TP}$$

$$\text{F1-score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

Deep Autoencoder

10-Fold Cross Validation



Deep Autoencoder

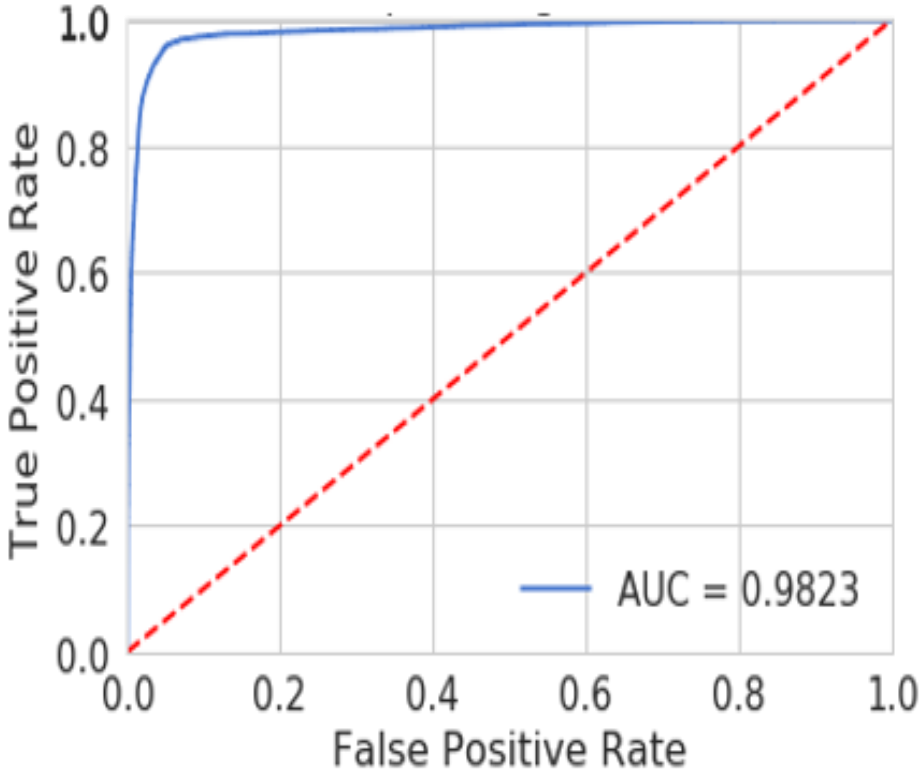
	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**

Classifier	Fold	TA	RMSE	Precision	Recall	F-Score	AUC
Deep Autoencoder	1st	98.30%	0.1129	0.982	0.982	0.982	0.9823
	2nd	98.81%	0.1111	0.992	0.992	0.992	0.9919
	3rd	98.76%	0.1117	0.986	0.986	0.986	0.9870
	4th	98.78%	0.1115	0.988	0.988	0.988	0.9886
	5th	98.69%	0.1119	0.979	0.980	0.980	0.9880
	6th	98.48%	0.1123	0.985	0.985	0.985	0.9860
	7th	**99.03%**	**0.1101**	**0.991**	**0.991**	**0.991**	**0.9901**
	8th	98.80%	0.1112	0.988	0.988	0.988	0.9885
	9th	98.82%	0.1110	0.989	0.988	0.988	0.9886
	10th	98.99%	0.1101	0.989	0.989	0.989	0.9892
	Avg	**98.75%**	**0.1114**	**0.987**	**0.987**	**0.987**	**0.9880**

listen to your data®

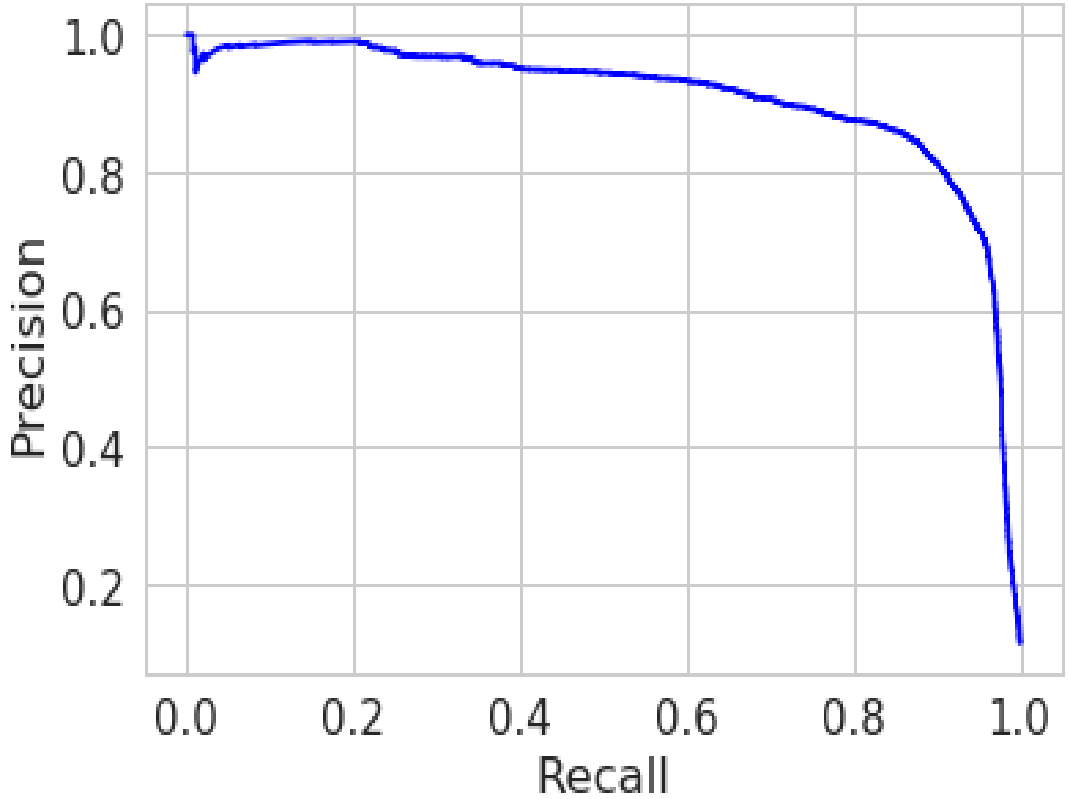
Deep Autoencoder

	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**



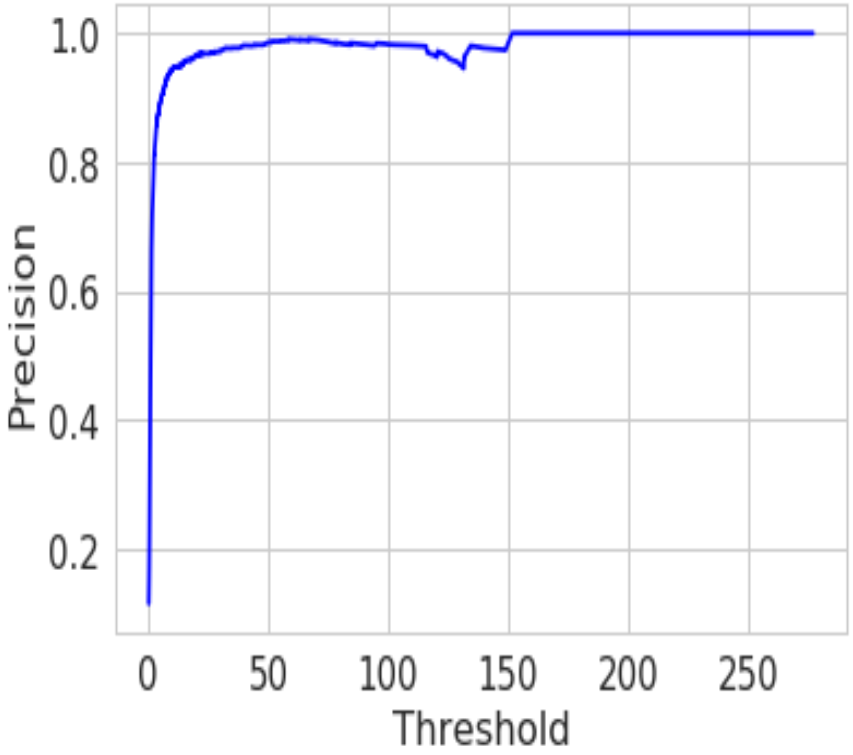
Deep Autoencoder

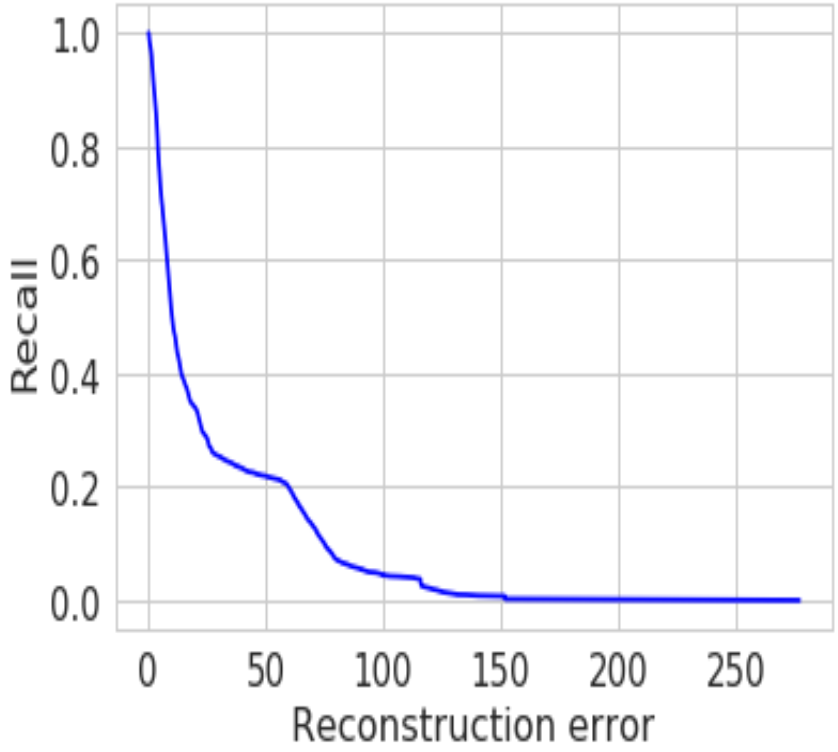
	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**



Deep Autoencoder

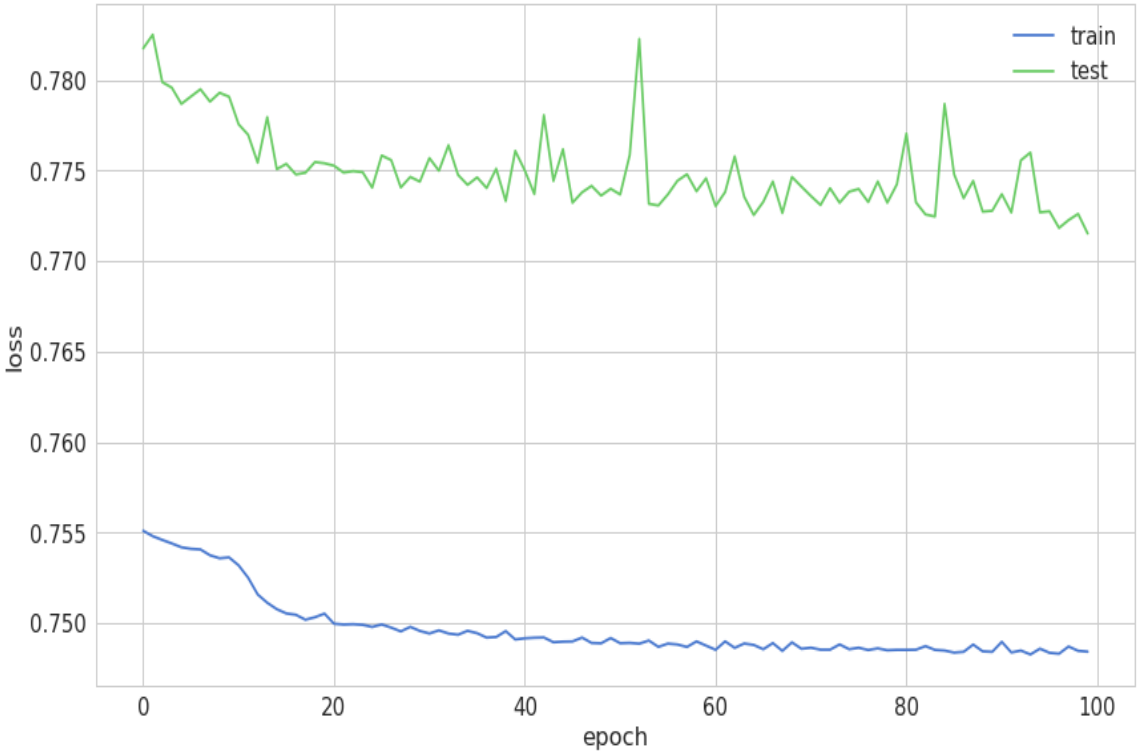
	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**





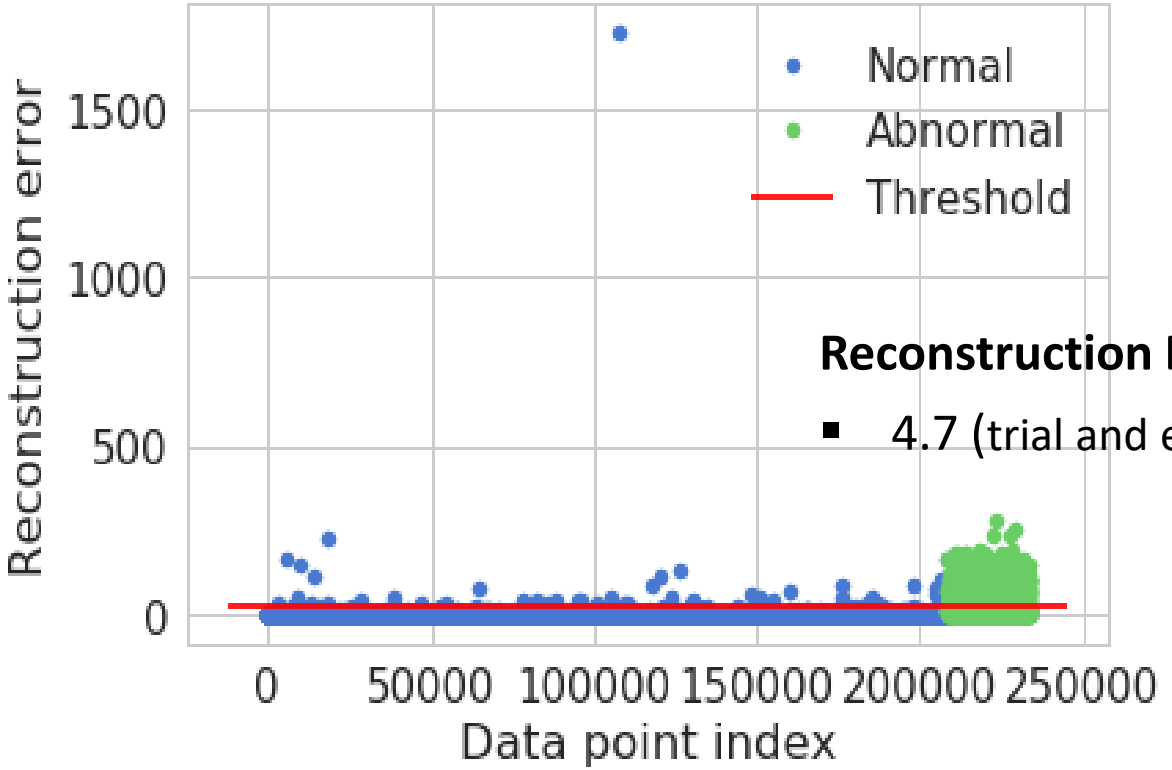
Deep Autoencoder

	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**



Deep Autoencoder

	Normal	Abnormal	
TP	**185.216**	341	**FN**
FP	2.542	**24.201**	**TN**



Reconstruction Error Threshold

- 4.7 (trial and error)

"*The best defense is a good offense*"

APPLICATIONS OF PROPOSED ARCHITECTURE

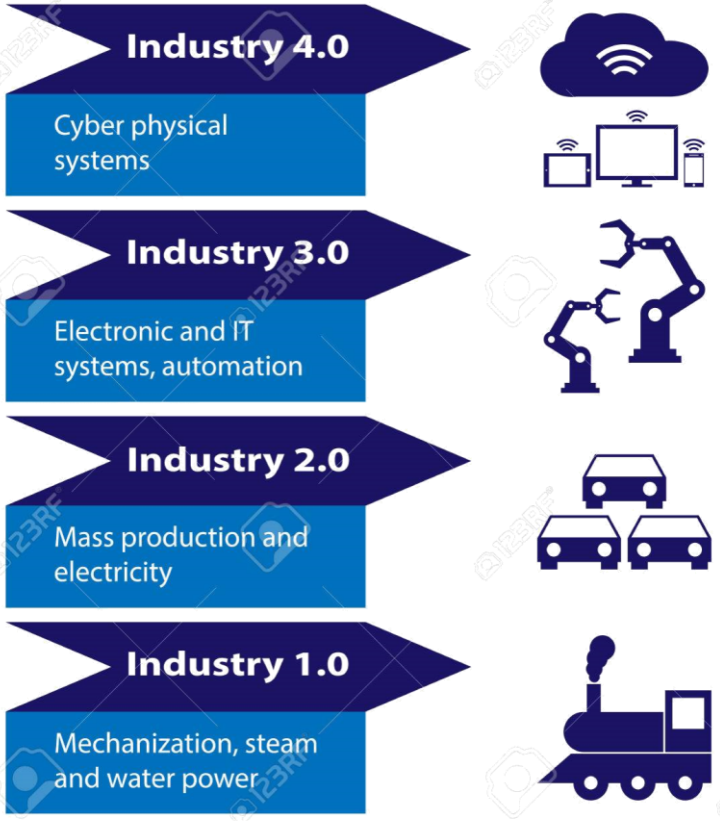
Proposed Architecture

Conceptual Framework Characteristics

- Security
- Privacy
- Quality
- Support
- Bidirectional communication
- Distributed – Decentralized services

Applications

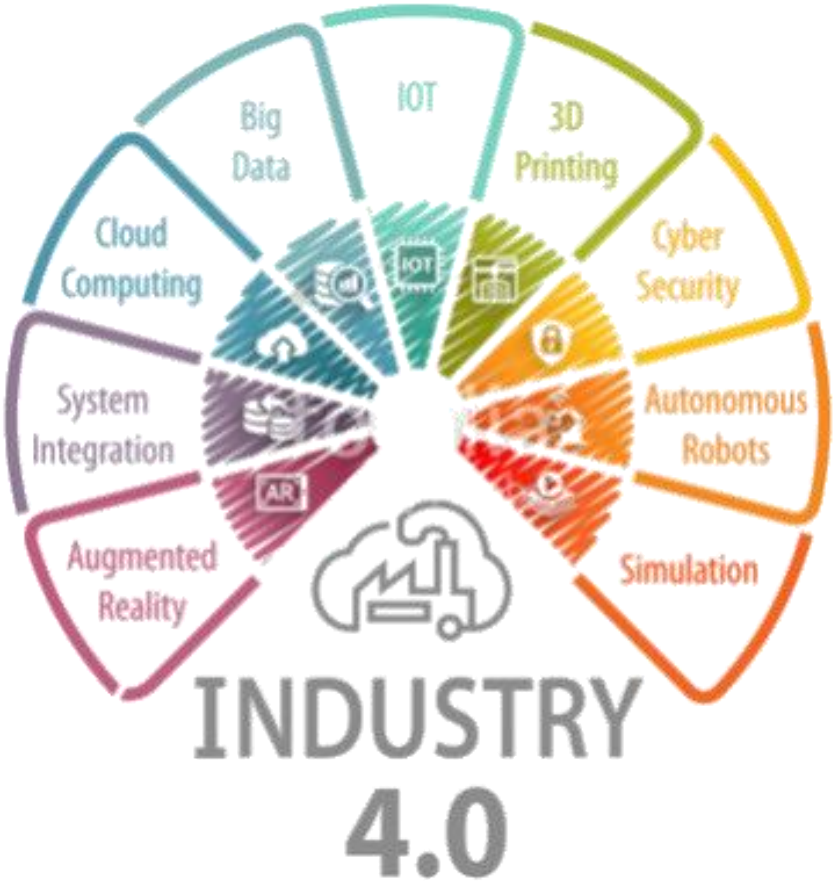
- On-Demand Manufacturing
- Traceability
- Product Certification
- Predictive Manufacturing
- Tracking Supplier Identity & Reputation
- Smart Diagnostics and Machine Maintenance

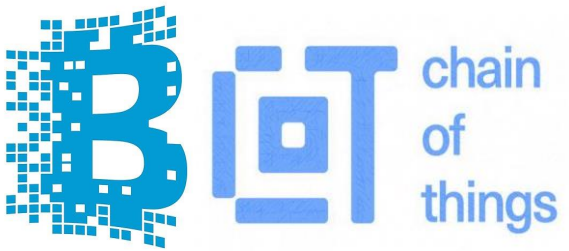


"At this point all the hard work is done"…

DISCUSSION AND FUTURE DIRECTION

Innovation

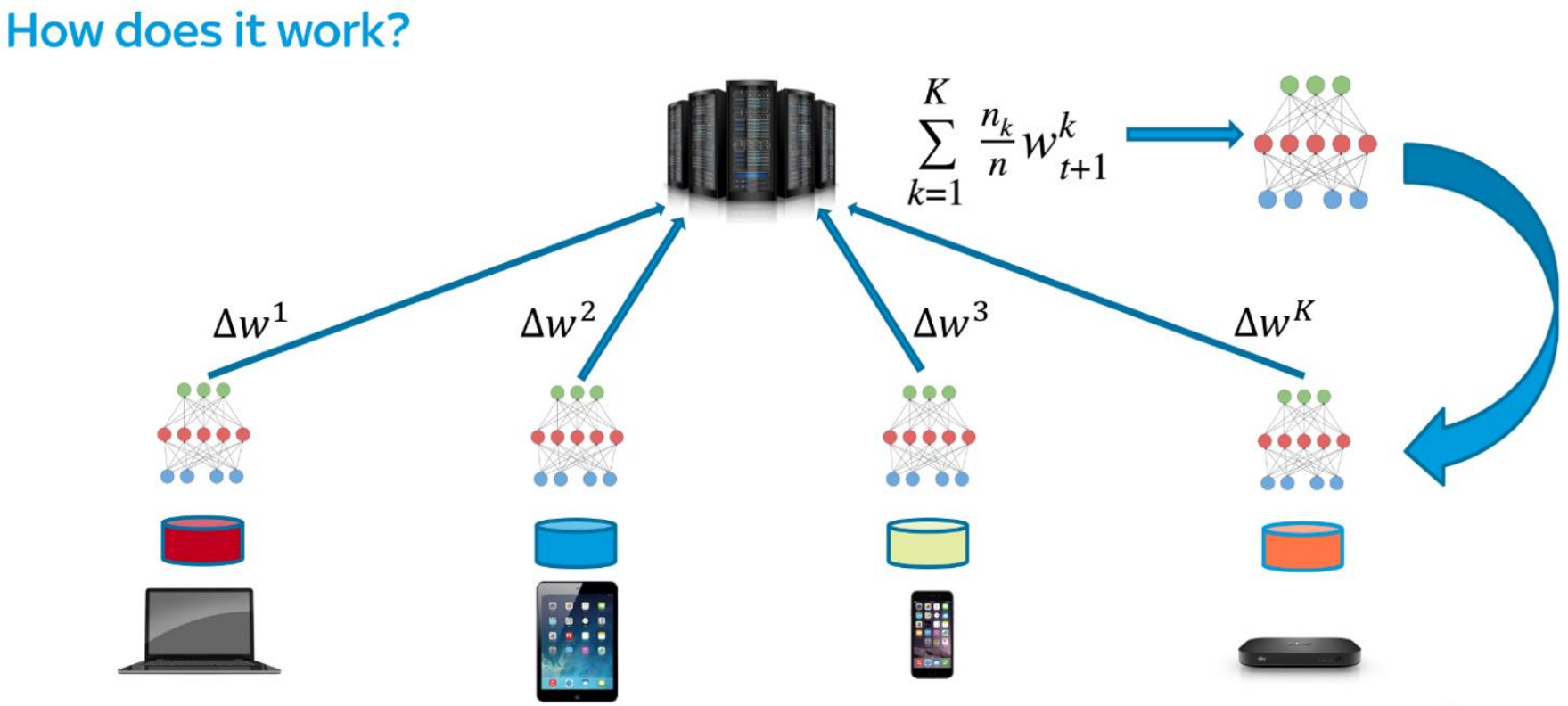




listen to your data®

Future Work

Federated Learning



References

[1] N. Teslya and I. Ryabchikov, "Blockchain-based platform architecture for industrial IoT," 2017 21st Conference of Open Innovations Association (FRUCT), Helsinki, 2017, pp. 321-329. doi: 10.23919/FRUCT.2017.8250199

[2] Arshdeep Bahga, Vijay K. Madisetti, Blockchain Platform for Industrial Internet of Things, Journal of Software Engineering and Applications 09(10):533-546, DOI: 10.4236/jsea.2016.910036

[3] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng and Y. Zhang, "Consortium Blockchain for Secure Energy Trading in Industrial Internet of Things," in IEEE Transactions on Industrial Informatics. doi: 10.1109/TII.2017.2786307

[4] J. Gao et al., "GridMonitoring: Secured Sovereign Blockchain Based Monitoring on Smart Grid," in IEEE Access, vol. 6, pp. 9917-9925, 2018. doi: 10.1109/ACCESS.2018.2806303

[5] M. Mylrea and S. N. G. Gourisetti, "Blockchain for smart grid resilience: Exchanging distributed energy at speed, scale and security," 2017 Resilience Week (RWS), Wilmington, DE, 2017, pp. 18-23. doi: 10.1109/RWEEK.2017.8088642

[6] P. Danzi, M. Angjelichinoski, Č. Stefanović and P. Popovski, "Distributed proportional-fairness control in microgrids via blockchain smart contracts," 2017 IEEE International Conference on Smart Grid Communications (SmartGridComm) pp 45-51.

```
##Loading the libraries
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
from scipy import stats
import tensorflow as tf
import seaborn as sns
from pylab import rcParams
from sklearn.model_selection import train_test_split
from keras.models import Model, load_model
from keras.layers import Input, Dense
from keras.callbacks import ModelCheckpoint, TensorBoard
from keras import regularizers
%matplotlib inline
sns.set(style='whitegrid', palette='muted', font_scale=1.5)
rcParams['figure.figsize'] = 14, 8
RANDOM_SEED = 42
LABELS = ["Normal", "Abnormal"]

##Loading the data
df = pd.read_csv("IIoT3.csv")

##Exploration
df.isnull().values.any()
count_classes = pd.value_counts(df['Class'], sort = True)
count_classes.plot(kind = 'bar', rot=0)
plt.title("Transaction class distribution")
plt.xticks(range(2), LABELS)
plt.xlabel("Class")
plt.ylabel("Frequency");
Abnormal = df[df.Class == 1]
Normal = df[df.Class == 0]

## Data pre-processing
from sklearn.preprocessing import StandardScaler
data = df.drop(['Time'], axis=1)
data['ID'] = StandardScaler().fit_transform(data['ID'].values.reshape(-1, 1))
X_train, X_test = train_test_split(data, test_size=0.2, random_state=RANDOM_SEED)
X_train = X_train[X_train.Class == 0]
X_train = X_train.drop(['Class'], axis=1)
y_test = X_test['Class']
X_test = X_test.drop(['Class'], axis=1)
X_train = X_train.values
X_test = X_test.values
```

listen to your data®

```
##Building the model
input_dim = X_train.shape[1]
encoding_dim = 14
input_layer = Input(shape=(input_dim, ))
encoder = Dense(encoding_dim, activation="tanh",
                activity_regularizer=regularizers.l1(10e-5))(input_layer)
encoder = Dense(int(encoding_dim / 2), activation="relu")(encoder)
code = Dense(int(encoding_dim - 11)
decoder = Dense(int(encoding_dim / 2), activation='tanh')(encoder)
decoder = Dense(input_dim, activation='relu')(decoder)
autoencoder = Model(inputs=input_layer, outputs=decoder)
nb_epoch = 100
batch_size = 32
autoencoder.compile(optimizer='adam',
                    loss='root_mean_squared_error',
                    metrics=['accuracy'])
checkpointer = ModelCheckpoint(filepath="model.h5",
                               verbose=0,
                               save_best_only=True)
tensorboard = TensorBoard(log_dir='./logs',
                          histogram_freq=0,
                          write_graph=True,
                          write_images=True)
history = autoencoder.fit(X_train, X_train,
                    epochs=nb_epoch,
                    batch_size=batch_size,
                    shuffle=True,
                    validation_data=(X_test, X_test),
                    verbose=1,
                    callbacks=[checkpointer, tensorboard]).history
autoencoder = load_model('model.h5')
```

```
##Evaluation
plt.plot(history['loss'])
plt.plot(history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper right');
predictions = autoencoder.predict(X_test)
mse = np.mean(np.power(X_test - predictions, 2), axis=1)
error_df = pd.DataFrame({'reconstruction_error': mse,
                         'true_class': y_test})
error_df.describe()
from sklearn.metrics import (confusion_matrix, precision_recall_curve, auc,
                             roc_curve, recall_score, classification_report, f1_score,
                             precision_recall_fscore_support)
fpr, tpr, thresholds = roc_curve(error_df.true_class, error_df.reconstruction_error)
roc_auc = auc(fpr, tpr)
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, label='AUC = %0.4f'% roc_auc)
plt.legend(loc='lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([-0.001, 1])
plt.ylim([0, 1.001])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show();
precision, recall, th = precision_recall_curve(error_df.true_class, error_df.reconstruction_error)
plt.plot(recall, precision, 'b', label='Precision-Recall curve')
plt.title('Recall vs Precision')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.show()
plt.plot(th, precision[1:], 'b', label='Threshold-Precision curve')
plt.title('Precision for different threshold values')
plt.xlabel('Threshold')
plt.ylabel('Precision')
plt.show()
plt.plot(th, recall[1:], 'b', label='Threshold-Recall curve')
plt.title('Recall for different threshold values')
plt.xlabel('Reconstruction error')
plt.ylabel('Recall')
plt.show()
```

62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 Windows NT 5.1; SV1; .NET CLR 1.1.4322 ... http://www.myflowershop.com/category.screen?category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322 ... http://www.myflowershop.com/category.screen?category_id=TEDDY&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 Windows NT 5.1; SV1; .NET CLR 1.1.4322 ... POST /category.screen?category_id=TEDDY" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322 ... GET /category.screen?category_id=TEDDY"

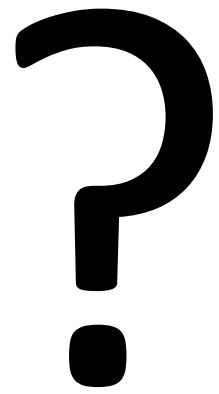
```
##Prediction
threshold = 4.7
groups = error_df.groupby('true_class')
fig, ax = plt.subplots()
for name, group in groups:
    ax.plot(group.index, group.reconstruction_error, marker='o', ms=5.5, linestyle='',
            label= "Abnormal" if name == 1 else "Normal")
ax.hlines(threshold, ax.get_xlim()[0], ax.get_xlim()[1], colors="r", zorder=100, label='Threshold')
ax.legend()
plt.title("Reconstruction error for different classes")
plt.ylabel("Reconstruction error")
plt.xlabel("Data point index")
plt.show();
y_pred = [1 if e > threshold else 0 for e in error_df.reconstruction_error.values]
conf_matrix = confusion_matrix(error_df.true_class, y_pred)
plt.figure(figsize=(12, 12))
sns.heatmap(conf_matrix, xticklabels=LABELS, yticklabels=LABELS, annot=True, fmt="d");
plt.title("Confusion matrix")
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show()
```

No system is safe!!!

62 - - [02/Feb/2011:16:00:23] "GET /product.screen?product_id=FI-FW-02&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 Windows NT 5.1; SV1; .NET CLR 1.1.4322 .NET CLR 2.0.50727; .NET CLR 3.0.04506.30)" "http://www.myflowershop.com/category.screen?category_id=FLOWERS" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30)" 495 /category.screen?category_id=TEDDY&JSESSIONID=SD9SL4FF4ADFF8 HTTP 1.1" 200 3439 Windows NT 5.1; SV1; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30)" "http://www.myflowershop.com/category.screen?category_id=TEDDY" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30)" 495


KEIN
SYSTEM
IST
SICHER
WHOAMI

Q&A



Thanks

My Publications

1. Anezakis, V.-D., Demertzis, K., Iliadis, L., 2018. Classifying with fuzzy chi-square test: The case of invasive species. AIP Conference Proceedings 1978, 290003. https://doi.org/10/gdtm5q
2. Anezakis, V.-D., Demertzis, K., Iliadis, L., Spartalis, S., 2017a. Hybrid intelligent modeling of wild fires risk. Evolving Systems 1–17. https://doi.org/10/gdp863
3. Anezakis, V.-D., Demertzis, K., Iliadis, L., Spartalis, S., 2016a. A Hybrid Soft Computing Approach Producing Robust Forest Fire Risk Indices, in: Artificial Intelligence Applications and Innovations, IFIP Advances in Information and Communication Technology. Presented at the IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, Cham, pp. 191–203. https://doi.org/10.1007/978-3-319-44944-9_17
4. Anezakis, V.-D., Dermetzis, K., Iliadis, L., Spartalis, S., 2016b. Fuzzy Cognitive Maps for Long-Term Prognosis of the Evolution of Atmospheric Pollution, Based on Climate Change Scenarios: The Case of Athens, in: Computational Collective Intelligence, Lecture Notes in Computer Science. Presented at the International Conference on Computational Collective Intelligence, Springer, Cham, pp. 175–186. https://doi.org/10.1007/978-3-319-45243-2_16
5. Anezakis, V.-D., Iliadis, L., Demertzis, K., Mallinis, G., 2017b. Hybrid Soft Computing Analytics of Cardiorespiratory Morbidity and Mortality Risk Due to Air Pollution, in: Information Systems for Crisis Response and Management in Mediterranean Countries, Lecture Notes in Business Information Processing. Presented at the International Conference on Information Systems for Crisis Response and Management in Mediterranean Countries, Springer, Cham, pp. 87–105. https://doi.org/10.1007/978-3-319-67633-3_8
6. Anezakis, V.D., Mallinis, G., Iliadis, L., Demertzis, K., 2018. Soft computing forecasting of cardiovascular and respiratory incidents based on climate change scenarios, in: 2018 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS). Presented at the 2018 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), pp. 1–8. https://doi.org/10.1109/EAIS.2018.8397174
7. Bougoudis, I., Demertzis, K., Iliadis, L., 2016a. Fast and low cost prediction of extreme air pollution values with hybrid unsupervised learning. Integrated Computer-Aided Engineering 23, 115–127. https://doi.org/10/f8dt4t
8. Bougoudis, I., Demertzis, K., Iliadis, L., 2016b. HISYCOL a hybrid computational intelligence system for combined machine learning: the case of air pollution modeling in Athens. Neural Comput & Applic 27, 1191–1206. https://doi.org/10/f8r7vf
9. Bougoudis, I., Demertzis, K., Iliadis, L., Anezakis, V.-D., Papaleonidas, A., 2018. FuSSFFra, a fuzzy semi-supervised forecasting framework: the case of the air pollution in Athens. Neural Computing and Applications 29. https://doi.org/10/gc9bbf
10. Bougoudis, I., Demertzis, K., Iliadis, L., Anezakis, V.-D., Papaleonidas, A., 2016c. Semi-supervised Hybrid Modeling of Atmospheric Pollution in Urban Centers, in: Engineering Applications of Neural Networks, Communications in Computer and Information Science. Presented at the International Conference on Engineering Applications of Neural Networks, Springer, Cham, pp. 51–63. https://doi.org/10.1007/978-3-319-44188-7_4
11. Demertzis, K., Iliadis, L., 2018a. A Computational Intelligence System Identifying Cyber-Attacks on Smart Energy Grids, in: Modern Discrete Mathematics and Analysis,

Springer Optimization and Its Applications. Springer, Cham, pp. 97–116. https://doi.org/10.1007/978-3-319-74325-7_5
12. Demertzis, K., Iliadis, L., 2018b. The Impact of Climate Change on Biodiversity: The Ecological Consequences of Invasive Species in Greece, in: Handbook of Climate Change Communication: Vol. 1, Climate Change Management. Springer, Cham, pp. 15–38. https://doi.org/10.1007/978-3-319-69838-0_2
13. Demertzis, K., Iliadis, L., 2017. Detecting invasive species with a bio-inspired semi-supervised neurocomputing approach: the case of Lagocephalus sceleratus. Neural Computing and Applications 28. https://doi.org/10/gbkgb7
14. Demertzis, K., Iliadis, L., 2016a. Bio-inspired Hybrid Intelligent Method for Detecting Android Malware, in: Knowledge, Information and Creativity Support Systems, Advances in Intelligent Systems and Computing. Springer, Cham, pp. 289–304. https://doi.org/10.1007/978-3-319-27478-2_20
15. Demertzis, K., Iliadis, L., 2016b. Adaptive Elitist Differential Evolution Extreme Learning Machines on Big Data: Intelligent Recognition of Invasive Species, in: Advances in Big Data, Advances in Intelligent Systems and Computing. Presented at the INNS Conference on Big Data, Springer, Cham, pp. 333–345. https://doi.org/10.1007/978-3-319-47898-2_34
16. Demertzis, K., Iliadis, L., 2015a. A Bio-Inspired Hybrid Artificial Intelligence Framework for Cyber Security, in: Computation, Cryptography, and Network Security. Springer, Cham, pp. 161–193. https://doi.org/10.1007/978-3-319-18275-9_7
17. Demertzis, K., Iliadis, L., 2015b. SAME: An Intelligent Anti-malware Extension for Android ART Virtual Machine, in: Computational Collective Intelligence, Lecture Notes in Computer Science. Springer, Cham, pp. 235–245. https://doi.org/10.1007/978-3-319-24306-1_23
18. Demertzis, K., Iliadis, L., 2015c. Evolving Smart URL Filter in a Zone-Based Policy Firewall for Detecting Algorithmically Generated Malicious Domains, in: Statistical Learning and Data Sciences, Lecture Notes in Computer Science. Presented at the International Symposium on Statistical Learning and Data Sciences, Springer, Cham, pp. 223–233. https://doi.org/10.1007/978-3-319-17091-6_17
19. Demertzis, K., Iliadis, L., 2015d. Intelligent Bio-Inspired Detection of Food Borne Pathogen by DNA Barcodes: The Case of Invasive Fish Species Lagocephalus Sceleratus, in: Engineering Applications of Neural Networks, Communications in Computer and Information Science. Presented at the International Conference on Engineering Applications of Neural Networks, Springer, Cham, pp. 89–99. https://doi.org/10.1007/978-3-319-23983-5_9
20. Demertzis, K., Iliadis, L., 2014. Evolving Computational Intelligence System for Malware Detection, in: Advanced Information Systems Engineering Workshops, Lecture Notes in Business Information Processing. Presented at the International Conference on Advanced Information Systems Engineering, Springer, Cham, pp. 322–334. https://doi.org/10.1007/978-3-319-07869-4_30
21. Demertzis, K., Iliadis, L., 2013. A Hybrid Network Anomaly and Intrusion Detection Approach Based on Evolving Spiking Neural Network Classification, in: E-Democracy, Security, Privacy and Trust in a Digital World, Communications in Computer and Information Science. Presented at the International Conference on e-Democracy, Springer, Cham, pp. 11–23. https://doi.org/10.1007/978-3-319-11710-2_2

22. Demertzis, Konstantinos, Iliadis, L., Anezakis, V.-D., 2017a. Commentary: Aedes albopictus and Aedes japonicus—two invasive mosquito species with different temperature niches in Europe. Front. Environ. Sci. 5. https://doi.org/10/gdp865
23. Demertzis, Kostantinos, Iliadis, L., Avramidis, S., El-Kassaby, Y.A., 2017. Machine learning use in predicting interior spruce wood density utilizing progeny test information. Neural Comput & Applic 28, 505–519. https://doi.org/10/gdp86z
24. Demertzis, Konstantinos, Iliadis, L., Spartalis, S., 2017b. A Spiking One-Class Anomaly Detection Framework for Cyber-Security on Industrial Control Systems, in: Engineering Applications of Neural Networks, Communications in Computer and Information Science. Presented at the International Conference on Engineering Applications of Neural Networks, Springer, Cham, pp. 122–134. https://doi.org/10.1007/978-3-319-65172-9_11
25. Demertzis, K., Iliadis, L.S., Anezakis, V.-D., 2018a. An innovative soft computing system for smart energy grids cybersecurity. Advances in Building Energy Research 12, 3–24. https://doi.org/10/gdp862
26. Demertzis, K., Iliadis, L.S., Anezakis, V.-D., 2018b. Extreme deep learning in biosecurity: the case of machine hearing for marine species identification. Journal of Information and Telecommunication 0, 1–19. https://doi.org/10/gdwszn
27. Dimou, V., Anezakis, V.-D., Demertzis, K., Iliadis, L., 2018. Comparative analysis of exhaust emissions caused by chainsaws with soft computing and statistical approaches. Int. J. Environ. Sci. Technol. 15, 1597–1608. https://doi.org/10/gdp864
28. Anezakis, VD., Demertzis, K., Iliadis, L. et al. Evolving Systems (2017). https://doi.org/10.1007/s12530-017-9196-6, Hybrid intelligent modeling of wild fires risk, Springer.
29. Demertzis K., Anezakis VD., Iliadis L., Spartalis S. (2018) Temporal Modeling of Invasive Species' Migration in Greece from Neighboring Countries Using Fuzzy Cognitive Maps. In: Iliadis L., Maglogiannis I., Plagianakos V. (eds) Artificial Intelligence Applications and Innovations. AIAI 2018. IFIP Advances in Information and Communication Technology, vol 519. Springer, Cham.
30. Konstantinos Rantos, George Drosatos, Konstantinos Demertzis, Christos Ilioudis and Alexandros Papanikolaou. Blockchain-based Consents Management for Personal Data Processing in the IoT Ecosystem. In proceedings of the 15th International Conference on Security and Cryptography (SECRYPT 2018), part of ICETE, pages 572-577, SCITEPRESS, Porto, Portugal, 26-28 July 2018.

View publication stats