Knowledge-based pattern recognition and visualization of error logs of time-based engine sensor data: Requirements engineering and tool-support

Viet Tiep Do, 27 May 2015

Software Engineering for Business Information Systems (sebis)
Department of Informatics
Technische Universität München, Germany

www.matthes.in.tum.de
“Knowledge-based pattern recognition and visualization of error logs of time-based engine sensor data: Requirements engineering and tool-support”

<table>
<thead>
<tr>
<th>1.</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.</td>
<td>Concept</td>
</tr>
<tr>
<td>3.</td>
<td>Tool-support</td>
</tr>
<tr>
<td>4.</td>
<td>Results</td>
</tr>
</tbody>
</table>
Industrial partner

IAV GmbH
Ingenieurgesellschaft Auto und Verkehr

- **Type**: Limited liability company: GmbH
- **Industry**: Automotive industry
- **Founded**: Berlin, Germany (1983)
- **Founder**: Prof. Dr. Hermann Appel
- **Headquarters**: Berlin, Germany
- **Number of locations**: 16 operations across Germany, several Subsidiaries
- **Area served**: Worldwide
- **Key people**: Kurt Blumenröder, President, CEO; Michael Schubert, President, CFO
- **Services**: Powertrain, Electronics and Vehicle Development
- **Revenue**: 595 million Euro (2013)
- **Number of employees**: 5,700 worldwide (2013)
- **Website**: IAV.com
Error logs of time-based engine sensor data

Series engines → Field test → Error data → Supervision and support of series engine

- Error analysis
- Root-Cause identification
- Error-handling procedure

Engine Development
Context of engine error data

- On board – data logger in Engine Control Unit, generates error data files (MDF-Format).
- Different channels are continuously measured. When a registered Event is detected, these measured channels - in a short time interval around the Event - will be cut out and permanently saved in ECU.
- Error data files consist of dynamic (time-based) and static measured values.

• Several causes could lead to one Event. A cause is called an engine error pattern, which composes of several conditions.
• Error data files helps to identify Root-Cause.

- Engine start temperature: [70]
- Mileage: [1234]
- Gear number: [3]
- Current speed: [130]
- ... 

- Round speed: [1000; 900; 800; 700 ..., 0]
- Time_RoundSpeed: [-3; -2; -1; 0; 1; 2; 3]
- Cylinder pressure: [20,2; 19,8; 12,3; 15,5;...]
- Time_CylinderPressure: [-1,0; -0,5; 0; 0,5;..] 
- ...
Event: Engine goes down!

Starter’s round speed, engine doesn’t start

Event happened at time 0
Air mass = 4 kg/h points to closed intake valve
Error pattern discovery

Research Question: Which phases does a knowledge-based pattern discovery process contain?

General pattern recognition/discovery: Speech recognition, optical character recognition, etc.

Knowledge-based pattern discovery: Engine error pattern.

Measurement

- Conversion
  - Measured channels extraction
  - Measured channels reduction
  - Pattern discovery

Recording

- Preprocessing
- Feature extraction
- Feature reduction
- Classification

Abstract

Concrete

Manual evaluation process

AND other Conditions...

OR ...
Research Question: How can a tool support the analysis of engine error data?

1. **Pattern discovery**
   - Tool supports the manual evaluation
   - Export data
   - Visualize different measured channels

2. **Pattern definition**
   - **Static Condition:**
     - Engine temperature is larger than 500 °C
     - \textit{Static condition} = \textit{Measured channel} + \textit{predicate} + \textit{threshold value}
     - 4 predicate: “is larger than”, “is smaller than”, “is exact equal”, “is around”
   - **Dynamic Condition:** Time series pattern (signal pattern) definition:
     - Use already evaluated data from the pattern discovery
3. Pattern recognition

- Static conditions: Compare values to determine whether the condition is satisfied
- Dynamic conditions: time series matching:
  - Consider particular features of engine error data
  - Euclidean Distance: time series $x$, $y$, length $M$, threshold value $\varepsilon$
    \[
    d_{L_2}(x, y) = \left( \sum_{i=1}^{M} (x_i - y_i)^2 \right)^{1/2} < \varepsilon
    \]
  - Discrete Fourier Transformation:
    \[
    d_{FC}(x, y) = \left( \sum_{i=1}^{M} |\tilde{x}_i - \tilde{y}_i|^2 \right)^{1/2} < \varepsilon
    \]
- Tolerance range:
  - Permissible deviation $\sigma$
**Characteristic of measured data & user requirements**

<table>
<thead>
<tr>
<th>Time window</th>
<th>Absolute level</th>
<th>Different sampling rates &amp; missing values</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="#" alt="Graph" /></td>
<td><img src="#" alt="Graph" /></td>
<td><img src="#" alt="Graph" /></td>
</tr>
</tbody>
</table>

- **Time window contradicts horizontal shift**
- **To reduce complexity** ➞ **no vertical shift**
- **No vertical shift**
- **No amplitude scaling**
- **No transformation** (rotation)
- **Interpolation**
- **Smallest sampling rates**

---

**Compare value vectors of two signals with equal length, identical time vectors.**
1. **Static Condition**

- Area of interest $\Delta$: represents the interval, in which users are still interested in the value, although the static condition is not satisfied.
- In the area of interest, the matching degree $\mu$ is calculated linear.
- Example: “Engine temperature ($T_{\text{engine}}$) is larger than 100 °C“
  - Area of interest $\Delta = 20$ °C $\Rightarrow$ Intervall [80 °C, 100 °C]
  - $T_{\text{engine}} = 120$ °C $\Rightarrow \mu = 1.0$
  - $T_{\text{engine}} = 85$ °C $\Rightarrow \mu = 1 - (100 - 85)/20 = 0.25$
  - $T_{\text{engine}} = 95$ °C $\Rightarrow \mu = 1 - (100 - 95)/20 = 0.75$
  - $T_{\text{engine}} = 70$ °C $\Rightarrow \mu = 0.0$
2. **Dynamic Condition**

- Permissible deviation $\sigma$: user-defined $\Rightarrow$ threshold value $\varepsilon$
- Euclidean distance:
  - Calculate the Euclidean distance:
    \[
    d_{ED}(x, y) = \left( \sum_{i=1}^{M} (x_i - y_i)^2 \right)^{1/2}
    \]
  - Compare distance with threshold value:
    - If $d_{ED} \leq \varepsilon$: $\mu = 1.0$
    - If $d_{ED} \geq 2\varepsilon$: $\mu = 0.0$
    - If $\varepsilon < d_{ED} < 2\varepsilon$: $\mu = 1 - \frac{d_{ED} - \varepsilon}{\varepsilon}$
Tool-support: system architecture

- Modular design
- No installation of Software
- Changes, new features easy to implement

**Diagram:**

- **Client-Tier**
  - Web-Container
    - Managed Bean
    - XHTML
    - User

- **Middle-Tier**
  - EJB-Container
    - Abstract Facade
    - DataAnalysis
    - readMDF

- **Data-Tier**
  - Database
  - Engine error files
Indirect MDF files reading

Java

(1) call

(5) read

Python mdfReader

(4) return

(2) read

XML Files

MDF Files

Messkanäledaten

```
<MDFChannelListXml>
  <MDFChannelXml>
    <name>Drehzahl</name>
    <description>Motordrehzahl</description>
    <unit>1/min</unit>
    <data>2000,1000,500,0,0,100</data>
    <time>-3.0,-2.0,-1.0,0.0,1.0</time>
  </MDFChannelXml>

  <MDFChannelXml>
    ...
  </MDFChannelXml>

  ...

</MDFChannelListXml>
```
Results

- Export important signal values in Excel’s table: VBA-scripts for further edition

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>1200</td>
<td>100</td>
<td>0</td>
<td>600</td>
<td>200</td>
<td>…</td>
</tr>
<tr>
<td>File 2</td>
<td>900</td>
<td>100</td>
<td>0</td>
<td>500</td>
<td>100</td>
<td>…</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

- Signal visualisation in PDF

- Automated classification of error files into matching error patterns (100% matched)

- Show pattern matching degrees: help user by further decision
Comparison of similarity degrees

- Three similarity search methods:
  - Euclidean distance
  - Tolerance range
  - Discrete Fourier-transformation
- 100 real signals of rotation speed
- Same permissible deviation, same time window
Comparison of similarity degrees

<table>
<thead>
<tr>
<th>Relative deviation</th>
<th>Equal 0 %</th>
<th>From 0 to 10 %</th>
<th>From 10 to 20 %</th>
<th>From 20 to 30 %</th>
<th>More than 30 %</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of signals</strong></td>
<td>10</td>
<td>7</td>
<td>35</td>
<td>8</td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Tolerance range</th>
<th>Euclidean distance</th>
<th>Discrete Fourier-transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of totally matching signals</strong></td>
<td>10</td>
<td>63</td>
<td>48</td>
</tr>
<tr>
<td><strong>Average matching degree</strong></td>
<td>0,85</td>
<td>0,69</td>
<td>0,68</td>
</tr>
</tbody>
</table>

- 10 % exact same matching degrees
- 60 % relative deviations less than 30%
- Tolerance range compares each point ➔ least tolerant method
- Euclidean distance summerize the difference of each points and compare the average distance ➔ more tolerant than other methods
- Matching degree depends strongly on the deviation
Conclusion & further work

- Software provides functions that fulfills user’s requirement.
- Adding further parameters or new features is easy to implement.
- Read MDF files directly in Java.
- Use of other similarity search methods.
- Pattern matching degree must be adapted to each particular method.
- Concept is without major changes applicable for many other kinds of measured data in the automotive area: transmission error, supervision of sensor, long-time quality data etc.
- Other domains e.g. medical diagnosis.
Thank you for your attention!