IRVINE: A Design Study on Analyzing Correlation Patterns of Electrical Engines

Joscha Eirich, Jakob Bonart, Dominik Jäckle, Michael Sedlmair,
Ute Schmid, Kai Fischbach, Tobias Schreck, and Jürgen Bernard

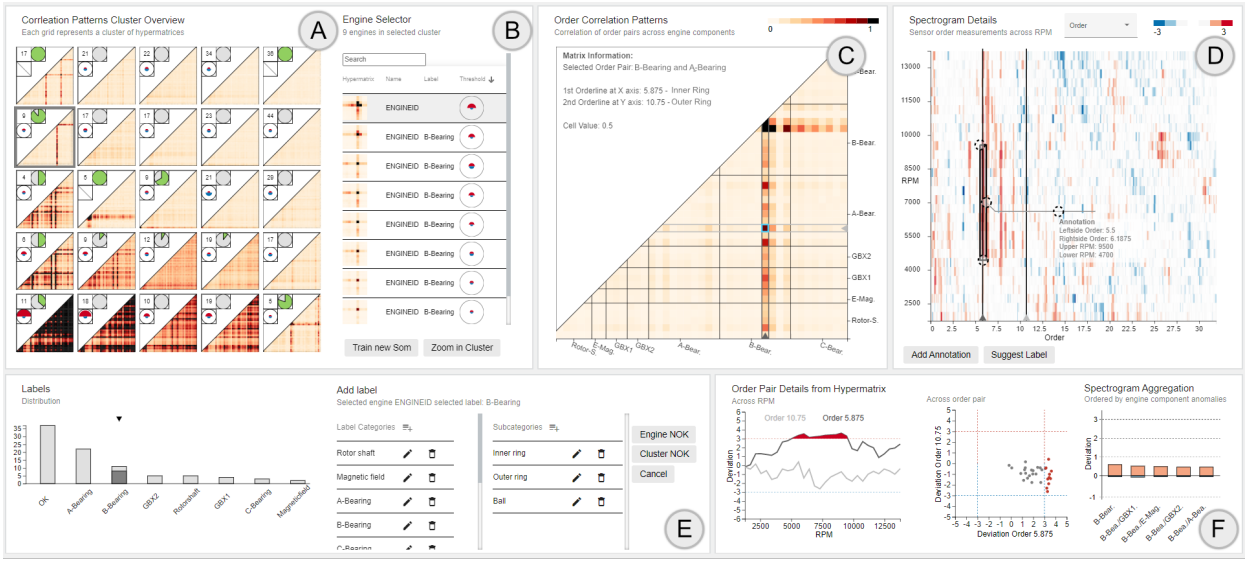


Fig. 1. The IRVINE system. Users have an overview over clusters in (A). They can select clusters in (A) and engines in (B). After selecting an engine in (B), the acoustic signature of the engine is displayed in (C) and respective raw acoustic measurements in (D). Detailed information about selections from (C) is shown as line chart and scatter-plot and bar chart in (F). After the analysis of an engine, the user can assign a label in (E) and provide an annotation for the label in (D).

Abstract—In this design study, we present IRVINE, a Visual Analytics (VA) system, which facilitates the analysis of acoustic data to detect and understand previously unknown errors in the manufacturing of electrical engines. In serial manufacturing processes, signatures from acoustic data provide valuable information on how the relationship between multiple produced engines serves to detect and understand previously unknown errors. To analyze such signatures, IRVINE leverages interactive clustering and data labeling techniques, allowing users to analyze clusters of engines with similar signatures, drill down to groups of engines, and select an engine of interest. Furthermore, IRVINE allows to assign labels to engines and clusters and annotate the cause of an error in the acoustic raw measurement of an engine. Since labels and annotations represent valuable knowledge, they are conserved in a knowledge database to be available for other stakeholders. We contribute a design study, where we developed IRVINE in four main iterations with engineers from a company in the automotive sector. To validate IRVINE, we conducted a field study with six domain experts. Our results suggest a high usability and usefulness of IRVINE as part of the improvement of a real-world manufacturing process. Specifically, with IRVINE domain experts were able to label and annotate produced electrical engines more than 30% faster.

Keywords: Design study, interactive labeling, interactive clustering.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces (GUI); User-centered design

- *J. Eirich, U. Schmid, and K. Fischbach are with University of Bamberg*
- *J. Bonart is with Fraunhofer IWU*
- *M. Sedlmair is with University of Stuttgart*
- *T. Schreck is with Graz University of Technology*
- *J. Bernard is with University of Zurich*
- *J.Eirich, J.Bonart and D.Jäckle are with BMW Group*

Manuscript received xx xxx. 202x; accepted xx xxx. 202x. Date of Publication xx xxx. 202x; date of current version xx xxx. 202x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.202x.xxxxxxx

1 Introduction

The automotive industry is currently in the midst of its greatest change in the last 100 years, namely in the direction of electromobility [28]. With the setup of electrical engines (from here on: engines), of course, new, previously unknown errors are introduced. For car manufacturers, it is extremely important to identify and understand these errors to meet high-quality standards. However, detecting and understanding such errors is currently hampered by two major challenges: **(1)** Serial manufacturing processes produce a large quantity of parts, hence, resulting in a vast amount of recorded sensor data during testing. On-site interviews with test engineers showed that they are required to analyze more than forty interdependent signals from hundreds of produced engines manually and per hour. Furthermore, the measured signals can be related to multiple sub-components of an engine. This situation poses a major challenge and as a result test engineers are capable to

only analyze few parts in detail in the given time. **(2)** Because only a few engines can be analyzed and their errors classified in detail, the gained knowledge is particularly precious for the overall improvement of the testing procedures. Yet, it is currently unknown how the gained knowledge can be stored or transferred. To address the named challenges, we present *IRVINE (InteRactiVe cluseterINg labEling)*, a Visual Analytics (VA) system that results from a design study [49] project carried out together with automotive engineers at BMW.

We base our study on acoustic data that is collected through the propagation of sound inside the engines. The analysis of acoustic data allows specific errors to be recognized. For example, a dirty bearing sounds differently from a clean bearing. The interaction between different acoustic frequencies forms so-called signatures. Signatures typically comprise one primary and multiple secondary error symptoms. On top of raw acoustic data, we also include the derived correlations in acoustic signatures, which enable interdependent error analyses. Given the collected acoustic data, *IRVINE* provides capabilities for clustering similar signature correlations. To gain a fine-grained overview of similar signatures, clustering can be performed interactively by engineers on a subset of clusters or engines with previously provided labels. Engineers can then select specific engines of a cluster for an in-depth analysis. To record their findings we introduce knowledge capturing capabilities through labeling and annotation. In this regard, we refer to labeling as the error class (e.g. B-Bearing error), which serves as a basis for the classification of further errors. In contrast, we refer to annotations as a means to record the reason for an error within the raw sensor data (e.g. Threshold violation in B-Bearing measurement).

To better support the goals and tasks of engineers, we first familiarized with their domain and gained an understanding of their problems. The *IRVINE* system was iteratively designed, developed, and evaluated in close collaboration with engineers at BMW. The overall goal of the system is to support engineers in the detection and analysis of error-prone produced engines. *IRVINE* was hereby tailored to fit their domain-specific requirements and designed to be seamlessly integrated into their daily work. As a result, the systems can handle vast amounts of data enabling engineers to easily analyze multiple engines, their signatures, and acoustic raw data. Labels and annotations are stored in a knowledge base, which is used to both enhance *IRVINE* and to be available for different stakeholders, who can benefit from labels and annotations.

In summary, our contributions are: (1) The problem characterization and abstraction of the studied use case; (2) the reporting of the interactive design of IRVINE; and (3) the evaluation of IRVINE together with six automotive engineers and reflections of our design process.

2 Related Work

We start by giving a brief summary on related work about *interactive clustering* (Section 2.1), *interactive labeling* (Section 2.2), and previous *design studies in the automotive sector* (Section 2.3).

2.1 Visual Interactive Clustering

Interactive clustering approaches have been proposed for various data types, including trajectory data [3], text documents [40], (social) networks [37], and time series data [8]. Furthermore, other visualization approaches focus specifically on the visual analysis of sensor data [1] including a variety of analysis goals, such as segmentation, clustering, or classification [38]. In scenarios where very large data sets are analyzed, some approaches make use of matrix-based visualizations, for instance to group similar matrices that show changes in brain connectivity networks [4]. With respect to our clustering scenario, we focus on the grouping of large amounts of acoustic signatures of engines on their similarities, with the help of matrix visualizations. Some approaches explicitly leverage domain knowledge, e.g. Yang et al. [59], focusing on interactive steering methods to visually constrained clustering using both user and publicly available knowledge. The selection of specific subsets of interest where another subsequent computation of the clustering is applied plays also an important role [42].

Interactive clustering is often combined with dimensionality reduction techniques [43] to benefit from the complementary strengths of a) reducing the number of instances from many to a few (via clustering) and b) reducing multiple data dimensions to a low-dimensional visual representation (dimensionality reduction). We use the Self-Organizing Maps (SOM) [32] clustering algorithm, which naturally combines both steps. Vesanto was one of the pioneers who showed the benefits of SOMs for visual cluster analysis [55], and many methodological contributions to interactive clustering followed from VA research. Schreck et al. [46] presented a VA system that allowed the interactive visual initialization, quality assessment, and refinement of SOMs. *IRVINE* also supports the interactive refinement of SOMs [41], with both with different parameters and data subsets of interest. The SOM has also been used for the visual analysis of sensor data, in MotionExplorer [12] to facilitate exploratory search and in FuryExplorer [57] for the comparison of cluster patterns with attached metadata. Finally, the *SOMFlow* system serves as a platform for the interactive exploration of multiple SOMs, the analysis and refinement of intermediate results, and the back-and-forth navigation along the analytical workflow [42].

In our design study, we use interactive clustering to support the engineers in the analysis of acoustic data. Particularly inspiring for the specific design of *IRVINE* are the iterative training and refinement, the training of subsets of interest, and the analysis of individual clusters and cluster elements in detail for downstream labeling tasks.

2.2 Visual Interactive Labeling

As well as interactive clustering, labeling is also a frequently supported task in VA [20]. In this regard, Bernard et al. [13] propose the concept of *Visual-Interactive-Labeling* (VIAL), to label yet unknown data in an interactive exploratory setup. VIAL hereby bridges the gap between active learning [50] and advanced visualization concepts, where the type of labels depends on the given task and approach. Categorical labels count to the more common types, which can either be of binary or multi-valued nature [9]. While binary labels would allow simple user feedback, such as "ok vs. not ok", *IRVINE* supports multi-valued labels, enabling a more specific tagging of different classes for a data instance. Other examples for systems that enable categorical labeling are provided for textual documents [25], bio images [7], handwritten digits [9], or video streams [27]. Considering our use case, we focus on the assignment of categorical multi-valued labels.

A second important type of labels is continuous labels, which are often applied when a more fine-grained degree of interestingness is required. Here, systems exist for candidate rating and evaluation [56] or to choose between irrelevant and relevant views [6]. Yet other labeling approaches allow the comparison of pairs of objects [10] or groups of objects [16] in combination with algorithmic models using this implicit feedback to adjust the attribute weightings or the feature space. Finally, another type of user feedback is to annotate features or data attributes directly. For instance, approaches exist that support the dynamic evaluation of feature subsets and resulting models [60] or to annotate images while relating model results to their input features [34]. With respect to our use case, engineers will be able to annotate local regions of interest within the analyzed feature space.

Many of the labeling and annotation approaches were built to train some kind of more or less explainable machine learning model, such as decision trees [44], or support vector machines [25] to name a few. However, especially when users are not familiar with machine learning, previously labeled and annotated data instances, may already provide valuable information to guide analyses. Thus instead of creating a "black-box" classifier [39], we will focus on the storage and availability of labels and annotations to guide user analyses.

2.3 Design Studies in the Manufacturing Sector

Design studies in the automotive sector are mostly carried out for engineering design and anomaly detection. In this context, efforts were carried out to visualize the exploration of multi-criteria alternatives for rotor designs [19], in-car communication networks [47, 48], or anomaly detection with test stations from large scale manufacturing processes [21, 53]. While some of the mentioned studies acknowledge the need for storing expert knowledge [21], the only systems we found which addresses the problem of specifically leveraging that kind of knowledge is *Cardiogram* [48]. *Cardiogram* addresses the

problem of debugging masses of traces from in-car communications networks to become error-free. In turn, the problem at hand which *IRVINE* addresses is the analysis of acoustic data, which necessitates different kinds of visualization approaches. Thus, grounded on previous findings, we did build a system that allows domain experts to externalize their knowledge from the analysis of acoustic data, supporting them in their high cognition task of analyzing engines.

3 METHODS

During this study, we primarily followed Sedlmair et al.'s nine-stage framework for design studies [49]. In addition, we used the *Nested Model* for visualization design and validation by Munzner [35]. The nested model guides a more detailed problem characterization, the data operation and abstraction, the visual encoding and interaction design, and the algorithm design.

Our system development went through four main iterations, during which we interviewed engineers, tested design alternatives, and held critical discussions with visualization experts. Each iteration was carried out in close collaboration with one engineer at BMW with extensive experience in the optimization and design of test procedures for *engines*. He accompanied the system development with knowledge about the problem domain (Section 4.2), the data abstraction (Section 4.3), and resulting tasks (Section 4.4). He also gave a constant stream of feedback on the visual design of our system. The exchange with the engineer took place on up to four meetings per week ranging from 30-60 minutes. During these meetings, fundamental characterization and design aspects were discussed and open issues clarified, while he provided us with feedback about relevant tasks and the system design. An evaluation of the system was carried out with four engineers at BMW. Based on the feedback from the evaluation, we refined *IRVINE*, which resulted in increased labeling and annotation speed with two more engineers. Methodological details for this downstream evaluation are provided in Section 8.

4 ABSTRACTIONS

We report the characterization of the problem domain in the field of quality control in the manufacturing sector. First, we provide an introduction to automated part testing in automotive engineering (Section 4.1), followed by a description of the domain of our collaborators (Section 4.2). Finally, we report on the task abstraction, supporting engineers with the analysis of error-prone engines (Section 4.4).

4.1 The Automotive Engineering Domain

The manufacturing of engines requires testing each engine in various stations along the assembly line. To that end, engineers analyze acoustic data by recording the noises of engines when being simulated with real-world conditions on a test bench. With *IRVINE*, we build upon the common case of acceleration tests, where a speed ramp is running from 2500 to 14000 revolutions per minute (rpm) to capture a sonic image of the engine at all relevant rotations. The measurement of these acoustic measurements enables the detailed analysis of engine sub-components (e.g. the gear or the bearing) and allows to allocate the primary symptom of an error (e.g. a gear with a scratched surface sounds different from an undamaged gear). According to the measurements of acoustic data, engineers can exactly identify the sub-components, which are known a priori from lab experiments carried out at earlier development stages.

What is challenging for engineers though, is the existence of secondary error mechanisms accompanying each specific primary error. In fact, fault mechanisms are visible in the primary error source, but also partially in their secondary error sources. The engineers reported that a good understanding of the interplay between primary and secondary errors sources is extremely valuable to further improve automated part testing with acoustic measurements. At BMW, engineers have developed a procedure for the analysis of relations between primary and secondary error symptoms, based on the systematic *correlations* of vast amounts of pairs of measurement values. Aggregated forms of these correlation results are called the **signature** of an engine. A detailed description of the signature computation process is outlined in Section 5.1.

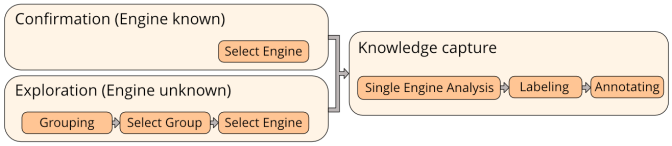


Fig. 2. Data analysis workflow of engineers. Engineers are either confirmers or explorers with the shared goal to select, analyze, label, and annotate a single engine.

Based on the observation of engineers and interviews conducted in the early stages of the project, we abstracted the principal workflow of engineers for the analysis of engines through acoustic data. In general, engineers switch between two main goals referring to (1) identifying engines of interest as well as (2) analyzing, labeling, and annotating engines. The workflow is shown in Figure 2, offering interesting nuances with respect to engine identification. Currently, the dominating information-seeking behavior [29, 51] of engineers is of confirmatory nature. Confirmers already know engines of interest, aiming for the validation of hypotheses about errors in an engine. In addition, the characterization of the domain problem in Section 4.2 shows the need for exploratory analysis support. Explorers are open to large varieties of engines and respective signatures, aiming at formulating new hypotheses about unknown engines and signatures. The workflow for unknown engines roughly ranges from multiple engines, over a small selection of engines, down to single engines of interest. Both explorers and confirmers share the same goal downstream: engineers review the signature of an engine in detail to allocate symptoms and compare it to the raw acoustic measurements to validate these symptoms. As a result, a label can be assigned to the engine. Finally, engineers also make annotations directly in the acoustic raw data to externalize knowledge about the source of an error.

4.2 Domain Problem

Both the confirmation and the exploration of engines as outlined in Figure 2 are time-consuming and tedious processes due to the following two main problems: First, large amounts of available signatures and recorded acoustic raw data measurements make it almost impossible for engineers to keep track of all engines at a desired level of detail. As a result, the detailed analysis, labeling, and annotation of engines can only be performed in an anecdotal manner. Only the confirmation of specific engines is supported, while an exploration of unknown signatures from unknown engines remains an open issue. Here, engineers reported that above all the efficient grouping of similar signatures from engines is not possible at the moment. The systematic analysis of all engines and error types falls short. Second, the knowledge, which is created during the detailed analysis of engines has a high impact on the improvement of automated part testing procedures. For engineers, it is especially useful to know, which error was observed in engines and where the source of the error can be seen in the data. However, occurred errors in engines are labeled too seldom by engineers, while the annotation of the cause of an error is not supported at all.

4.3 Data Abstraction

Engineers record their data inside a test bench with a sensor for measuring the noise of the engine. For that purpose, the engine's rpm is steered under controlled conditions to analyze the behavior of engines. Measuring the noise allows the evaluation of the acoustic properties of components and their technical condition. The result of such acoustic measurements is a three-dimensional data structure consisting of *loudness* measured across all possible combinations of *rpm* and *orders*. To shift focus on anomalies, the loudness values are often replaced by *residuals*, e.g., by the deviation from measured values to mean values μ of an ensemble of engines (expected value), divided by the standard deviation σ (see also eq. (1)) A 2D pixel-based visualization of these measurements often used by engineers is the **spectrogram**, as shown in Figure 3.

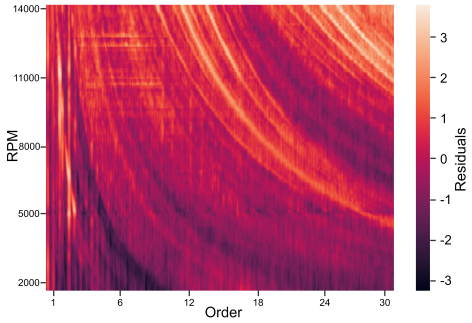


Fig. 3. Spectrogram of acoustic measurements for an engine. Rpm is shown on the y- and orders on the x-axis. The color indicates the volume of the measurement compared to the mean distribution of engines.

The **rpm** value is displayed on the y-axis, describing the acceleration of the engine up to a maximum speed. The **order** is displayed on the x-axis, which is the relation between a measured frequency and the speed of the engine during measurement, consequently describing how often an excitation occurs per revolution. Orders are beneficial, as they allow the fine-grained analysis of sub-components through an expert's eye. By analyzing the geometry of the rotating engine, engineers are able to derive the measured sub-component of an engine up to a very detailed level (e.g. 24th teeth of a gear). **Residuals** are shown on a color scale, where brighter colors represent particularly loud orders. According to the engineers, in serial manufacturing regions of loud frequencies are a good indicator for different types of errors. The three-dimensional measurement data forms the basic representation of an engine through abstract data, providing the raw data for any analysis scenario on engine errors in IRVINE.

Figure 3 shows a residual order-spectrogram. Specifically, 512 order lines (each column in Figure 3) ranging from 2500 - 14000 rpm are recorded for each engine. To identify signatures in order combinations that might result in a faulty engine, theoretically every possible order combination has to be analyzed manually for each engine. For engineers, this would result in 262,144 possible combinations, which is not feasible for a manual analysis. Hence, the engineers narrowed down the orders to the 41 most relevant ones. Based on the informed selection of orders, they can be connected to the seven main sub-components of an engine (*rotor-shaft-, electromagnetic-, first gear-, second gear-, A-bearing-, B-bearing-, and C-bearing orders*). Knowing the connected sub-components for individual orders allows the fine-grained analysis and labeling of engine errors at a sub-component level.

4.4 Task Abstraction

The intensive collaboration with engineers at BMW helped us to understand the domain, the domain problem, and the desired workflow for the analysis.In the following, we present the task abstraction that will support engineers in reaching their goals. Likewise, these tasks will serve as primary design targets for IRVINE.

$\mathbf{T}_1$ *Gain overview of engines*: By taking the role of explorers, engineers need a structured overview of the data, in our case of engines represented by their acoustic signals. Grouping engines by the similarity of acoustic signals using clustering algorithms would be desirable. In addition, interactive grouping to adapt to the information need of individual engineers would be useful, e.g., based on different steering parameters (such as the number of groups) or filtered subsets of engines.

$\mathbf{T}_2$ *Drill-down to engines*: Engineers exploring large numbers of engines need support for the drill-down to engines of interest. The information need may differ between clusters of engines, the labeling status, or the degree of the anomaly of engines.

$\mathbf{T}_3$ *Identify engine of interest*: The workflow of both explorers and confirmers includes the identification of single engines as an entry point into an in-depth analysis. Identification may be through knowledge about a particular engine, by special or even unique acoustic signatures, or by traversing several (similar) engines in the exploration process.

$\mathbf{T}_4$ *Analyze single engine*: Engineers need to analyze individual engines in detail for being able to assign labels on a profound basis. Single engines are both represented by their acoustic signature and the raw acoustic measurements. The analysis is also supported by stored domain knowledge from the systems knowledge base.

$\mathbf{T}_5$ *Assign label to engine*: Engineers need to assign labels of error categories. In close collaboration, we formed a default alphabet of label categories as follows: *Error/Electromagnetic-Field*, *Error/First Gear*, *Error/Second Gear*, *Error/A-Bearing*, *Error/B-Bearing*, *Error/C-Bearing*, *No error*. In addition, engineers also expressed the need to leave the label alphabet open for modifications, as knowledge about error variations will constantly grow as IRVINE is used. Finally, to enhance efficiency in combination with $\mathbf{T}_2$, it would be desirable to label single engines but also multiple similar engines at a glance.

$\mathbf{T}_6$ *Annotate acoustic measurements*: The cause of a labeled engine can be annotated by marking the respective region inside of the acoustic raw data of an engine (the spectrogram) serving two purposes. First, annotations can be used to review how similar labels were annotated by different users. Second, a sufficient amount of annotations for a given label will allow building thresholds for semi-automatic error detection.

5 Non-visual Data Analysis Support

In this section, we present the measurement of acoustic data and pre-processing steps, carried out to prepare the data. Based on the data abstraction (Section 4.3), we describe the computation of an engines signature (Section 5.1). Next, we describe details on the feature extraction and model training for our clustering approach (Section 5.2).

5.1 The Computation of Signatures

To analyze relations between primary and secondary symptoms of errors, engineers calculate correlations in-between spectrograms. The motivation is given in Section 4.3 and the schematic process for each engine is outlined in Figure 4. An exemplary process is described for a single signal pair (signal A and B) and one engine to increase clarity. This process is consequently applied to all available signals and engines.

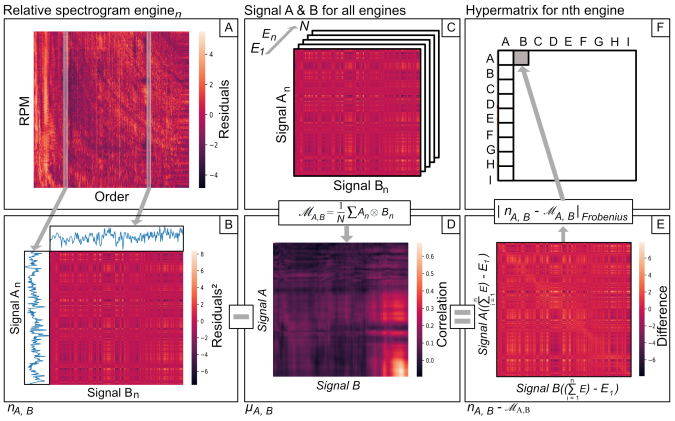


Fig. 4. Computation of our Hypermatrix. From a given spectrogram in (A), we extract two columns that are correlated with each other in (B). This is done for each signal pair over all engines in (C). Next, we subtract the resulting mean signal combination in (D) from each signal combination of a single engine which results in the deviation of a signal pair from one engine to all other engines in (E). Each signal pair is then aggregated and stored in a new matrix in (F).

As a first step, we calculate the mean and standard deviation for the ensemble of engines at hand. Consequently, the relative deviation of the i-th engine to the mean μ for each (rpm, order)-tuple is then expressed in units of the standard deviation σ:

$$\text{Residual}(ord, rpm)_i = \frac{\text{value}(ord, rpm)_i - \mu(ord, rpm)}{\sigma(ord, rpm)}. \quad (1)$$

An example of the resulting residual spectrogram described in Section 4.3 is given in (A). Next, we extract two measurements A and B (being 1D- curves each), and calculate the outer product for the pair, resulting in a 2D-matrix (see (B)). The choice of possible pairs is restricted to 41 relevant orders from the data abstraction, which are used to derive relevant order combinations. Calculating the mean value of the 2D-matrices over all engines for each entry (C) results in the correlation matrix (D). This correlation matrix effectively consists of Pearson correlations for pairs of rpm-values of two extracted orders. Therefore, the resolution regarding different engine speeds and the corresponding orders is still retained. To extract the difference in the correlation, we subtract the correlation matrix from each outer product resulting in a matrix describing correlations inside each measurement (E). This matrix is then reduced onto its cumulated deviation using the Frobenius-norm. Consequently, each cumulated deviation for each pair of orders is then ordered into the reduced-Difference-Correlation-Matrix, which we call **Hypermatrix**.

Figure 5 shows three exemplary *Hypermatrices* of different engines and their corresponding signature patterns. The first belongs to an *OK* engine with no errors. The second represents an engine with an anomaly in the B-Bearing but no error and the third an error in the B-Bearing. Each matrix was build with 41 signals, provided by our domain expert, which can be related to one of an engine's sub-components as described in the data abstraction. The region for each sub-component was marked by our domain expert with black lines. Hence, secondary error sources in sub-components can be identified by reviewing correlations in each region. Especially, in the second and third *Hypermatrix* one can clearly see which order is the primary error source in the signature and which orders resonate with the excitation thus representing secondary error sources.

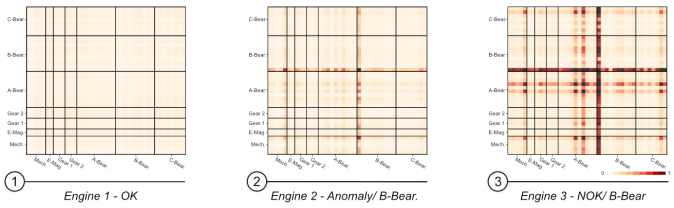


Fig. 5. Hypermatrices showing acoustic signatures. (1) with no error, (2) an anomaly in the B-Bearing, and (3) an error in the B-Bering.

5.2 Feature Engineering and Model Training

To group similar acoustic signatures, we extract features from each engine's *Hypermatrix*. The feature extraction process is shown in Figure 6. The combination of all seven sub-components results in 28 possible combinations and is depicted as Region (*R*) in Figure 6. From each *R*, the sub-matrix is extracted. After experimenting with different feature sets, we received the best clustering results extracting the *maximum* of each of the 28 resulting matrices. This results in a 28x1 feature vector.

The engineers agreed on the usefulness of this approach, as in their application domain, louder noises tend to be the cause for an error. However, different features can also be used as input by applying minor changes to the feature extraction process.

To cluster similar *Hypermatrices*, the 28x1 feature vector is used as input for a Self-Organizing Map (SOM) [32], as it nicely combines clustering with dimensionality reduction functionality. For the computation of the SOM, we follow a simplified version of the standard training process described by Kohonen [31]. We set the SOM grid size such as to expect at least one data vector per node, which accounts for very specific error types. We initialize the SOM prototype vector dimensions with random numbers between 0 and 1 and train the SOM by iterating over the input data vectors and adjusting the SOM nodes. Specifically, we find for each input data vector the best matching SOM prototype unit (BMU) according to *Euclidean* distance. We then adjust the BMU and its neighborhood according to a linearly decreasing learning rate and circular neighborhood kernel. This configuration comprises the initial implementation of our SOM and can be changed by the user as described in detail in Section 6.2. We note that our heuristic setting of parameters already gave us robust results for our application, hence, we did not see the need for parameter optimizations. In principle, also other visual clustering techniques may be applied besides SOM. We particularly chose SOM because of its robustness in our application domain, and as it gives an overlap-free rectangular layout that well supports visual comparison tasks.

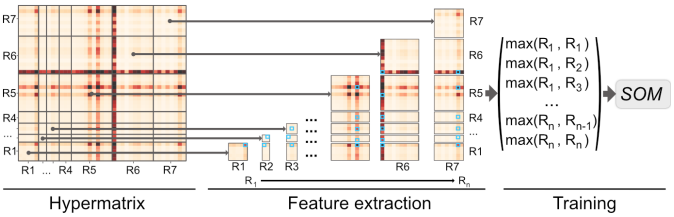


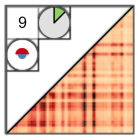
Fig. 6. Feature engineering process. All sub-component combinations in the in the *Hypermatrix* form 28 Regions (R) of interest. From each R, the sub-matrix is extracted and from each sub-matrix, the maximum is computed. All maximas of regions form the input vector for the SOM.

6 The IRVINE System

Figure 1 shows an overview of the main views of *IRVINE*, as also outlined in Section 6.1. Each view is marked from A-E, where we will use this notation hereafter to refer to *IRVINE's* individual views. Due to non-disclosure agreements with BMW, we are not able to show engine ids.

6.1 Overview of the System

In (A), all clusters of similar *Hypermatrices* as outlined in the Sections 5.1 and 5.2 are displayed. Each *Hypermtarix* in the grid is represented by the mean aggregation of all *Hypermatrices* in a cluster of engines. The grid view serves to get an overview over all clusters and their individual properties ($\mathbf{T}_1$) and to support the decision of which cluster to select ($\mathbf{T}_2$). To visually encode a *Hypermatrix*, we use a sequential color scale. This is appropriate since all *Hypermatrix* values range from 0-1, where 1 represents a high correlation between a pair of sub-components and 0 a low correlation.



Each grid in the cluster view also contains three additional rectangles. The first shows the number of engines in a cluster. The second represents the number of already labeled engines as a pie chart, where green are labeled engines and grey not labeled ones. The aggregated deviation of all engines in a cluster to the serial distribution of signatures from all available engines is displayed as two colored glyphs. Here, red arcs represent engines, which contain a deviation greater than zero and blue lower zero. This separation is important because engines that are deviating upwards are louder and downwards are quieter. Hence, users get an immediate overview of how many engines are in each cluster, how many engines are already labeled, and which cluster contains the most anomalous engines. In Figure 1 the selected cluster is shown by its grey stroke.

(B) shows engines in a cluster, the *Hypermatrix* of each engine as small multiple, and the aggregated deviation of the signature of a single engine to all other engines. This view is designed to get an overview of the engine's properties and compare them to other engines in the cluster. Thus, it supports the user in selecting an engine ($\mathbf{T}_3$). If the user has a specific engine id of interest the engine list can also be filtered accordingly and thus allows for confirmation as shown in Figure 2. The *Hypermatrix* of each engine is represented in the engine list view as a small multiple, while the same glyph representation as in the cluster view is used. Initially, the list is sorted according to the euclidean distance of engines to their centroids in each cluster. However, the user can sort the list according to the deviation of engines greater than zero. In Figure 1 the selected cluster contains 9 engines, sorted according to their anomaly score and contains similar *Hypermatrices*. By clicking on "Zoom in Cluster" a detailed overview of all *Hypermatrices* in the cluster replaces the view in (A), which is demonstrated in Figure 8-1.

Of course, users have the option to return to the initial view. The user can also retrain clusters by clicking on "Train new SOM". The process to train clusters is outlined in detail in Section 6.2.

In (C), the detailed *Hypermatrix* of a selected engine is shown. Additional information about single selected cells in the matrix is displayed in the upper left triangle. This view is designed to support the user in the analysis of a selected engine ($\mathbf{T}_4$). A matrix representation is adequate to represent the relation between pairs of sub-components (e.g. Gear and rotor shaft). As pointed out in Section 5.1, the regions of sub-components in the *Hypermatrix* are marked with additional black lines. The same color scheme as in the cluster view for *Hypermatrices* is applied. The selection of a cell in the *Hypermatrix* is supported by additional lines and triangles (dark and light grey) for each axis.

(D) visualizes a spectrogram of the selected engine according to Section 4.3. This type of visualization is the same one, engineers analyze during their daily routines and is used to support the analysis of a single engine ($\mathbf{T}_4$). Here, a diverging color scale is used. Blueish colors represent acoustic measurements, which are more quiet compared to all other engines, and reddish colors louder ones. This kind of color scale is appropriate since all values spread around 0 and deviate in different directions. By hovering over a cell in (C) the according orders of a pair of sub-components are displayed in the spectrogram. In the example in Figure 1, the sub-component pair (B-Bearing and A-Bearing) is selected. The former is shown with a triangle in dark grey and the latter with light grey. An engine can be annotated by clicking on "Add Annotation" ($\mathbf{T}_6$), which is outlined in detail in Section 6.4

(E) shows the distribution of already assigned labels as a bar chart. The main purpose of the view is to support the labeling of (selections of) engines ($\mathbf{T}_5$), as outlined in detail in Section 6.3. To show label distributions, a bar chart is an obvious choice. It would have been possible to use a pie chart, but for the labels that would have broken the guideline that there should be no more than six segments [24]. In the example in Figure 1, all but one engine in the cluster are labeled as B-Bearing error.

(F) shows three different views. First, the line chart displays two orders from the spectrogram across their rpm values. Second, the scatterplot shows the correlation of the selected order pair. Third, the bar chart shows the aggregated deviations for a region in the *Hypermatrix* to all other regions as indicated in Figure 6. These views are designed to facilitate the analysis of an engine ($\mathbf{T}_4$), where their input data are displayed when hovering over a cell in (C). In our application domain, deviations above and below three tend to be reasons for errors in the selected part. Thus, lines above and below this limit are marked as red for (+3) and blue for (-3). The purpose of the scatterplot is to provide additional information on how the two selected order lines correlate with each other. An additional overview about the five most deviating sub-component pairs in (B) is provided as a bar chart view in (E). To be consistent with our use of colors, red bars represent aggregated deviations greater than zero and blue ones lower zero.

6.2 Interactive Clustering

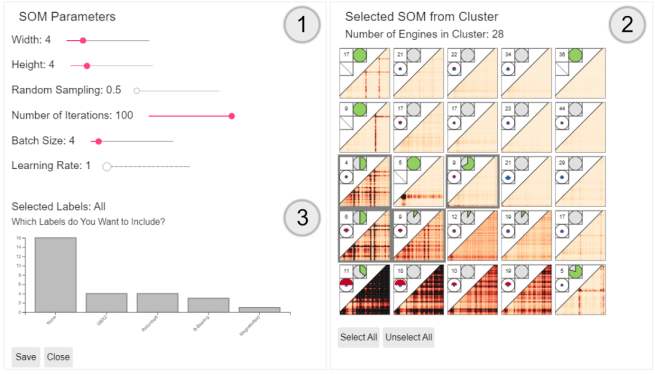


Fig. 7. Dialog to train SOM. Users can select SOM parameters in (1), choose input clusters in (2), and filter for relevant labels in (3).

To facilitate and speed up labeling, users can perform clustering based on a subset of engines in (A) - "Train new SOM". Figure 7 shows the dialog window with the three possible options for user interaction marked from 1-3. In (1), the user can define the input parameters to train a new SOM, such as batch size or learning rate.

As input data, the user can filter all engines based on their cluster (2) or label (3). In (2), the user selects engines by clicking on a cluster in the grid to form a subset of new engines. In (3), the user selects relevant labels of interest by clicking on a bar in the bar chart. After the parameterization, the SOM is trained as described in Section 5.2 and replaces (A) in Figure 1. However, users always can return the initial visualization.

6.3 Interactive Labeling

After an analysis is complete, the user can assign a label to the component in (E) with the two list views ($\mathbf{T}_5$). Labels can also have subcategories (e.g. *"B-Bearing/Inner ring"*). The user is able to create new categories and subcategories or update and delete them in (E). After a label is selected, users can either label a single engine or the entire cluster. However, in our design study, we experienced cluster labeling only for very small *not-OK* or large *OK* clusters. Provided labels serve for the retraining of the clustering as shown in Section 6.2 and are stored in the system's database to be available to other engineers. Entered labels further support three tasks. First, they give an additional overview over groups of engines ($\mathbf{T}_1$), because clusters which are already completely labeled are less interesting for an analysis. Second, they support the selection of an engine ($\mathbf{T}_3$), since engines that contain a label are also less probable to be selected. Third, they help in the analysis of a selected engine ($\mathbf{T}_4$). This is because they are immediately displayed in the engine list view in (B) and thus give hints on the probability of a label for the selected engine. If for example, 4 out of 5 engines in a cluster contain the same label, it is probable that the last engine also contains the same error and thus should be labeled equally.

6.4 Annotation of Sensor Data

When an engine is labeled, the cause of an error can further be annotated by the user in the spectrogram in (D). Here, the user can select the specific region in the spectrogram by a rectangle selection as shown in Figure 8-2 and -3. By dragging the edges of the rectangle, users are provided with feedback in a tooltip to which order the left and right side and to which rpm value the upper and lower side of the rectangle belongs. As well as labels, annotations can support analyses in multiple ways.

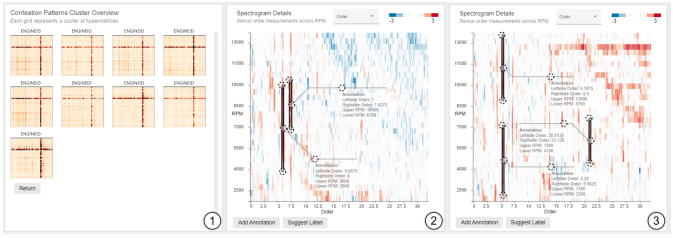


Fig. 8. (1) Drill-down of Hypermatrices from the selected cluster in Figure 1(A) and errors in the A-Bearing in (2) and a B-Bearing in (3) with according annotations.

First, users are able to review annotations from previous analyses of other engineers. By selecting a label from the bar chart view in Figure 1-E all other annotations for the same label are displayed in the spectrogram. This supports the user in the analysis of an engine ($\mathbf{T}_4$).

Second, when an engine was annotated, the user can request a label and annotation suggestion from the system. Here, all labels and respective annotations are queried from the database. Since each annotation exactly specifies the range of columns (Left Order/ right Order) and rows (Upper rpm/ Lower rpm), it allows to narrow down the search space for an anomaly in the spectrogram. Next, the three annotations and according labels with the highest threshold violations are displayed to the user. If there are no violations, the system suggests that the selected engine seems to be *OK*. We choose to give the user

only the opportunity to request an annotation from the system after an annotation input was manually made by the user. With this design choice we intent to avoid blind trust in a system recommendation and thus a decision bias. The more annotations are made, the better the suggestion of further annotations becomes. All labels and annotations are stored in the system's database and can be used for different purposes, such as model training. Thus, a continuous stream of knowledge from domain experts is stored [36] with *IRVINE* via labeling and annotating.

6.5 Implementation

The system is a single-page web application written in Typescript, HTML, and CSS using the framework *Angular Js*. All views are based on *D3.js* [14]. To improve rendering speed, all SVGs are rendered as canvas elements. *IRVINE* runs on a Docker Container on a virtual machine, so that each employee inside the BMW network can access it via a public URL. The input data is processed in a separate Python application and stored in a SQL Database. All API calls run on a separate Python Flask application hosted on a virtual machine from BMW.

7 Usage Scenarios

This section outlines two usage scenarios for the interactive clustering and labeling with *IRVINE*, in line with the groups of required functionalities of Section 4.4. The first scenario (Section 7.1) shows how a new clustering can be performed to retrieve a group of similar signatures. The goal hereby lies in the fast detection of interesting signatures for labeling. The second scenario (Section 7.2) shows how users perform a detailed analysis of an engine, provide a label for a B-Bearing error, and annotate the respective acoustic measurement.

7.1 Scenario 1: Interactive Clustering

The engineer Alexandra is an explorer. For her analysis, she has 434 engines available, all of which are unknown to her at the start. Her goal is to assign labels to unlabeled engines ($\mathbf{T}_5$). She starts IRVINE, leading to the analysis state as shown in Figure 1 with a SOM clustering with 5x5 grid cells. Alexandra is interested in engines with a clear signature. She selects four grid cells with an overall of 28 engines and four different labels, namely the second gear (GBX2), the rotor shaft, the B-Bearing, and the magnetic field, as shown in Figure 7-3. To gain an overview of the characteristics of only these 28 engines ($\mathbf{T}_1$), she decides to re-train clustering with 4x4 cells only using these engines. Due to the small number of selected engines, she chooses a relatively small batch size, while for the other parameters she keeps the default settings.

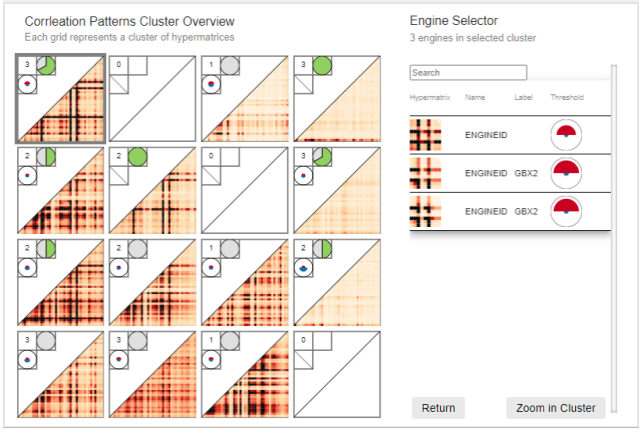


Fig. 9. Clusters resulting from a SOM training with 28 engines. The selected cluster contains three engines, where two are already labeled with an error in the second gear.

Alexandra analyzes the clustering result as shown in Figure 9 and realizes that, as she expected, all clusters contain a small number of engines between zero and three engines. In this way, she can focus on very small subsets of engines, when browsing through individual grid cells. At a glance, she identifies many different signatures, while stronger errors (with a high degree of red colors) align on the left of the SOM. Interestingly, the two clusters with the most apparent signatures were already labeled by other engineers before.

She decides to browse further and selects the top left cluster ($\mathbf{T}_2$), containing engines with well-distinguishable signatures. Two of the three engines already contain labels, hinting at errors in the second gear. The *Hypermatrix* of the unlabeled engine in the engine list view in Figure 9 is very similar to the labeled engines. Alexandra thus comes to the conclusion to label this engine also with an error in the second gear ($\mathbf{T}_5$). The proper validation of an assigned label requires a detailed analysis of the selected engine. This scenario is outlined in Section 7.2.

7.2 Scenario 2: Interactive Labeling and Annotating

The engineer Thomas is a confirmer. He uses the same data set as Alexandra but wants to analyze engines with a specific signature he discovered in previous analyses ($\mathbf{T}_4$). The cluster with the interesting signature is outlined in Figure 1 with a grey stroke in (A). For this cluster, only one engine has not been labeled yet, which is why Thomas selects this engine for a detailed analysis ($\mathbf{T}_3$). Being a confirmer, Thomas has a hypothesis that *an error in the B-Bearing can also be seen in the acoustic measurement of an A-Bearing*. By reviewing the *Hypermatrix* of the selected engine in Figure 1 (C), Thomas notices that in fact there seems to be an anomaly originating in the B-Bearing, indicated by the dark red colors. This assumption is supported by reviewing the bar chart in (F), where also the B-Bearing is marked as the biggest anomaly in the engine. By hovering over the *Hypermatrix*, he selects orders that can be related to the inner ring of the B-Bearing and the outer ring of the A-Bering as shown in the upper white triangle in (C). He notices that in the line chart and scatterplot the selected order line for the inner ring of the B-Bearing is above a threshold of three, which is marked as red in (F). He can also see that there seems to be a week negative linear relationship between the inner ring of the B-Bearing and the outer ring of the A-Bearing. Thus, he decides to label the engine as B-Bearing error ($\mathbf{T}_5$), which is supported by the fact that all other engines in the cluster are also labeled as B-Bearing error as shown in the engine list view in (B). In the spectrogram in (D) the region where the order of the inner ring of the B-Bearing is shown with a black line and dark grey triangle, he identifies the highest residual values. Therefore, he clicks on "Add Annotation" in (D) and annotates the respective region of the error ($\mathbf{T}_6$). He now confirms his hypothesis that in fact for the sample of 434 engines, B-Bearing errors can also be seen in acoustic measurements of A-Bearings.

8 User Evaluation

In this section, we introduce our evaluation methodology (Section 8.1) and present our study findings (Section 8.2).

8.1 Methodology

Our evaluation has two goals. First, to validate the usefulness and usability of the proposed technical considerations and the resulting visualization in terms of effectiveness for automotive engineers. Second, to evaluate if labeling speed increased after implementing the user feedback from the first evaluation round as part of our iterative design process. The analysis of complex data, where domain knowledge is essential is a high-level cognitive task. Such tasks are however difficult to measure quantitatively and objectively [54]. As for real-world scenarios, data, users, and tasks are important, we perform a qualitative field study to evaluate the usefulness and usability of *IRVINE*. In this type of study, qualitative coding of user feedback is combined with a quantitative usability scale [19]. To evaluate, whether labeling speed increased after including user feedback, we measure how many labels and annotations users made by using the system for twenty minutes. We then compare the results to self-reported labeling speed before the system introduction, before we implemented user feedback, and after the user feedback.

Participants: The study was carried out with six engineers (others than the lead user) from BMW, responsible for the development of testing procedures for the manufacturing of electrical vehicles. They were all male between 23 and 33 years old and had a mean working

experience of 4 years in the problem domain, all with a background in mechanical engineering. Inside BMW only very few engineers with a sufficient level of knowledge about the analysis of acoustic measurements of engines exist. Thus, only a low number of potential candidates are able to properly evaluate *IRVINE*. However, this is rather common in design studies, where presented visualizations often tackle very specific problems, which can be addressed by only a few users [11, 19, 47, 48].

Data: For the first round of interviews, we used acoustic data from 434 randomly selected engines over a period of six months. The data did not contain any labels nor annotations. For the second round of interviews, acoustic data from 308 completely new engines from a period of four months were uploaded to the system.

Task: The following task was given to each user: *"Please find error-prone engines and provide labels and annotations for each engine."* Engineers had 20 minutes to find as many error-prone engines as possible. An exemplary execution that we observed during the development with our lead engineer can be the following: First, select a cluster and then an engine from the cluster. Next, analyze the engines' *Hypermatrix*, by hovering over its cells and find a sub-component pair of interest. Then, make a hypothesis, for example, *the resonance of the first gear can also be observed in the rotor-shaft*. Next, inspect the spectrogram, the line charts, the scatterplot, and the bar chart, to accept or reject this hypothesis. If an engine contains an error, select a label from existing labels in the engine list view or create and save a new label as free text and annotate its cause in the spectrogram.

Procedure: As the concept of our *Hypermatrix* and resulting clustering is rather complex, a system introduction was carried out in a kick-off group session. Here, two researchers were present, one tacking notes and the other explaining the system components to the engineers. The session took place online and lasted for 80 minutes. Next, interviews were scheduled with each participant. To evaluate the usability and usefulness of *IRVINE*, four interviews were conducted in the form of a think-aloud session [52]. Each session took on average 60 minutes and involved a short walk-through of the system, open-ended questions [45] about the usage, and a usability questionnaire. All interviews were held online, where each engineer executed the same predefined task. The notes from the kick-off and think-aloud study were analyzed using a qualitative coding methodology [18]. Repeated ideas or statements in the feedback were assigned with codes extracted from the data. Afterward, the codes were grouped into abstract categories to summarize the study results that are also aligned to a set of questions proposed by Lam et al. [33], in the context of user experience. To quantitatively assess the usability of our system, we applied the *System Usability Scale (SUS)* [45]. Due to our small sample size, the SUS scale does not provide empirical evidence of the usability of our visualization but rather a rough direction to support our assumptions of our design choices. In addition, we measured how many labels and annotations were made by the users during the task execution. To evaluate if labeling speed did improve after the implementation of the user feedback, additional two interviews were conducted in the form of a think-aloud session [52]. Here, engineers executed the previously defined task, where we again measured how many labels and annotations were made.

8.2 Findings

We observed that all engineers used the general system workflow as outlined in Section 8.1. First, all engineers selected a cluster, where one engineer also retrained a new SOM and noted, *"I want to have a more detailed view from these similar clusters to better see emerging signatures"*. Next, engineers selected an engine from the engine list. Here, one engineer also did zoom in the cluster, noting that *"for me it is easier, to see all Hypermatrices in one view, to choose a relevant engine"*. All engineers noted that the engine list is very helpful to identify relevant engines. One expert also noted that *"It is good to know that I can filter this list, because I often have specific engine IDs beforehand, which I need to evaluate in detail"*.

Next, engineers turned to the *Hypermatrix* and specifically looked for cross structures in the matrix. This was reported as relevant, where one engineer explained *"This view helps to evaluate how this resonance affects other parts of an engine"*. Next, engineers made hypotheses regarding the signatures of engines and verified them using the spectrogram, the line charts, or the scatterplots. Here, the use of residual values instead of absolute values in decibel was noted to be especially helpful (*"Normally I have to compare two engines, where it is often tedious to find a perfect engine as ground truth."*)

Engineers then assigned labels to the selected engines, where two of them created new label categories in the list view. Three engineers also requested a label from the system after their label input and in all three cases agreed with the systems' suggestion. Next, engineers annotated the cause for the detected error in the spectrogram and reported that it is very helpful that their previous inputs are stored in the system to guide their analyses. One engineer also mentioned that this kind of knowledge helps for discussions with other stakeholders, such as analysts ("It is good to either print it out or use the tool itself to show the analysis to other non-domain related stakeholders").

Engineers also recommended some system improvements, which we implemented in a fourth development cycle in our system. Apart from minor issues, for example, small font sizes, engineers requested to include subcategories for each label. Furthermore, they requested features to better detect relevant engines of interest. This resulted for example in the creation of pie charts that show how many labels exist in each cluster or glyphs, which show the anomaly score of clusters and engines as one can see in Figure 1.

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
Expert 1	7.5	10	7.5	5	7.5	7.5	10	10	5	7.5	77.5
Expert 2	7.5	10	7.5	10	7.5	10	5	10	7.5	10	85
Expert 3	10	10	7.5	10	7.5	10	10	7.5	10	10	92.5
Expert 4	7.5	7.5	7.5	7.5	7.5	10	10	7.5	10	10	85
Avg.	8.1	9.4	7.5	8.1	7.5	9.4	8.8	8.8	8.1	9.4	85

Table 1. Results of the System Usability Scale [45] with four engineers.

Considering the quantitative results of the usability survey, our system provides excellent usability according to the adjective equivalent of the achieved SUS score [5]. We found that our systems' usability is with 85 highly above the average score of 68 [45]. The individual scores are outlined in Table 1. Even though we evaluated the system with only four engineers, we were confident that our system did reach a sufficient level of usability and thus focused in later interviews only on evaluating labeling speed. This assumption was supported since the remaining two engineers did not report any recommendations to improve the system.

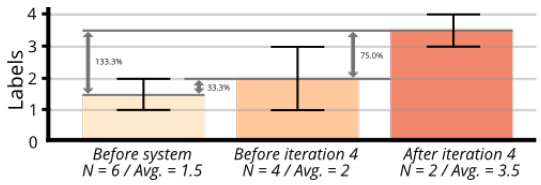


Fig. 10. Improvement of labeling speed compared to the status quo before the introduction of IRVINE, before Iteration 4 and after Iteration 4.

Figure 10 outlines the improvement of labeling speed, which was reached by using *IRVINE*. Before the systems' introduction engineers reported to be able to provide one or two labels in a manual analysis in twenty minutes. However, they noted that it is rather difficult to provide an exact number of labeling speeds since errors and resulting analyses differ significantly. Here, one engineer noted *"There are errors, which can be detected in minutes and others that take up to two hours. However, generally speaking, I normally can provide up to three labels within an hour"*. Even though this number of labels is rather vague, we can use it as a rough direction on how *IRVINE* improved labeling speed.

During our study, we were able to measure the exact number of provided labels during the task execution. As Figure 10 shows, engineers were able to label engines 33.3% faster than before the system introduction and 75.0% after we included the user feedback. If we compare labeling speed to the final system, we can see an improvement of 133%. Even though engineers noted that each analysis heavily depends on the

engine and thus labeling speed is generally speaking hard to quantify, we state that with *IRVINE* labeling speed, however, improved.

9 Discussion

In this design study, we contribute the problem characterization and abstraction of the analysis of acoustic signatures in the manufacturing of electrical engines. We further report the interactive design of the presented VA system *IRVINE*, which we evaluated together with six automotive engineers. Our design study represents a very detailed view of the problem domain. From a more abstract point of view, we integrated a VA system into the workflows of domain experts to support them with the interactive clustering and labeling of sensor data. To the best of our knowledge, the problem we described in Section 4.2 is unique inside BMW and has not been addressed by other researchers so far. We thus argue that a design study with a detailed analysis of the problem domain and resulting abstractions was necessary to successfully support domain experts with their very specific tasks.

The overall success of our approach is demonstrated by the fact that study participants rated *IRVINE* with a high usability and usefulness in combination with an increased labeling speed. Our evaluation, however, had a number of limitations. Probably the most important one is the fairly small sample size of participants. However, as mentioned in Section 8.1 it is rather unusual to build solutions for specific domain problems, which address a community bigger than a couple of dozen users. A second limitation is the quantification of labeling speed. As mentioned in our Evaluation, almost every engine has to be treated more or less differently. Especially regarding electrical engines, where product behavior is not that well understood compared to more mature technologies, such as petroleum engines, it is thus hard to determine beforehand how long it will take to produce a label or to carry out a detailed analysis. Nevertheless, we believe that our system increases labeling and analysis speed compared to a manual analysis of the data and serves well to better detect and understand errors in engines.

Although the design was specific to a particular domain, there are some aspects that provide guidance to the design to other domains. Some of that guidance results in the following suggestions to transform high-dimensional data and design similar systems:

1) Cluster Signatures: The SOM visualization helps engineers to immediately detect clusters of similar signatures. The provided small multiples also support the decision, which clusters to analyze first. The detection of a group of potential anomalous engines was known to be an important problem, but before the introduction of *IRVINE*, engineers had to manually compare single engines in a slow error-prone process. In contrast, our SOM visualization allows for a robust and fast identification of relevant engines of interest. The benefits of SOM grid visualizations were previously noted in a use case to analyze speech signals [42].

2) Use Hypermatrices: The *Hypermatrix* view helps experts in the fast allocation of local anomalies in engines. Our approach to compute and visualize *Hypermatrices* can further be used to design systems, where similar data to spectrograms are used as input data. For example, apart from our application domain, spectrograms are applied in the analysis and synthesis of speech signals [23], seismic activities [17], or the medical sector [26], where often sonograms - a similar representation - are applied.

3) Combine Labeling and Annotations: Apart from providing labels for single data instances, as shown in other approaches [44, 58], annotating specific regions directly in matrix views of high-dimensional data can provide more detailed information about the cause of an error. One might argue that suggested labels and annotations from the system are preferable to omit human biases. In our case, however, the primary goal is not only to detect errors but also to understand the reason for their occurrence. This kind of gained knowledge [9] is more relevant than algorithms, which are maybe capable to detect errors fast, but often remain a black box [39]. Thus, we believe that is important to keep experts in an active role in high stakes decision making instead of degrading them to validate or reject model suggestions [2].

4) Make Externalized Knowledge Available: We present guided workflows with our visualization on how to externalize and store knowledge. Here, matrix views, line charts, bar charts, and scatterplots, helped in creating and validating hypotheses. *IRVINE* aggregates knowledge in the form of a label and annotations. This knowledge does not only help domain-related experts to guide their analyses but also other stakeholders outside the application domain [22]. Labels, for example, can be queried by analysts to train machine learning classifiers and annotations can provide valuable insights on the relevant feature space of each label.

We believe that these four recommendations can help other researchers when investigating complex data or domain problems. For instance, Kim et al. [30] and Brattain et al [15] both performed studies on the analysis of sonogram data. To identify secondary symptoms of illnesses they can also compute *Hypermatrices* as demonstrated in our study. Theses *Hypermatrices* could then be clustered by using a similar SOM implementation as demonstrated in Section 5.2. Resulting similar clusters can then be labeled by medical experts and regions of interest directly annotated in the sonograms. This externalized knowledge aids to train other medical experts in the analysis of sonogram data.

The design also had some limitations, which we briefly summarize here. In our visualization, we use data of only one sensor in one test bench. However, using data from different test stations or sensors provides new challenges for visualization designs. One challenge for example is the right mapping of produced components to their recorded data across multiple stations and the visualization of multiple test stations. A solution for this can be the detection of anomalies for multiple stations with network views, and the visualization of according *Hypermatrices* and spectrograms in separate views.

Further, annotations are limited to a rectangular selection. In our case, this is appropriate, since data in the spectrogram has either a vertical or horizontal relation. In use cases where patterns are vertically and horizontally dependent at the same time, annotating data via lasso functionalities might be more adequate.

Finally, to get a comprehensive overview for the *Hypermatrix*, our lead engineer narrowed down 41 relevant orders from the 512 available ones in the spectrograms. This effort was carried out manually and based on years of experience of the engineer in the problem domain. However, this effort can be supported automatically, for example, by computing most deviating orders of a sample of spectrograms to support the selection of important orders before computing the *Hypermatrix*.

10 Conclusion and Future Work

This paper presents a design study on the development of a visualization approach to analyze signatures and acoustic measurements of engines. The resulting VA system *IRVINE* leverages interactive data labeling and clustering approaches to facilitate the analysis of high amounts of acoustic data to detect and understand previously unknown errors. *IRVINE* comprises five different core visualizations. (1) The cluster view and an engine list give an overview over similar signatures of engines. Furthermore, they allow for drilling down to a group of engines and the selection of an engine of interest. *IRVINE* also allows a more fine-grained clustering by retraining the initial SOM based on a selection of sub-clusters and labels. (2) The *Hypermatrix* view allows the analysis of signatures and the creation of hypotheses regarding anomalies in the data. (3) The spectrogram view represents the ground truth of the data to validate previously made hypotheses. (4) Line charts, scatterplots, and bar charts, map *Hypermatrix* selections to spectrograms. (5) Labeling and annotation features allow users to tag specific engines and store knowledge in the system. *IRVINE* evolved iteratively, where we closely worked with engineers from BMW, tested design alternatives, and held critical discussions. The success of our design is shown by the high usability scores, high reported usefulness, and an increased labeling and annotation speed.

There are several avenues for our research. One is to investigate the connection of sensor data across multiple testing stations and the resulting challenges for future visualizations. A second one is to investigate how externalized knowledge enables other stakeholders outside the application domain, for example, data analysts, to improve their work. Finally, the effectiveness of our visualization concept should be investigated in other application domains.

REFERENCES

[1] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Springer, 1st ed., 2011.

[2] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 35(4):105–120, 2014.

[3] G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, and F. Giannotti. Interactive visual clustering of large collections of trajectories. In *Visual Analytics Science and Technology*, pp. 3–10, 2009.

[4] B. Bach, N. Henry-Riche, T. Dwyer, T. Madhyastha, J.-D. Fekete, and T. Grabowski. Small multipiles: Piling time to explore temporal patterns in dynamic networks. *Computer Graphics Forum*, 34, 2015.

[5] A. Bangor, P. Kortum, and J. Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of Usability Studies*, 4:114–123, 2009.

[6] M. Behrisch, F. Korkmaz, L. Shao, and T. Schreck. Feedback-driven interactive exploration of large multidimensional data supported by visual classifier. In *Visual Analytics Science and Technology*, pp. 43–52, 2014.

[7] S. Berg, D. Kutra, T. Kroeger, C. Straehle, B. Kausler, C. Haubold, M. Schiegg, J. Ales, T. Beier, M. Rudy, K. Eren, J. Cervantes, B. Xu, F. Beuttenmueller, A. Wolny, C. Zhang, U. Köthe, F. Hamprecht, and A. Kreshuk. ilastik: interactive machine learning for (bio)image analysis. *Nature Methods*, 16:1–7, 2019.

[8] J. Bernard, D. Daberkow, D. Fellner, K. Fischer, O. Koepler, J. Kohlhammer, M. Runnwerth, T. Ruppert, T. Schreck, and I. Sens. Visinfo: a digital library system for time series research data based on exploratory search—a user-centered design approach. *International Journal on Digital Libraries*, 16(1), 2015.

[9] J. Bernard, M. Hutter, M. Zeppelzauer, D. Fellner, and M. Sedlmair. Comparing visual-interactive labeling with active learning: An experimental study. *Transactions on Visualization and Computer Graphics*, PP:1–1, 2017.

[10] J. Bernard, C. Ritter, D. Sessler, M. Zeppelzauer, J. Kohlhammer, and D. Fellner. Visual-interactive similarity search for complex objects by example of soccer player analysis. In *Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 3, pp. 75–87, 2017.

[11] J. Bernard, D. Sessler, J. Kohlhammer, and R. A. Ruddle. Using dashboard networks to visualize multiple patient histories: A design study on post-operative prostate cancer. *Transactions on Visualization and Computer Graphics*, 25(3):1615–1628, 2019.

[12] J. Bernard, N. Wilhelm, B. Krüger, T. May, T. Schreck, and J. Kohlhammer. Motionexplorer: Exploratory search in human motion capture data based on hierarchical aggregation. *Transactions on Visualization and Computer Graphics*, 19(12):2257–2266, 2013.

[13] J. Bernard, M. Zeppelzauer, M. Sedlmair, and W. Aigner. Vial: a unified process for visual interactive labeling. *The Visual Computer*, 34, 2018.

[14] M. Bostock, V. Ogievetsky, and J. Heer. D-3: Data-driven documents. *transactions on visualization and computer graphics*, 17:2301–9, 2011.

[15] L. J. Brattain, B. A. Telfer, M. Dhyani, J. R. Grajo, and A. E. Samir. Machine learning for medical ultrasound: status, methods, and future opportunities. *Abdominal Radiology*, 43(4):786–799, 2018.

[16] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-function: Learning distance functions interactively. In *Visual Analytics Science and Technology*, pp. 83–92, 2012.

[17] V. W. Chao, Y. Wu, L. Zhao, V. Tsai, and C.-H. Chen. Seismologically determined bedload flux during the typhoon season. *Scientific Reports*, 5, 2015.

[18] K. Charmaz. *Constructing grounded theory : a practical guide through qualitative analysis*. Sage Publications, London; Thousand Oaks, Calif., 2006.

[19] L. Cibulski, H. Mitterhofer, T. May, and J. Kohlhammer. Paved: Pareto front visualization for engineering design. *Computer Graphics Forum*, 39:405–416, 2020.

[20] J. J. Dudley and P. O. Kristensson. A review of user interface design for interactive machine learning. *Transactions on Interactive Intelligent Systems*, 8(2), 2018.

[21] J. Eirich, D. Jäckle, T. Schreck, J. Bonart, O. Posegga, and K. Fischbach. Vima: Modeling and visualization of high dimensional machine sensor data leveraging multiple sources of domain knowledge. In *Visualization in Data Science at IEEE VIS*, 2020.

[22] J. Eirich, D. Jäckle, S. Werrlich, and T. Schreck. Visual analytics in organizational knowledge creation: A case study. In *European Conference on Information Systems*, 04 2021.

[23] J. L. Flanagan. *Speech analysis; synthesis and perception*. Springer, 2nd ed., 1972.

[24] M. Hardin, D. Hom, and L. Williams. Which chart or graph is right for you?, 2016.

[25] F. Heimerl, S. Koch, H. Bosch, and T. Ertl. Visual classifier training for text document retrieval. *Transactions on Visualization and Computer Graphics*, 18(12):2839–2848, 2012.

[26] S. Hughes. Medical ultrasound imaging. *Physics Education*, 36:468, 2001.

[27] B. Höferlin, R. Netzel, M. Höferlin, D. Weiskopf, and G. Heidemann. Inter-active learning of ad-hoc classifiers for video visual analytics. In *Visual Analytics Science and Technology*, pp. 23–32, 2012.

[28] A. Kampker, K. Kreiskother, N. Lutz, V. Gauckler, and M. Hehl. Re-ramp-up management of scalable production systems in the automotive industry. In *Industrial Technology and Management*, pp. 137–141, 2019.

[29] D. A. Keim. Information visualization and visual data mining. *Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.

[30] S. Kim, B. K. Seo, J. Lee, K. Cho, K. Lee, B.-K. Je, H. Y. Kim, Y.-S. Kim, and J.-H. Lee. Correlation of ultrasound findings with histology, tumor grade, and biological markers in breast cancer. *Acta oncologica*, 47:1531–8, 2008.

[31] T. Kohonen. Essentials of the self-organizing map. *Neural Networks*, 37:52–65, 2013.

[32] T. Kohonen, M. R. Schroeder, and T. S. Huang. *Self-Organizing Maps*. Springer, Berlin, Heidelberg, 3rd ed., 2001.

[33] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical studies in information visualization: Seven scenarios. *Transactions on visualization and computer graphics*, 18, 2011.

[34] J. Moehrmann, S. Bernstein, T. Schlegel, G. Werner, and G. Heidemann. Improving the usability of hierarchical representations for interactively labeling large image data sets. In *Human computer interactions*, p. 618–627. Springer, Berlin, Heidelberg, 2011.

[35] T. Munzner. A nested model for visualization design and validation. *Visualization and Computer Graphics*, 15:921 – 928, 2010.

[36] I. Nonaka and H. Takeuchi. *The knowledge-creating company: How japanese companies create the dynamics of innovation*. Oxford University Press, New York, 1995.

[37] A. Pister, P. Buono, J.-D. Fekete, C. Plaisant, and P. Valdivia. Integrating prior knowledge in mixed initiative social network clustering, 2020.

[38] C. Ratanamahatana, J. Lin, D. Gunopulos, E. Keogh, M. Vlachos, and G. Das. *Mining Time Series Data*, pp. 1049–1077. Springer, 2010.

[39] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, 2019.

[40] T. Ruppert, M. Staab, A. Bannach, H. Lücke-Tieke, J. Bernard, A. Kuijper, and J. Kohlhammer. Visual interactive creation and validation of text clustering workflows to explore document collections. *Electronic Imaging*, 2017(1):46–57, 2017.

[41] D. Sacha, Y. Asano, C. Rohrdantz, F. Hamborg, D. Keim, B. Braun, and M. Butt. Self organizing maps for the visual analysis of pitch contours. *Computational Linguistics*, 2015.

[42] D. Sacha, M. Kraus, J. Bernard, M. Behrisch, T. Schreck, Y. Asano, and D. A. Keim. Somflow: Guided exploratory cluster analysis with self-organizing maps and analytic provenance. *Transactions on Visualization and Computer Graphics*, 24(1):120–130, 2018.

[43] D. Sacha, L. Zhang, M. Sedlmair, J. A. Lee, J. Peltonen, D. Weiskopf, S. C. North, and D. A. Keim. Visual interaction with dimensionality reduction: A structured literature analysis. *Transactions on Visualization and Computer Graphics*, 23(1):241–250, 2017.

[44] A. Sarkar, M. Spott, A. Blackwell, and M. Jamnik. Visual discovery and model-driven explanation of time series patterns. In *Visual Languages and Human-Centric Computing*, pp. 78–86, 2016.

[45] J. Sauro. Measuring usability with the system usability scale (sus), Accessed 05.11.2020.

[46] T. Schreck, J. Bernard, T. von Landesberger, and J. Kohlhammer. Visual cluster analysis of trajectory data with interactive kohonen maps. *Information Visualization, Palgrave Macmillan*, 8(1):14–29, 2009.

[47] M. Sedlmair, A. Frank, T. Munzner, and A. Butz. Relex: Visualization for actively changing overlay network specifications. *Visualization and Computer Graphics*, 18:2729–2738, 2012.

[48] M. Sedlmair, P. Isenberg, D. Baur, M. Mauerer, C. Pigorsch, and A. Butz. Cardiogram: Visual analytics for automotive engineers. In *Human Factors in Computing Systems*, pp. 1727–1736, 2011.

[49] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *Visualization and Computer Graphics*, 18:2431–2440, 2012.
[50] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
[51] B. Shneiderman. Inventing discovery tools: Combining information visualization with data mining. *Information Visualization*, 1(1):5–12, 2002.
[52] M. Someren, Y. Barnard, and J. Sandberg. *The Think Aloud Method - A Practical Guide to Modelling CognitiveProcesses*. Academic Press, 1994.
[53] J. Suschnigg, B. Mutlu, A. Fuchs, V. Sabol, S. Thalmann, and T. Schreck. Exploration of anomalies in cyclic multivariate industrial time series data for condition monitoring. In *Big Data Visual Exploration and Analytics*, 2020.
[54] M. Tory and T. Möller. Evaluating visualizations: Do expert reviews work? *Computer graphics and applications*, 25:8–11, 2005.
[55] J. Vesanto. Som-based data visualization methods. *Intelligent Data Analysis*, 3(2):111–126, 1999.
[56] N. Weber, M. Waechter, S. C. Amend, S. Guthe, and M. Goesele. Rapid, detail-preserving image downscaling. *Transactions on Graphics*, 6, 2016.
[57] N. Wilhelm, A. Vögele, R. Zsoldos, T. Licka, B. Krüger, and J. Bernard. Furyexplorer: Visual-interactive exploration of horse motion capture data. *Visualization and Data Analysis*, 2015.
[58] D. Wu, X. Wang, J. Su, B. Tang, and S. Wu. A labeling method for financial time series prediction based on trends. *Entropy*, 22(10), 2020.
[59] W. Yang, X. Wang, J. Lu, W. Dou, and S. Liu. Interactive steering of hierarchical clustering. *Transactions on Visualization and Computer Graphics*, 2020.
[60] J. Zhao, M. Karimzadeh, A. Masjedi, T. Wang, X. Zhang, M. M. Crawford, and D. S. Ebert. Featureexplorer: Interactive feature selection and exploration of regression models for hyperspectral images. In *Visualization Conference*, pp. 161–165, 2019.