Smart Factory Accelerator - ThingsBoard

~~~~~~~~~~~~~~~~~~~~~~~~

Smart-Factory-Accelerator is a real-time analytics and predictive maintenance solution built on top of ThingsBoard and QlikSense, enabled by AI and is distributed

under the Apache license.

~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~

DESCRIPTION

- With the emergence of Fourth Industrial Revolution, or Industry 4.0, a smart factory has turned into a reality. The development of technology to collect and

analyze data from sensors in real time provides the ability to monitor and proactively resolve issues that come up in the production process. It has become possible

to extremely easily keep track of the performance levels of the equipment well before it breaks down or technical failure, causing a hindrance in the overall

productivity of the factory. The Smart Factory Accelerator (SFA) is designed to initiate a result oriented approach that facilitates prevention of any possible

collapse in the machinery that would cause a negative impact on production. The solution aims to put in place a monitoring system that analyzes key causes of failure

so that factories can make Zero Downtime a reality.

- Using The Smart Factory Accelerator, the sensors, devices, people and process become a part of the connected ecosystem, where it is employed. All the relevant

data is aggregated, analyzed and thus acted upon. As modern factories are populated with complex and expensive equipment, it becomes essential to get a clear picture

of what is happening in the factory without any delay. Therefore, identifying bottlenecks in processing, taking proactive steps to deal with changing situations, and

increasing awareness of the operational system are key to sensor-based monitoring. The salient features of Smart Factory Accelerator are its IoT integration with

ThingsBoard and real-time alerting system with alert notification to employee’s Telegram Bot as well as on device manager of ThingsBoard, providing real time updates

of any machine malfunction to the operator to initiate a corrective measure.

~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~

Table of Contents

------------

1. Description

2. Technical Functionality

3. Installation

1. Dependencies

2. Setup Guide

4. Model Details

5. Help

~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~

TECHNICAL FUNCTIONALITY

The Smart Factory Accelerator is an Industry 4.0 compatible solution, providing :-

\* IoT integration with ThingsBoard using MQTT publish/subscribe protocol

\* Provisioning devices, assets and define relations between them

\* Analyze incoming telemetry and trigger Real-time Alerting mechanism

\* Rule Chain Development for specific functionalities

\* Predictive maintenance and Anomaly Detection for Smart Factories using Kafka and Advanced Analytics

\* Design dynamic and responsive Dashboards using customizable widgets

\* Managing users and assigning entities to relevant customers

\* Device Authentication and Security

\* ThingsBoard integration with Telegram Bot for alert notifications on smartphone/ Desktop version

~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~

System Requirements:

• 4 core CPU

• 16GB RAM

• Ubuntu Server 18.04 LTS/ Windows 10 Operating System

~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~

Technical Dependencies

~~~~~~~~~~~~~~~~~~~~~~~~~

SFA requires:

- Kafka

- MQTT Broker

- Java 8 (OpenJDK)

- ThingsBoard Community Edition v3.0.1

- Jetbrains PyCharm Community Edition 2019.2.2

- PostgreSQL v11.7

- Telegram Desktop/ Mobile application

- Python3.6:

-paho-mqtt Package (v1.5.0)

-pandas Package (v1.0.4)

-numpy Package (v1.18.5)

-kafka Package (v1.3.5)

-keras Package (v2.3.1)

-tensorflow Package (v2.2.0)

-logging Module

-sys Module

-json Module

-time Module

-requests Package (v2.23.0)

-scikit-learn Package (v0.20.3)

-h5py Package (v2.10.0)

-seaborn Package (v0.10.1)

-matplotlib Package (v2.3.1)

-ast Module

Installation Help -

-Kafka:

Installation link of Kafka:

<https://kafka.apache.org/quickstart>

-MQTT broker:

Installation link of MQTT:

<https://mosquitto.org/download/>

-Python3.6:

Installation link of Python3.6:

<https://www.python.org/downloads/release/python-368/>

-PostgreSQL:

Installation link of PostgreSQL:

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

-ThingsBoard:

Installation link of ThingsBoard CE:

<https://github.com/thingsboard/thingsboard/releases/download/v3.0.1/thingsboard-windows-3.0.1.zip>

-Java OpenJDK:

Installation link of Java 8 OpenJDK:

<https://adoptopenjdk.net/index.html>

~~~~~~~~~~~~

Setup Guide

~~~~~~~~~~~~

-Kafka:

For Linux:

Step 1: Start zookeeper-server:

> bin/zookeeper-server-start.sh config/zookeeper.properties

Start the Kafka server:

> bin/kafka-server-start.sh config/server.properties

Step 2: Create Topic:

Create a topic named "sensors" or "{any-name-of-your-choice}" with a single partition and only one replica:

> bin/kafka-topics.sh --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic sensors

For Windows:

On Windows platforms use bin\windows\ instead of bin/, and change the script extension to .bat

Step1: Start zookeeper-server:

>kafka\_2.12-2.0.0\bin\windows\zookeeper-server-start.bat kafka\_2.12-2.0.0\config\zookeeper.properties

Start the Kafka server:

> kafka\_2.12-2.0.0\bin\windows\kafka-server-start.bat kafka\_2.12-2.0.0\config\server.properties

Step2: Create Topic:

Create a topic named "sensors" or "{any-name-of-your-choice}" with a single partition and only one replica:

> kafka\_2.12-2.0.0\bin\windows\kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic sensors

~~~~~~~~~~~~

~~~~~~~~~~~~

-MQTT Broker:

For Windows:

Step 1: Open the command prompt (run as administrator)

Step 2: Run the following command:

1st Command: net start mosquitto

2nd Command: sc query mosquito (check status)

~~~~~~~~~~~~

~~~~~~~~~~~~

-Python-Script:

I. First, go to the datamart\_service and set your configurations in config.json file and set the following details:

a. http :

• host

• port

• token (Thingsboard device token)

b. Misc :

• interval\_ms

c. Data:

• file\_path (CSV file path)

After setting up the config file, start datamart\_driver.py file.

II. Now go into the dataadapter\_service and set your configurations in config.json file and set following details:

a. mqtt :

• host

• port

• topic (in our case it’s “rawmessage1” and "rawmessage2").

b. Kafka :

• host

• port

• topic (topic name that was created on the Kafka server, in our case “sensors-u1m1” and "RULsensors").

After setting up the config file, start raw\_message\_intercept.py file.

III. Now go to ml\_service and set your configurations in config.json file and set the following details:

a. http :

• host

• port

• token (Thingsboard device token)

• sleeptime (refresh time for data)

b. Kafka :

• host

• port

• topic (topic name that was created on Kafka, in our case it’s “sensors-u1m1” and "RULsensors")

After setting up the config file, start ml\_driver.py file.

~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~~

Telegram Mobile Application Account setup:

- Install Telegram Application on your Mobile

- Search for BotFather (Telegram Bot)

- Enter "/help" to look for commands for creating and setting up Telegram Bot

- Enter "/newbot" command to create a new bot and assign a name for your Bot

- After creation of your Bot, an Authentication token is generated which will be used in the REST Api Call Endpoint URL feature.

- Go to ThingsBoard Rule Engine and add a script which contains the Telegram message body to be sent (message name, device name, sensor value) to Telegram Bot.

- Get the ‘Chat ID’ of Telegram bot by sending a message to the bot. Refresh the web browser URL link of the bot for viewing the message schema, and search for

‘id’ key inside ‘chat’ parent key.

- In Rule Engine, add external-type Rule node of ‘REST API Call’ after the script node and enter the endpoint URL containing the authentication token, with

Request type as POST and header as ‘content-type’ with value ‘application/json’. (Authentication URL has been omitted for security reasons)

- Check notifications on your smartphone in Telegram app.

~~~~~~~~~~~~~~~~~~~~~~~~~~

~~~~~~~~~~~~~~~~~~~~~~~~~~

MODEL DETAILS :-

Anomaly Detection:

- Details:

Model: LSTM-Autoencoder

Framework used: Keras (with Tensorflow as backend)

Python Modules: scikit-learn, pandas, numpy, json, logging, seaborn, matplotlib, math

Data Normalization metric: MinMaxScaler

Hidden Layers: 5 (excluding input layer)

Activation function: ReLU (Rectifier Linear Unit - used on all layers )

Optimizer: Adam

Loss function: MAE (Mean Absolute error)

Epochs: 100

Batch Size:10

Validation\_split: 0.05 (while training)

Threshold (for calculating Anomaly check): 0.25 (obtained through distribution plot of Loss function value MAE - extremities of a curve towards the right)

~~~~~~~~~~~~~~~~~~~~~~~~~~

Help and Support

------------

For any queries reach out to us at smaccoe@teamcomputers.com