Comparison of Structured vs. Unstructured Data for Industrial Quality Analysis

Christian Hänig, Martin Schierle, Daniel Trabold *

Abstract—**Industrial methods for quality analysis massively rely on structured data describing product features and product usage. The analysis of such data is normally done using complex reporting or sophisticated data mining methods. Besides this structured data, companies very often also posses large amounts of unstructured text like call center reports, internet fora or repair order documents. Despite the rising interest in text mining applications for industrial usage, the uncertainty about the real benefits is still high.**

In this work, we will present a comparison of the usage of structured versus unstructured data on two quality analysis use cases: Early warning and the detection of repeat repairs.

Keywords: Text Mining, Data Mining, Quality Analysis

1 Introduction

Although Natural Language Processing (*NLP*) methods gained a lot of scientific interest over the past few decades, industrial use cases are still rare. Companies used to have mainly structured data (if there were data warehouses at all), and NLP methods were often complicated, unstandardized or just too slow.
Nowadays even small companies keep track of lots of textual data, for example call center reports, e-mail correspondences or repair order texts. In fact, with regard to the rise of the internet and especially the Web 2.0, textual content seems to explode. Luckily NLP methods also evolved – besides textual resources like WordNet, there are also more than sufficient software resources available for usage. Often pre-trained and standardized, it is easy to engineer systems using these modules and an according framework like UIMA (see [4]) or GATE (see [3]). Furthermore the current methods are (with respect to modern computer technology) fast enough to use, and there are also unsupervised methods available, to keep the amount of manual annotation of training sets down.
But there is still one question left – what exactly is the additional value of textual analysis compared to structured information? Considering the internet and the Web 2.0, the overvalue is conclusive, as there is no similarly extensive of information available in structured form. But with respect to internal data sources of a company and a specific industrial task, there is not much research in the comparison between structured and unstructured data. Admittedly this comparison is hard to perform, requiring large datasets of comparable data in structured and unstructured form.

This paper will try to answer this question for the specific task of quality analysis in the automotive domain. This task is not only of major importance for every manufacturing company, but it also requires methods of high accuracy. We will apply two different subtasks on a very large dataset of repair data, which contains as well structured information like damage and part codes, but also unstructured data in the form of a repair order text. This text is written by the mechanic, and contains the customer's complaint as well as the mechanics' repair actions.

2 Text Pre-processing

First step of the NLP workflow is dealing with text preprocessing to increase the textual quality. Therefore the language is detected based on letter n-grams, using the NGramJ library which is based on [2].
After the language recognition and the tokenization using regular expressions several cleaning steps are performed. These include context-sensitive replacement of abbreviations as well as context-sensitive spell-checking using neighbourhood word co-occurrences. The cleaning steps are described in more detail in [7].
The application scenarios described in the following sections are all based on concepts (or: unnamed entities) which are detected in the text using a domain specific taxonomy containing a restricted vocabulary of technical terms. The taxonomy is organized as a multilingual and poly-hierarchical knowledge base, and was derived from available company sources and extended manually. The hierarchy is organized in terms of semantics instead of technical arrangements. To achieve multilingualism we represent every word as a language independent concept, and every concept may have more than one parent (e.g. a *radio fuse* may be situated under *radio system* as well as under *fuses*). Every concept can be defined in several languages and with several synonyms per language. For example *fuses* is just the English word for a concept that is called *Sicherungen* in German and *fusibles* in French.

*Daimler AG, Research and Technology, 89081 Ulm, Germany, {christian.haenig, martin.schierle, daniel.trabold}@daimler.com

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The handling of ambiguities is ensured by the optional specification of part-of-speech tags per word, and a context per synonym for each language. Therefore *pump* can be identified as a component if the word is used as a noun, or as an action when used as a verb. The pos-tags where obtained by a Hidden-Markov-Model based pos tagger, which was trained on approx. 26000 manually annotated words from our domain, and yields an accuracy of 92.2% [8].

In order to extract relations from plain text, we add information about structure using syntactic trees. This is a crucial step for relation extraction and allows the discovery of relations of any arity (e. g. component symptom or component symptom condition). Syntactical parsers trained on general-purpose treebanks do not yield accurate results due to the absence of annotated corpora containing domain specific structures and terminology. Thus, we use an unsupervised parser as in [5] which solely needs raw text with annotated syntactic word classes. The unsuParse algorithm learns the structure of a language based on neighbourhood co-occurrences and preferred positions of tokens. To circumvent data sparseness it uses word classes instead of word forms. Those syntactic classes can be annotated in an unsupervised way (unsuPOS, see [1]) to stay language and domain independent. Additionally, we assign semantic tags like *component*, *symptom* and *location*, which are derived from the categories of our knowledge base described above.

We use a set of predefined heuristic rules to extract relations from those resulting parse trees. Every rule contains information about the two participating categories of concepts and a maximum distance. This distance is calculated as the sum of the word distance and the node distance within the parse tree. The detailed algorithm is described in [6]. It yields very accurate results for relation extraction on repair orders (precision of 86% - 89% and recall between 67% and 84% depending on the type of relation).

3 Early Warning

For an international manufacturer of premium brand vehicles it is crucial to be aware of any kind of quality problem as early as possible. Even some days can make a difference with respect to customer satisfaction and media attention. Beside the impact on brand perception and marketing, there are also legal issues like liability to consider.
Therefore every manufacturer runs *Early Warning* processes as part of the everyday business intelligence. Input to these algorithms is usually structured data like part codes or damage codes. In addition to these data sources, some companies also posses large amounts of unstructured texts like call center reports or repair order texts, but their potential for early warning is still unclear.
This section describes how early warning algorithms can be applied on textual input, and evaluates the results on historical data from our company.
Input to the algorithm is a set of approximately 2.5 million repair cases $R = (r_1, ...r_n)$, all taken for one specific model family, and restricted to cases in the US. We applied this restriction as repair order texts are only written down continuously in the US. For every repair one unstructured text is recorded, normally as noisy text of at most several ten words of domain language. After the application of the information extraction steps outlined in section 2, we will end up with the following data per repair:

1. The code for the defective component, as given by the company's damage codes
2. The code for the observed symptom, as given by the company's damage codes
3. A set of concept-ids of components as extracted from the text
4. A set of relations between components and symptoms as extracted from the text

Being now only confronted with structured data, the Early Warning algorithm can be applied to the company's damage codes and the text respectively. The comparison is done on a component level and a relation (component with symptom) level, because those two approaches differ in complexity of the applied algorithms.
The algorithm which we will use is similar to the real process in our company[1], but simplified in some specific points which are of no importance to the comparison of structured data and unstructured data. We will furthermore only create warnings for cars with less than six months of usage. For the further work, we will define the following:

1. A specific damage code from all codes D is defined by d_l. For simplicity components and relations extracted from text are transformed into codes in order to apply the same algorithm to all data.
2. A specific test month is denoted by m_i, a specific production month by p_j.
3. C defines the set of cars, while $C(p_j)$ defines the set of cars from the given production month. We will use C_{6+} for the set of cars used more than 6 months, and C_{6-} for the other cars respectively. Be aware, that the time of usage is not implied by the production month, as the car might have been sold later.

[1]Thanks to Dr. Matthias Grabert for his friendly help with the company's Early Warning mechanisms.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

4. Repairs are denoted by R, R_{6-} is used for cars with less than 6 months of usage. $R_{6-}(m_i, p_i)$ is the set of repairs on cars from the given production month in the given test month, which had less than 6 months of usage.

As a first step, all codes D_S are identified, which show a seasonal behaviour. These are excluded from the early warning detection as an analysis of these codes is far more complicated.
The damage rate X of a given month m_i and damage d_j is defined by the ratio of cars having a repair of d_i in m_i to the amount of all cars C in usage in this month.

$$X_m(m_i, d_j) = \frac{|R_{6-}(m_i, d_j)|}{|C_{6-}(m_i)|} \tag{1}$$

The dataset with all repairs is divided in two subsets, one for training, and one for evaluation. On the training set (which covers two complete years of data) the damage rates $X(m_i, d_i)$ are calculated for every damage code and every calendar month (without respect to the production month). These values are input to a multivariate linear regression, assuming that seasonal damages (like problems with heater or air-conditioning) follow a trigonometric function over test months:

$$\begin{aligned} f(m_i) &= a + bx_1 + cx_2 \\ x_1 &= \sin(X(m_i)) \\ x_2 &= \cos(X(m_i)) \end{aligned} \tag{2}$$

The *coefficient of determination* R^2 is used to determine the quality of the regression. Every code with $R^2 > \sigma$ is considered to be seasonally influenced. The first part of the evaluation will be the comparison of seasonal codes from structured and unstructured data.
For the early warning process itself, a different damage rate is calculated. With respect to the seasonalities we calculated X based on the repairs in a given calendar month. Aiming at the identification of production anomalies, the subject of analysis is a given production month. The test-month only defines the month of the analysis (as part of the daily quality analysis process), but the repairs will be counted **up to this month**. Additionally only cars are considered which completed their first six months of usage in or before the test month m_l. The damage rate $X'(p_i, d_j, m_l)$ of a given production month p_i and damage d_j in a given test month m_l is defined as follows:

$$X'(p_i, d_j, m_l) = \frac{\sum\limits_{k \in \{i \ldots l\}} |R_{6-}(p_i, d_j, m_k)|}{|C_{6+}(p_i, d_j, m_l)|} \tag{3}$$

This value is calculated once for every code for the training data (but for no specific production date), and denotes the error probability $p = X'$. By assuming an underlying binomial distribution of erroneous cars, we can infer the mean μ and the standard deviation σ for a given population C_{6+}. For the warning process, the damage rate X' is calculated for the cross product of damage codes, test months and production months, and compared with the upper control limit (UCL):

$$UCL = \mu + 3\sigma \tag{4}$$

If the damage rate for the given damage code and production month is higher than the UCL, the system will generate a warning. The comparison between the early warning processes is done by comparing the generated warnings.
For the evaluation one has to be aware of one important fact: despite all the company's knowledge about historical quality issues, it is nearly impossible to state which source of data is right, if there are additional or missing warnings. We cannot rely on domain experts or the companies documentation for this evaluation, as they are biased by the early warning systems used at this time. Furthermore one structured damage code normally maps to several text codes (because the text might name several components where the code only names one), leading to much more warnings generated from text than from structured data. Therefore a direct mapping of warnings is not possible either. To deal with these issues, we evaluated the following way:

1. Warnings (and seasonalities) generated from text were clustered using co-occurrence significances of the related concepts (calculated using the t-score measure) and the Markov Clustering algorithm [9]. This is done to keep the manual effort for evaluation down, but all clusters were manually reviewed and corrected to exclude the clustering process as possible erroneous influence. This leads to the text clusters C_T.

2. The warnings (and seasonalities) from structured data were manually clustered. But as they are defined and used distinctly, only few codes were clustered together - leading to the structured data clusters C_S.

3. By including as much information as possible, the clusters from both early warning (and seasonality) calculations are manually compared and mapped if possible. Two warnings are considered equal if they are temporarily (within 6 month) and semanticaly (describing identical problems, depending on each other or having the same root cause) close.

After this process, we can calculate the agreement A between the two approaches as follows:

$$A(C_T, C_S) = \frac{|C_T \cap C_S|}{|C_T| \cup |C_S|} \tag{5}$$

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Table 1: Agreement of calculated seasonalities and warnings

	Component based	Relation based
Seasonalities	69%	45%
Warnings	41%	37%

We argue that a high agreement of structured and unstructured information can be considered as a proof that textual data can be seen as an at least comparable source of data with respect to early warning. The results of the agreement calculation can be seen in table 1. Examples for analogous seasonalities can be seen in figures 1 and 2, examples for warnings in figures 3 and 4.

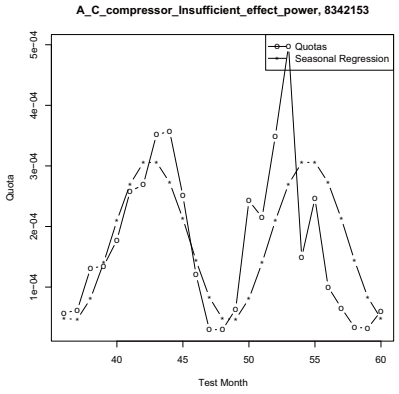

Figure 1: Seasonal behaviour of the structured code 'AC compressor insufficient effect power'

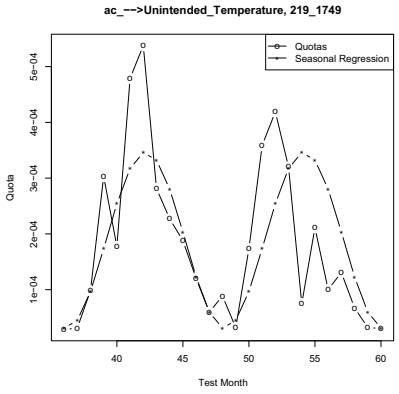

Figure 2: Seasonal behaviour of the relation 'AC unintended temperature' extracted from text

The results show, that the agreement between the generated warnings and seasonalities is too high to be neglected. Nearly half of the calculated items could be mapped. It is to mention, that many of the other items were rather similar in nature, or even identical but related

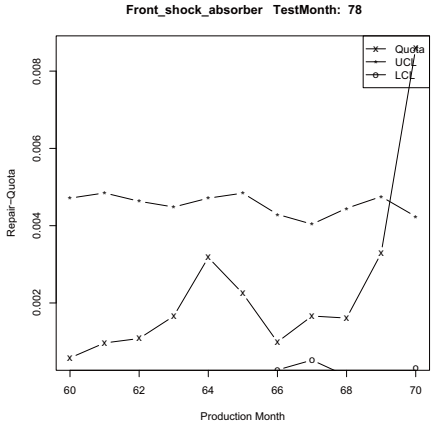

Figure 3: Warning generated from structured data for the shock absorbers

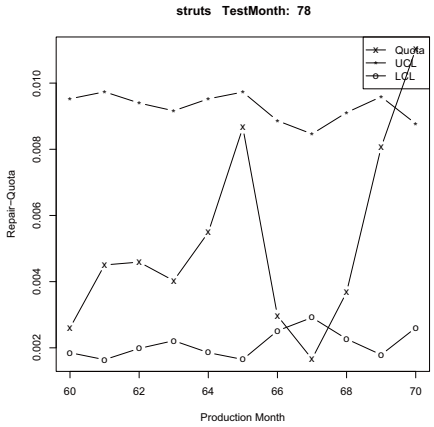

Figure 4: Warning generated from text for struts

to quite different test months. Regarding the items which could be mapped, we became aware of three interesting observations:

1. Damage codes were designed to uniquely identify a specific part of the car. They are highly technical and very specific in nature. The concepts and relations extracted from text are more customer centric and general. While technical codes are better suited to find erroneous part of the car, the text is better suited to identify the general misbehaviour of the car, as noted by the customer. For example we identified several different structured codes of the air conditioning as seasonally influenced, but mainly only two different problems derived from the text: An air conditioning which doesn't blow cold, and an air conditioning which smells bad. Of course there might be lots of different technical parts of the air conditioning possible to fail, but the customer will allways only notice two distinct problems. Therefore we conclude that an early warning based on text can

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

be important to become aware of customer problems, instead of car problems.

2. The information from the text is perfectly suited to reinforce and verify the structured information. Warnings which were found by both approaches can be seen as confirmed. On the other hand, the text might help to find erroneous repairs or encoding problems. Our evaluation identified several examples where the structured code didn't fit to the text.

3. For all the warnings in agreement, we took a closer look on **when** the warnings were generated. It turned out, that the structured data warned earlier in five cases for components (two cases for relations), while the text was faster in seven cases (one for relations). In average of these cases, the structured data was two months faster (1.5 for relations), the text was 2.1 months faster (2 for relations). Therefore we conclude that text warnings are important for quality analysis, as they might occur earlier.

In summary we conclude, that textual early warning can be done with sensible results. Although the results might not be completely identical to the ones from structured data, they give interesting insights and reveal problems from the customers view. Furthermore the traditional results can be confirmed and enriched with additional information.

4 Repeat Repair Detection

A repeat repair is a second (or further) repair attempt to a customer's complaint. Thus, a high *fixed first visit rate* indicates excellent work and satisfied customers in the dealership. Of course, having a problem not solved during the first dealership visit leads to negative responses. But knowledge about successful repair attempts – that ones not being followed by a repeat repair – can be used to gather comprehensive information about the dealership process. Statistics about the success rate of repairs identify:

- the repair approach to a given problem achieving the highest success rate,
- the spare parts which are used in succeeding repair attempts,
- the mechanic yielding best results.

Hence, accurate detection of repeat repairs based on textual descriptions of the problems is able to benefit the complete scheduling process including spare part ordering and assignment of the accountable mechanic. Service operators can provide customers with the mechanic holding corresponding skills for the customer's problem, can order the spare parts which will be used most likely and when the customer comes in, all needed resources will be available.

Summing up, there are basically two challenges. Firstly, statistics regarding the success of repair approaches have to be accurate to be able to provide reliable predictions. Secondly, the customer's complaint – available as textual data – has to be analyzed and classified to access statistics of the addressed problem. While methods using structured and/or unstructured data can be used to calculate repair attempt statistics, the second challenge can only be approached by text mining algorithms as structured data will only be available *after* repair actions are completed.

In this section, we want to introduce a new methodology using textual data provided by the customer to detect repeat repairs. Additionally, we will compare it to the performance of currently applied approaches using structured data.

4.1 Methods using structured information

For every repair action taken, a so called *repair order* will be created. It contains free text fields like the customer's complaint, the cause of the problem and the corrective actions. While those three fields where typed in by the technician, structured information is added by a service advisor after completion of the repair. Two types of codes are of special interest here: labour operation codes encode the applied repair actions, and part codes are used to encode employed spare parts. To detect repeat repairs using *structured data*, codes for labour operations and parts are manually clustered to recognize different labour operations applied to similar symptoms. Repair orders which are classified into the same category are considered to be repeat repairs.

4.2 Analyzing textual data

Textual information is unstructured data and thus, we need to create a new methodology to find similar problem descriptions. Basically, there are three clues for detecting a repeat repair:

1. Direct reference: Some repair orders contain a direct reference to former repair orders. This reference can be given as a date or – as in most cases – as a repair order number. For example:

> **customer states vehicle pulsates when stopping.**
> ***see previous ro# 121753***

Those references are easy to extract, of high accuracy but also very rare and exist only for about

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

2.5% of all repeat repairs.

2. Expression of repetition: Some customers mention the repetition in the complaint. The most common (and still friendly) utterance that can be observed is:

> **a/c is blowing warm *again***

These expressions are — similar to direct references — very rare and barely existent.

3. Similar textual problem description: Similar problem descriptions refer to similar problems and thus, similar problem descriptions of a customer – within a certain range of time and mileage – lead to the assumption, that the first dealership visit did not fix the problem.

Text based repeat repair detection classifies a repair order as repeat repair, if at least one of the above mentioned clues is present. While references to former attempts and utterances of repetition are basically pattern matching problems, the third one is more complex.

We use feature vectors v_{text} for repair order representation as used in information retrieval. Words are weighted using their relative frequencies while stop words are neglected. To calculate similarities sim_{text} between those vector representations, we apply the cosine measure.

As language provides various ways to express the same subject or problem, this basic approach cannot detect all repeat repairs. Thus, we use our internal knowledge base which is able to deal with synonymous expressions. We build vectors $v_{taxonomy}$ containing the extracted concepts of our taxonomy to additionally match similar expressions like *does not work* and *inoperative*. The weights depend on the relative frequency of the corresponding concepts and are multiplied by 0.5 for each level above the matched expression in the hierarchy of our taxonomy.

The vectors v_{text} and $v_{taxonomy}$for the repair order *center cap is missing* are given in equ. 6 and 7.

$$v_{text}\,(\text{center cap is missing}) = \begin{pmatrix} center & 0.33 \\ cap & 0.33 \\ missing & 0.33 \end{pmatrix} \quad (6)$$

$$v_{tax}\,(\text{center cap is missing}) = \begin{pmatrix} center\ cap & 0.31 \\ tires & 0.15 \\ chassis\ and\ suspension & 0.08 \\ missing & 0.31 \\ loss\ of\ something & 0.15 \end{pmatrix} \quad (7)$$

Some repair orders contain more than one problem. It is possible, that a textual description containing similar components and/or symptoms leads to high similarity scores although the concepts are not related the same way as in the other repair order. To avoid those false classifications, both repair orders must have at least one relation in common.

4.3 Evaluation

To evaluate and compare the performance of those three different approaches, we have chosen 608 repair order pairs (the repair orders of such a pair belong to the same vehicle) randomly and manually annotated the second as repeat repair or normal repair order. This evaluation sample contains 146 repeat repairs. Calculated scores for precision, recall and f-score are given in table 2. The results show the superior coverage and higher accuracy of textual data for this task. Even the basic approach using the text without concept recognition outperforms structured data easily. The following two examples illustrate the advantages of text based classification.

Table 2: Precision and Recall values

Method	P	R	F-Score
Labour operations and parts	0.74	0.46	0.57
v_{text}	0.79	0.81	0.80
$v_{taxonomy}$	**0.82**	**0.94**	**0.88**

4.3.1 Example 1: Labour operations and parts match exactly

Labour operations of both repair orders match exactly and so do the used spare parts. But – obviously to humans reading the text – it is not a repeat repair. The textual data reveals that a center cap fell off in both cases, but different ones were affected. Codes for labour operations are not that fine grained as it would be necessary for some tasks. In this case, structural data leads to a false alert.

> **customer states *driver front* wheel center cap is missing**
> Labour operations: 22600501
> Parts: CH52013653-AA

> **customer states the *passenger side rear* hub cap fell off**
> Labour operartions: 22600501
> Parts: CH52013653-AA

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

4.3.2 Example 2: No structured data matches

The second example shows a really obvious repeat repair. The customer states the same complaint *again*. Though the technician applies different corrective actions to the same problem, codified information is not able to find the linkage between these two repair orders.

customer states *a/c blows out hot air* at times check and advise
Labour operations: 24010102 (Leak check)
Parts: CH5015778-AA CH82300329

***a/c is blowing warm* again**
Labour operations: 08194995 (Reprogram control module)
Parts:

4.4 Conclusions

Comparing methods based on structured respectively unstructured data for repeat repair detection comes up with a clear result: unstructured data contains more precise information and outperforms the current approach using labour operation and part codes. Additionally, methods analyzing textual data can be applied before structured data is even available and can be used to improve dealership processes.

5 Conclusions and further work

Summarizing the results of this work, we conclude that unstructured data is (if available) a useful extension to structured data and needs to be analyzed as well. With respect to early warning, textual data yields comparable results to damage codes and provided interesting insights in the customer's point of view. Regarding the task of repeat repair detection, unstructured information is even superior to structured data due to fine-grained information like *conditions* and *locations* which is available in textual data.

In our future work, we will examine the influence of textual data to further tasks like root cause analysis and identification of erroneous encodings. Another crucial improvement will be the adaption to other languages and domains besides optimizations of the relation extraction algorithms.

References

[1] C. Biemann. Unsupervised part-of-speech tagging employing efficient graph clustering. In *Proceedings of the COLING/ACL-06 Student Research Workshop*, Sydney, Australia, 2006.

[2] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, Las Vegas, US, 1994.

[3] H. Cunningham. *Software Architecture for Language Engineering*. PhD thesis, University of Sheffield, 2000. `http://gate.ac.uk/sale/thesis/`.

[4] D. Ferrucci and A. Lally. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering*, 2004.

[5] C. Hänig, S. Bordag, and U. Quasthoff. Unsuparse: Unsupervised parsing with unsupervised part of speech tagging. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, 2008.

[6] C. Hänig and M. Schierle. Relation extraction based on unsupervised syntactic parsing. In *Proceedings of the conference on Text Mining Services*, Leipzig, Germany, 2009.

[7] M. Schierle and S. Schulz. Bootstrapping algorithms for an application in the automotive domain. In *Proceedings of the Sixth International Conference on Machine Learning and Applications*, pages 198–203, Los Alamitos, CA, USA, 2007. IEEE Computer Society.

[8] D. Trabold and M. Schierle. Multilingual knowledge based concept recognition in textual data. 2008.

[9] S. M. van Dongen. *Graph clustering by flow simulation*. PhD thesis, University of Utrecht (The Netherlands), 2000.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)