Prototype: Machine analyser - User manual

Erik Lindgren & Niklas Allard

erik.lindgren95@gmail.com*, *niklas.allard96@hotmail.com

Linköping university, 2019

Requirements

These needs to be installed to use the prototype:

- Python 3.6.5 or later
- Scikit-learn, Panda & Numpy
- DB Browser for SQLITE

About

The completeness of a machine's description is defined by all the components it consists of, implying that the description for each component in the structure needs to be analysed to determine if a machine is complete. The prototype summarizes the data set as described in section 3.4 in *Exploring unsupervised anomaly detection in Bill of Materials structures* to represent the components in the structure. In this summarized data set each unique *Equipment number* represents a component and is iterated by the prototype to determine if the description of each component in the structure is complete.

In order to determine if a component is complete, the prototype needed a set of values to compare a components features against. Since the references did not exist, the prototype generates these references by using the algorithms to analyse and compare components with identical *Article number.* After each time a reference is generated or updated, the current component features is compared to the reference's allowed range in the reference library. Based on the amount of features that matches its corresponding allowed range, a quality-measurement is created. This quality-measurement could then be used to calculate how complete a component description is. This process was done to each component extracted from the machine data set. When each component in the structure had been iterated, the percentage of how complete the description for that machine was calculated based on the quality of the nodes.

For more details, read the provided pdf *Exploring unsupervised anomaly detection in Bill of Materials structures.*

Usage

The prototype needs a excel file representing a machine or other "data" represented as a BOM structure which needs to be analysed. A data-pool containing a multitude of components/machines is also needed, either as a excel or another format. If another format is used the *application.py* file needs to be altered.

It is recommended to reduce the content of the data-pool so that it only contains components relevant to the machine, i.e. only components with *Material no.* which exists in the machine. This greatly reduces the execution time. This can either be done in the *runapp.py* file with the function *generate datapool,* keep in mind that this functions also summarizes the data-pool. Before the prototype can used the data-pool needs to be summarized in order for the prototype to iterate and use the components available. This can also be done *runapp.py* file with the function *generate datapool.*

The prototype uses a local database to store references. This database needs to be initialized with the provided schema, *schema.sql.*

To run the prototype, place the excel files which will be used in the same folder as *application.py* and run the *runapp.py* file. This will generate references in the database and a excel file *result.x* containing the results of the analysis.

Configurations

The features used can either be replaced or expanded with more feature. To change the features, *application.py, database_helper.py* and *schema.sql* needs to be altered with the new features to ensure that the right features are extracted and stored in the reference library. Follow the comment available in the files.

The algorithms and the parameters used can also be altered or replaced with other methods. The algorithm and parameters are changed in the *generate_reference* function in *application.py.* Follow the comments available.