Zschech, Patrick; Horn, Richard; Höschele, Daniel; Janiesch, Christian; Heinrich, Kai

Intelligent User Assistance for Automated Data Mining Method Selection

Business & Information Systems Engineering

Provided in Cooperation with:
Springer Nature

Suggested Citation: Zschech, Patrick; Horn, Richard; Höschele, Daniel; Janiesch, Christian; Heinrich, Kai (2020) : Intelligent User Assistance for Automated Data Mining Method Selection, Business & Information Systems Engineering, ISSN 1867-0202, Springer Fachmedien Wiesbaden, Wiesbaden, Vol. 62, Iss. 3, pp. 227-247, https://doi.org/10.1007/s12599-020-00642-3

This Version is available at:
https://hdl.handle.net/10419/289170

Bus Inf Syst Eng 62(3):227–247 (2020)
https://doi.org/10.1007/s12599-020-00642-3

RESEARCH PAPER

Intelligent User Assistance for Automated Data Mining Method Selection

Patrick Zschech · Richard Horn · Daniel Höschele · Christian Janiesch · Kai Heinrich

Received: 17 July 2019 / Accepted: 17 February 2020 / Published online: 18 March 2020

Abstract In any data science and analytics project, the task of mapping a domain-specific problem to an adequate set of data mining methods by experts of the field is a crucial step. However, these experts are not always available and data mining novices may be required to perform the task. While there are several research efforts for automated method selection as a means of support, only a few approaches consider the particularities of problems expressed in the natural and domain-specific language of the novice. The study proposes the design of an intelligent assistance system that takes problem descriptions articulated in natural language as an input and offers advice regarding the most suitable class of data mining methods. Following a design science research approach, the paper (i) outlines the problem setting with an exemplary scenario from industrial practice, (ii) derives design requirements, (iii) develops design principles and proposes design features, (iv) develops and implements the IT artifact using several methods such as embeddings, keyword extractions, topic models, and text classifiers, (v) demonstrates and evaluates the implemented prototype based on different classification pipelines, and (vi) discusses the results' practical and theoretical contributions. The best performing classification pipelines show high accuracies when applied to validation data and are capable of creating a suitable mapping that exceeds the performance of joint novice assessments and simpler means of text mining. The research provides a promising foundation for further enhancements, either as a stand-alone intelligent assistance system or as an add-on to already existing data science and analytics platforms.

Keywords Intelligent user assistance system · Automated method selection · Data science · Natural language processing · Design science research

Accepted after three revisions by the editors of the Special Issue.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s12599-020-00642-3) contains supplementary material, which is available to authorized users.

P. Zschech (✉) · R. Horn · D. Höschele · C. Janiesch · K. Heinrich
Lehrstuhl für Wirtschaftsinformatik, Business Intelligence Research, TU Dresden, 01062 Dresden, Germany
e-mail: patrick.zschech@tu-dresden.de

R. Horn
e-mail: richard.horn@mailbox.tu-dresden.de

D. Höschele
e-mail: daniel.hoeschele@mailbox.tu-dresden.de

C. Janiesch
e-mail: christian.janiesch@tu-dresden.de

K. Heinrich
e-mail: kai.heinrich@tu-dresden.de

1 Introduction

Data science and analytics (DSA) projects are generally multidisciplinary and therefore require combined expertise from several areas, such as profound domain knowledge, analytical modeling skills, and experience in collecting and processing data from heterogeneous IT systems (Mikalef and Krogstie 2019). Consequently, there have been various initiatives to support the implementation of data-driven projects in a stepwise manner, such as the knowledge discovery in databases (KDD) process model (Fayyad et al. 1996) or the cross-industry standard process for data mining (CRISP-DM) methodology (Wirth and Hipp 2000).

Such artifacts give instructions for all relevant tasks from data preparation to analytical modeling and

evaluation with the purpose to provide guidance and structure to the overall DSA implementation process (Kurgan and Musilek 2006). Simultaneously, leading software vendors such as SAS, RapidMiner, Microsoft, IBM, and KNIME offer more and more standardized functionalities within their DSA platforms and applications to make it easier for users with varying background knowledge to use data mining (DM) methods in DSA projects in a systematic and repeatable manner (Serban et al. 2013).

However, despite improved tool support, one crucial step still remains a challenging task throughout the DSA implementation process: The mapping between (i) the *problem space* expressed in the language and the concepts of the domain-specific problem setting, and (ii) the *class of generic DM methods* providing an algorithmic solution for data-driven decision support (Choinski and Chudziak 2009; Eckert and Ehmke 2017). This step requires a translation that determines the character of the subsequent DSA implementation process and, thus, the success of the whole project (Hogl 2003). Usually the translation is carried out by well-trained DSA experts, who bring the necessary skills to merge both contexts, that is the methodical skills needed for a typical data lifecycle as well as the required business understanding to grasp the underlying problem characteristics and achieve the desired outcome towards economic goals (Debortoli et al. 2014; Schumann et al. 2016).

In practice, however, such DSA experts are a rare and costly species and may not be available at all times (Zschech et al. 2018; Huber et al. 2019). Consequently, the mapping task remains to be carried out by multiple stakeholders in terms of DSA experts and domain experts with the latter often being DM novices. Hence, it is an iterative and time-consuming task with unclear prospects and, as Serban et al. (2013) put it, "novice analysts are typically completely overwhelmed".

Against this background, we aim to design and develop an intelligent assistance system (IAS) (Maedche et al. 2016) that is able to support the mapping of the problem space and the class of DM methods for DM novices. It is conceivable that such a system could later be offered as a plugin for commercial DSA platforms.

While there have been several efforts to automatically select DM methods in general, only few approaches take into account the particularities of problem context expressed in natural and domain-specific language, which helps domain experts to maintain their familiar setting without the necessity of acquiring deeper DSA knowledge (Hogl 2003; Eckert and Ehmke 2017). To the best of our knowledge, no commercial off-the-shelf software offers adequate support for natural-language-based DM method selection.

Summarizing, our research question can be formulated as:

> What are the design principles and design features of a text-based IAS for automated DM method selection (TbIAS) targeted at domain experts that are DM novices?

For our research, we follow a Design Science Research (DSR) methodology (Peffers et al. 2007). After introducing relevant foundations and related work in Sect. 2, we outline our DSR approach in more detail in Sect. 3. In Sect. 4, we detail the mapping task and gather design requirements, which we use to formulate design principles and derive appropriate design features. In Sect. 5, we proceed with the design and development steps and conduct several evaluation studies in Sect. 6. In Sect. 7, we discuss theoretical and practical implications as well as current limitations. We summarize the results and offer concluding remarks in Sect. 8.

2 Foundations and Related Work

2.1 Intelligent Assistance for Data Analysis and Automated Method Selection

Due to the technological and methodical DSA achievements in recent years, enabling platforms such as RapidMiner, KNIME and SAS Enterprise Miner are increasingly extending their functionality by providing a large number of DM methods and simplifying the application, inspection, and evaluation of analysis operators and their results. Simultaneously, it is getting increasingly complex, especially for DM novices, to keep track of the field and decide in which context which techniques and operators are the most appropriate. Therefore, a variety of IAS types have been developed, guiding users through all stages of the data analysis process (Serban et al. 2013). Of particular interest is the support of automatically selecting appropriate data analysis methods at diverse levels of abstraction, ranging from specific algorithms up to more superior techniques and method classes. For this purpose, multiple approaches have been proposed to facilitate this kind of task, which can be divided into the three fields of *expert systems*, *meta-learning systems*, and *question answering systems.*

2.1.1 Expert Systems (ES)

Rule-based ES provide the simplest type of support. They consist of a knowledge base derived from hand-crafted expert rules. In our case, the rules should dictate which DM methods are applicable under which circumstances (Serban et al. 2013). Exemplary systems can be found in contributions by Danubianu (2008) and Dabab et al. (2018), where DM methods are determined by a variety of

selection criteria, such as the type and number of data observations, the specific role of data variables, or the presence of missing values. While the latter approach is particularly restricted to classification problems, the former allows for a broader scope by also considering additional method classes such as association rule mining, cluster analysis, and deviation detection. However, both approaches remain at a generic level as they only rely on dataset features and method characteristics. Moreover, rule-based ES have the disadvantage that they cannot cope with natural-language input to express particularities of domain-specific problem settings.

2.1.2 *Meta-learning Systems (MLS)*

While the previous group of systems is built manually on hand-crafted rules, another approach is to derive such rules automatically by discovering the relationship between measurable characteristics about a problem at hand and the performance of different algorithms or techniques. Previous work in this area can be broadly summarized under the terms *meta-learning* (Lemke et al. 2015) and *per-instance algorithm selection* (Kerschke et al. 2019). However, the focus of existing contributions in these areas is limited to a rather narrow scope, since they are primarily concerned with the selection of the best performing algorithm for a clearly defined DM task, such as specific classifiers for a binary classification, based on given dataset meta-characteristics (Serban et al. 2013). Vainshtein et al. (2018) developed an MLS which does not only consider meta-characteristics from the data itself but also incorporates textual features from technical problem descriptions. These features were built on word-embeddings derived from academic publications.

2.1.3 *Question Answering Systems (QAS)*

The purpose of QAS is to enable natural-language queries on large-scaled databases containing structured and unstructured information (Athenikos and Han 2010). Thus, QAS can be described as a form of IAS for information retrieval that produce a specific answer rather than a list of items based on an unstructured, textual user input (Wang et al. 2018). The development of QAS is a complex endeavor as they deal with multiple tasks: (i) question classification, (ii) answer extraction, and (iii) answer selection (Guda et al. 2011). Since there is usually a variety of different question types, the intent of a user is classified and respective annotations are added to the query in the step question classification. Then the QAS calculates matches between the knowledge base and the annotated query and generates a candidate list. Finally, an answer from the candidate list is selected. This implies that the QAS problem can be formulated with multiple types of questions and answers with a wide variety of relevant entities.

Gupta and Gupta (2012) distinguish between two fundamental types: (i) systems that are based on simple *information retrieval* and *natural language processing* (NLP) methods (sQAS) and (ii) systems that depend upon the reasoning with natural language (rQAS). While sQAS are domain independent and use basic methods, such as syntax processing and named entity tagging, rQAS are specialized towards a specific domain and require a profound knowledge base to use methods such as semantic analysis and high reasoning (Gupta and Gupta 2012). Hence, answers from rQAS are usually more sophisticated and synthesized in comparison to text snippets of sQAS which were merely ranked. For example, Hogl (2003) designed a knowledge-based system in which domain experts can communicate with the system via a QAS interface using natural language and directly obtain usable DM analysis results inferred from connected databases. The actual selection of DM methods runs "invisibly" in the background and depends on the content and the type of the questions. However, as the underlying system was designed as an rQAS, it requires the modeling of a sophisticated and complex knowledge base that includes predefined QA elements as well as a strong formalization of methodical and conceptual knowledge.

In summary, there are several approaches dealing with the automated selection of methods in the DSA context from slightly different perspectives. However, for the problem at hand, they are only applicable to a limited extent, as they either (i) focus on a too narrow scope, (ii) require high modeling efforts for a sophisticated knowledge base, (iii) require prior knowledge of questions asked, or (iv) cannot process natural-language inputs containing domain-specific expressions. Nevertheless, some promising directions can be identified from previous work considering methods that belong to the field of text mining (TM) and particularly text classification.

2.2 Text Mining and Text Classification

TM is concerned with the inference of meaningful information from texts that are stored in structured databases, semi-structured documents (e.g., XML or JSON), or unstructured plain text documents using automated procedures involving different algorithms (Aggarwal and Zhai 2012). Because of the inherent syntactic and semantic features of text data, TM is at the center of an intersection of multiple disciplines that often overlap, such as information retrieval, NLP, and machine learning (ML) (Allahyari et al. 2017). The automated extraction of knowledge from texts spans several tasks and methods that

can be broadly summarized by the following categories: *text representation*, *text preprocessing, text summarization*, *text clustering*, *text classification and regression*, and *information extraction* (Hotho et al. 2005; Aggarwal and Zhai 2012; Allahyari et al. 2017).

2.2.1 Text Representation

To automatically process text data, a data structure is required that is more suitable than a plain text file (Hotho et al. 2005). Hence, raw text is usually transformed into a matrix representation, such as the vector space model proposed by Salton et al. (1975). Considering the complexity of text representations, basically we can distinguish between (i) simple representations, such as one-hot encodings and bag-of-words, and (ii) more advanced approaches, often referred to as *distributed representations* or *embeddings* primarily based on neural networks (Mikolov et al. 2013). Due to the sparsity of simple vector models, distributed representations are considered state-of-the-art in modern NLP applications, as they are able to represent text in dense vectors. Furthermore, these approaches allow the inclusion of semantics by building vector models based on the words' context and can be extended to create so-called paragraph embeddings that include semantics of whole sentences or documents (Perone et al. 2018). The area is currently subject to rapid developments, where continuous innovations such as FastText (Bojanowski et al. 2017), deep averaging networks (DAN) (Iyyer et al. 2015), or Google's universal sentence encoder (USE) (Cer et al. 2018) allow the creation of advanced embedding models with increasingly better capabilities to process given context information.

2.2.2 Text Preprocessing

The preprocessing of text is a key component in many TM applications to prepare natural-language data for subsequent modeling tasks. Frequently used methods include for example word- and sentence tokenization, stop word removal, part-of-speech (POS) tagging, parsing, lemmatization, and stemming (Jurafsky and Martin 2008).

2.2.3 Text Summarization

The goal of text summarization is to reduce the length and detail of a text to provide a concise overview of a large document concerning a topic. Two important approaches in this area are *keyword extractions* and *topic modeling*. The first group includes algorithms such as YAKE! (Campos et al. 2018) or TextRank (Mihalcea and Tarau 2004) that are used to automatically identify key terms, phrases, or segments that best describe the subject of a document. Topic modeling, on the other hand, aims at discovering latent semantic structures from a collection of documents in the form of abstract topics by finding related groups of words that best represent the information in the collection. For this purpose, probabilistic models like latent Dirichlet allocation (LDA) (Blei et al. 2003) are applied to extract (i) probability distributions over words that define a topic and (ii) probability distributions over topics that define a document.

2.2.4 Text Clustering

Text clustering is concerned with finding groups of similar documents in a collection of documents. For this purpose, different types of clustering algorithms can be applied, such as *hierarchical algorithms* (e.g., agglomerative vs. divisive clustering), *partitioning algorithms* (e.g., k-means), or *probabilistic algorithms*. The latter type is closely related to the field of topic modeling as already introduced for the task of text summarization (Allahyari et al. 2017).

2.2.5 Text Classification and Regression

The general purpose of text classification is to automatically assign text documents to one or more predefined categories. Frequently used examples are the support vector machine (SVM), naïve Bayes, *k*-nearest neighbor (KNN), or boosting trees (Bishop 2006; Kowsari et al. 2019). Furthermore, recent studies increasingly focus on neural networks that consist of multiple, hierarchically organized processing layers, which is often referred to as deep learning (DL) (LeCun et al. 2015). Their multi-layered architecture allows them to be fed with complex inputs and then automatically discover internal representations at different levels of abstraction that are needed for classification tasks. There are multiple architectures available such as deep feedforward networks, convolutional networks, or recurrent networks (Goodfellow et al. 2016). However, focusing on text data, recurrent architectures such as long short-term memory (LSTM) networks (Hochreiter and Schmidhuber 1997) or gated recurrent units (GRU) (Cho et al. 2014) are most often the preferred choice due of their ability to better handle sequential data. Text regression models are similar to classification models but instead of predicting a nominal class variable they predict outputs on a numerical scale.

2.2.6 Information Extraction

Information extraction aims at automatically extracting structured information from texts. It comprises two fundamental sub-tasks: *named-entity recognition* and *relation extraction*. While the former focuses on the localization of

named entities and their classification into predefined categories, such as organizations, character names, or locations, the latter aims at seeking and locating semantic relations between entities (Allahyari et al. 2017).

Despite the successful use of TM applications in different classification tasks such as email filtering, content-based item recommendation, or document categorization (Aggarwal and Zhai 2012; Kowsari et al. 2019), there has been no application so far that specifically addresses the task of mapping domain-specific problem descriptions to DM methods. Nevertheless, several of the employed techniques represent promising approaches to tackle the problem method selection.

3 Research Approach

DSR is a fundamental paradigm in information systems research concerned with the construction of socio-technical artifacts to solve organizational problems and derive prescriptive design knowledge (Gregor and Hevner 2013). Specifically, we follow the DSR methodology proposed by Peffers et al. (2007) consisting of the six steps of (i) problem identification and motivation, (ii) definition of the objectives for a solution, (iii) design and development, (iv) demonstration, (v) evaluation, and (vi) communication. The adoption of the methodology to our project is depicted in Fig. 1.

We start with a detailed description of the mapping problem in DSA projects. We use an illustrative example scenario based on experiences obtained from industrial practice and relate our observations to prior requirements from literature (specifically Meth et al. 2015) (1). Subsequently, we translate the problem into design requirements and formulate design principles and features for a TbIAS (2). In doing so, we conform to Mode 3B of design theorizing as introduced by Drechsler and Hevner (2018) and seek to inform solution entity design by prior knowledge for entity realization to codify effective facets of the resulting artifact (Mode 4B).

In the next step (3), we develop the instantiated artifact. For the identification of suitable TM and classification methods, we carried out a literature review (Webster and Watson 2002) using the databases *ArXiv, IEEE Explore* and *ScienceDirect* as well as *Google Scholar*. Section 2 has already summarized the finding of our analysis. A broader investigation of those methods has been carried out in previous work (Zschech et al. 2019), where we discuss their suitability for certain processing and recommendation tasks. We build on these findings and concentrate on the TbIAS's design and the creation of a suitable learning base. Subsequently, we establish the system design based on multiple processing pipelines and demonstrate the prototype (4), contributing an effective solution entity (cf. Mode 6A of Drechsler and Hevner 2018).

The evaluation of the system design artifact is carried out in a comprehensive evaluation study based on real-world problem descriptions (5). The evaluation involves the examination of different design configurations and the judgement capacity of student subjects as representative DM novices. Finally, we communicate the results and their contribution to theory and practice as well as outline current limitations as opportunities for future work (6).

4 Design Requirements of DSA Projects and Design Principles for a System Design Artifact

4.1 Description of the Mapping Problem

In the following, we outline a typical initiation of a DSA project in such a way as is common in industrial practice to specify the particularities of the DSA mapping problem. A DSA project typically starts with a concrete business case in which data-driven decision support is sought (e.g., configuration management of equipment).

The experts of the field (e.g., technical engineers) with their respective domain understanding describe the problem of interest (e.g., inefficient machine operations) in their own language and provide the necessary context. Then, the DSA experts (e.g., either internal specialists or external

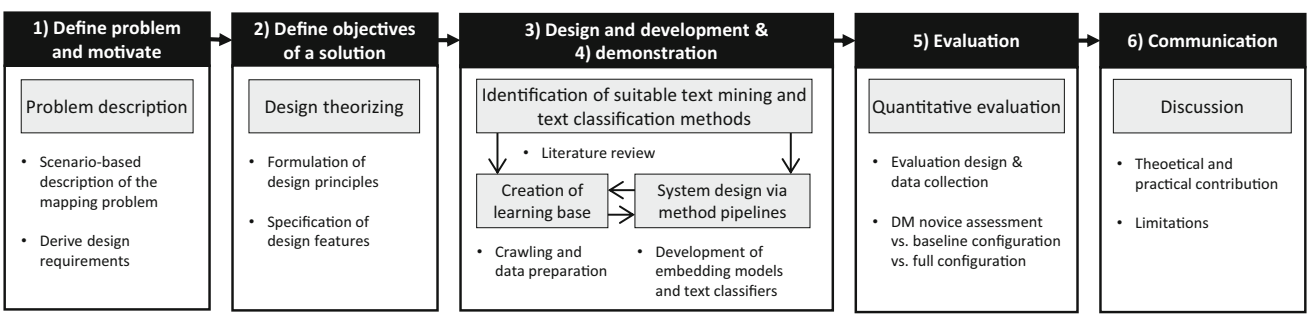

Fig. 1 Research approach

advisors) are consulted to abstract the specific problem instance in order to realize a mapping with a certain class of DM methods (e.g., regression, cluster analysis, association rule mining) which could possibly solve the problem. To realize the mapping, DSA experts focus on characteristic keywords and key phrases (e.g., forecast, similar groups, anomalies, frequent combinations) that could signal the methodical nature of the described problem setting. Furthermore, they recognize all relevant domain entities of interest and ignore any additional noise from the problem description that is not relevant for the mapping task.

As a result, the DSA experts communicate which class of DM methods might be suitable, what kind of input data is required, how it is processed, and what type of output can be expected. This helps the domain expert to obtain an entry point to the methodical field and derive a better understanding of a possible solution space. Thus, the DSA project in its initial stage can be described in terms of domain entities of interest, the DM method to be implemented, and the data assets to be used to create a blueprint for project realization (Brodsky et al. 2015; Zschech 2018; Hesenius et al. 2019).

As argued above, this scenario usually requires the presence of skilled DSA experts. To assist this type of mapping problem in an automated manner, the intended design artifact should provide the advice functionality to the domain expert instead of the DSA expert.

4.2 Extraction of Design Requirements

The above scenario and its inherent requirements allow us to derive meta-requirements for the design of a TbIAS. Our design requirements further derive from exchanges with practitioners and our own experience from previous research (Zschech et al. 2019). The presentation is structured along the projection of prior design requirements. In addition, Appendix A (available online via http://link.springer.com) contains more details of the practical rationale behind our design requirements.

Baskerville and Pries-Heje (2019) argue that projectability is a "forward-looking means to […] propagate design knowledge". While they focus on the projectability of design theories, we argue that the underlying design requirements already constitute design knowledge and can be projected independent of the artifact to better structure the requirements engineering (Rupp 2014) of other DSR projects.

We project the design requirements for a decision support system (DSS) from Meth et al. (2015) and use their prior knowledge for entity realization in terms the of human decision makers' goal and the resulting generic DSS design requirements (DSSDR). Meth et al. (2015) state that "the perceived advice quality, perceived cognitive effort, and perceived restrictiveness are important features of any DSS". They argue that any DSS should (i) increase the decision quality by providing advice with high advice quality, (ii) reduce the human decision maker's cognitive effort by providing decision support, as well as (iii) minimize system restrictiveness by allowing users to control strategy selection. We use their three DSSDR to inform our design requirements and ensure that they represent a balanced and holistic perspective on our solution artifact.

In the following, we present the design requirements as the foundation and impetus of our subsequent design. They guide us when developing a TbIAS as the main artifact of our research. Cf. Figure 2 for an overview of the relations between DSSDR, design requirements (DR), design principles (DP), and design features (DF).

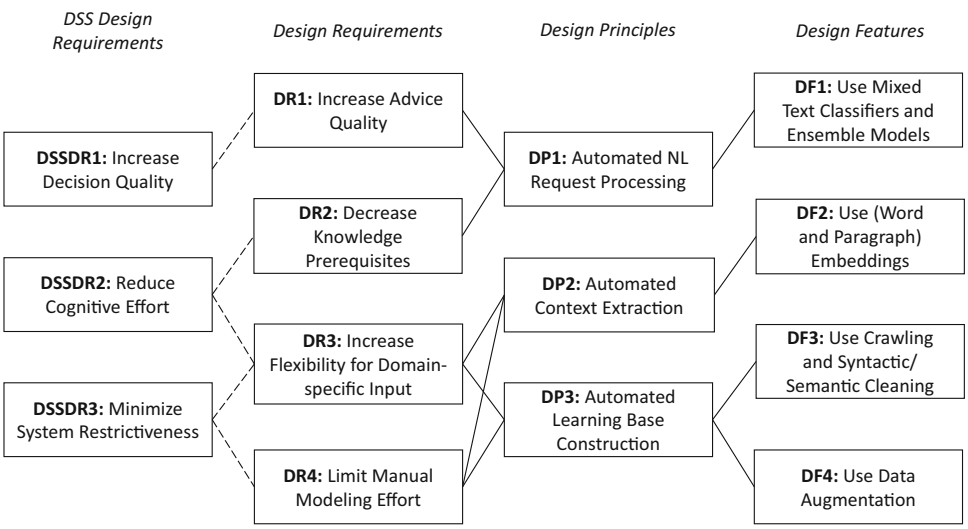

Fig. 2 Relations between DSSDR, design requirements, design principles, and design features

DR1 Increase advice quality for novice users.

As argued above, the limited availability and high cost of expert advice limits DSA project definitions and can result in iterative and time-consuming endeavors of DSA novices with unclear decision quality and, thus, chances of success. Hence, we argue that it is useful to conceive an (automated) means of assistance for domain experts who are DM novices to select a suitable class of DM methods for their business tasks independent of DSA experts. Naturally, the advice should be of improved quality over pure guessing, guesswork exchange with other DM novices, and other baseline configurations.

DR2 Decrease knowledge prerequisites for novice users.

Domain experts are not DSA experts. It is infeasible to assume that apart from their domain expertise, domain experts can bring along DM modeling and method knowledge to the table and perform the complex task of DM method selection successfully on their own. While they may be able to gain a certain degree of understanding of DM methods over time, they will remain DSA laymen. A lack of DM method knowledge also implies a lack of knowledge about keywords and definitions. Consequently, the advice should be available when asked for in the natural language of the domain expert to decrease knowledge prerequisites when interacting with the TbIAS.

DR3 Increase the flexibility for domain-specific input for novice users.

Domain experts require an assistance of DM methods in a plethora of use cases. Hence, the TbIAS has to ensure that assistance can be rendered regardless of the domain, functional area, or industry. That is, the domain expert must not be forced to translate his or her domain-specific request into a domain-independent query in order for the TbIAS to be able to process it. On the one hand, this contributes to reducing cognitive efforts of the domain experts, on the other hand it minimizes the restrictiveness of the user group for the system.

DR4 Limit manual modeling effort to construct the learning base.

The last requirement limits the scope of the design artifact in particular terms of economic efficiency. As with many ML-related approaches, the advice quality of the TbIAS correlates with the quality of the learning base as the basis for its assistance. There are many approaches to create these databases which range from automated construction to purely manual modeling as it is common for QAS. Yet only a fraction of them can be implemented in an economically feasible way and ensure a sufficiently large learning base that exhibits adequate degrees of freedom for our application of DM method selection. The requirement is further constrained by the limited public availability of labelled problem statements (Zschech et al. 2019). Summarizing, as the cost of human labor does not efficiently scale for learning base construction, it is necessary to limit the manual modeling effort as far as possible.

4.3 Formulation of Design Principles and Features

In the previous section, we formulated the requirements towards TbIAS in the form of design requirements. Our search process was an iterative process of building artifacts, demonstration and learning, and improvement. In doing so, we were able to formulate design principles, which describe a class of systems to assist novice user for DM method selection. During our conceptual phase, we formulated initial design principles and refined them in an iterative process of discussion and reflection with business professionals and researchers. Our research built on and benefited from the continuous exchange with industry. Our design principles provide general design considerations as a useful standard of conduct.

We formulated our design principles according to Chandra et al. (2015)'s proposal for effective formulation, including materiality, action, and boundary conditions. We do not claim that our design principles provide a replacement, but rather an enrichment of current practices. The collective boundary condition for all design principles is "for intelligent assistance in DM method selection for DM novices". Again, see Fig. 2 for an overview of the relations of our constructs to each other. See Appendix B for more details regarding our design rationale.

DP1 Provide the system with the functionality to process natural-language user requests automatically and in their entirety in order for the system to assist novice users in DM method selection.

As argued above, due to the cost and unavailability of DSA experts in DSA-related projects, other (automated) kinds of intelligence assistance must be rendered to domain experts who are DM novices to improve the quality of DM method selection. We suggest a system design artifact (Offermann et al. 2010) that is able to translate user problem statements into advice in the form of DM method class suggestions. To decrease knowledge prerequisites, the artifact must allow domain experts to input the problem statement of their DSA project in natural language. Hence, the system should be able to process not only complete sentences but a paragraph in its entirety, which captures the essence of the DM method selection problem space.

DP2 Provide the system with the functionality to extract the embedded context from the system's learning base automatically in order for the system to recognize and discriminate user requests regardless of their locale.

Natural language exhibits a high degree of complexity and richness in terms of grammatical structures, ambiguous word meanings, and additional context supplements. Any system design artifact must be able to filter out irrelevant noise and instead extract central constructs such as keywords and phrases that signal a match or at least the similarity between domain-specific problem descriptions and generic DM method descriptions. Consequently, the artifact must be able to abstract from the locale of the user, that is it must be able to understand the user request independent of the domain, industry, or functional area it is embedded in and provide advice that is useful for the problem space only. It must do so in an automated manner that does not require extensive manual modeling effort.

DP3 Provide the system with the functionality to construct the learning base automatically in order for the system to be economically feasible and exhibit adequate degrees of freedom.

Labelled problem descriptions from industrial practice are only sparsely available, as companies usually do not store such information in central and public repositories. Further, manual labelling incurs costs and does not scale. Thus, to ensure the practicability of our system design artifact, we suggest to follow the example of Vainshtein et al. (2018) and propose to build the corpus on texts from academic articles. This enables the automatic construction of a sufficiently large training base, which exhibits adequate degrees of freedom when processing domain-specific user requests.

In the next step, we specified design features as concrete instantiated capabilities towards the TbIAS's concrete implementation that satisfy the prescriptions of our design principles. DP1 and DP2 are addressed by a single design feature each while we propose to realize DP3 with two distinct yet complementary design features. The embodiment of the design features rests upon the previous literature review as summarized in Sect. 2. In the following, we provide only a brief overview of the design features. In Sect. 5, we present the complete system design artifact in more detail.

DF1 Use mixed text classifiers and ensemble models to automate natural-language request processing.

For the realization of DP1, we suggest to employ text classifiers from the field of ML. Such algorithms can support matching problems by extracting regularities between a target variable with predefined categories and high-dimensional input data, which is typically the case with natural language data (Aggarwal and Zhai 2012). The advantage is that the model building happens automatically by iteratively learning from labelled observations, which allows the TbIAS to detect complex patterns and relationships without being explicitly programmed (Bishop 2006). As generally a broad range of text classifiers is available with different learning capabilities (cf. Sect. 2.2), we implement a mixture of classifiers and additionally combine the best performing ones within an ensemble model to harmonize their individual strengths and weaknesses (Sagi and Rokach 2018).

DF2 Use (word and paragraph) embeddings to enable automated context extraction.

To enable the automated extraction of context, we suggest using different types of embedding models (cf. Sect. 2.2). Such models make it possible to represent text elements of various abstraction levels (i.e., words, sentences, paragraphs, etc.) in dense vectors to capture the semantic meaning between related terms and phrases (Mikolov et al. 2013). In this way, the semantics of different DM methods can be projected into the vector space, so that they can be used as automatically derived features for the text classifiers to more accurately solve the given matching problem. This is particularly relevant because of the lack of an extensive database with labelled problem descriptions.

DF3 Use crawling and syntactic/semantic cleaning techniques to construct the learning base automatically.

As described above, we build the TbIAS on a text corpus derived from academic articles to automatically construct a sufficiently large learning base. Similar to Vainshtein et al. (2018), we propose a database crawling approach, as it provides a solution to obtain a large number of articles with minimal effort. Moreover, we suggest applying several automated cleaning techniques, since crawled documents usually exhibit a high degree of syntactic noise and semantic outliers.

DF4 Use data augmentation techniques to construct the learning base automatically.

Despite the crawled text corpus, one cannot necessarily be sure that the resulting learning base possesses a sufficient variety of terms and expressions which may be required to realize an adequate mapping with domain-specific expressions embedded in problem descriptions. For this reason, we suggest implementing the feature of automated data augmentation in order to artificially increase the degree of term and phrase variability.

4.4 Outline of the System Design Artifact

In the following, we present the archetypical process of rendering intelligent assistance through a TbIAS. Figure 3 provides a summary of the selection process.

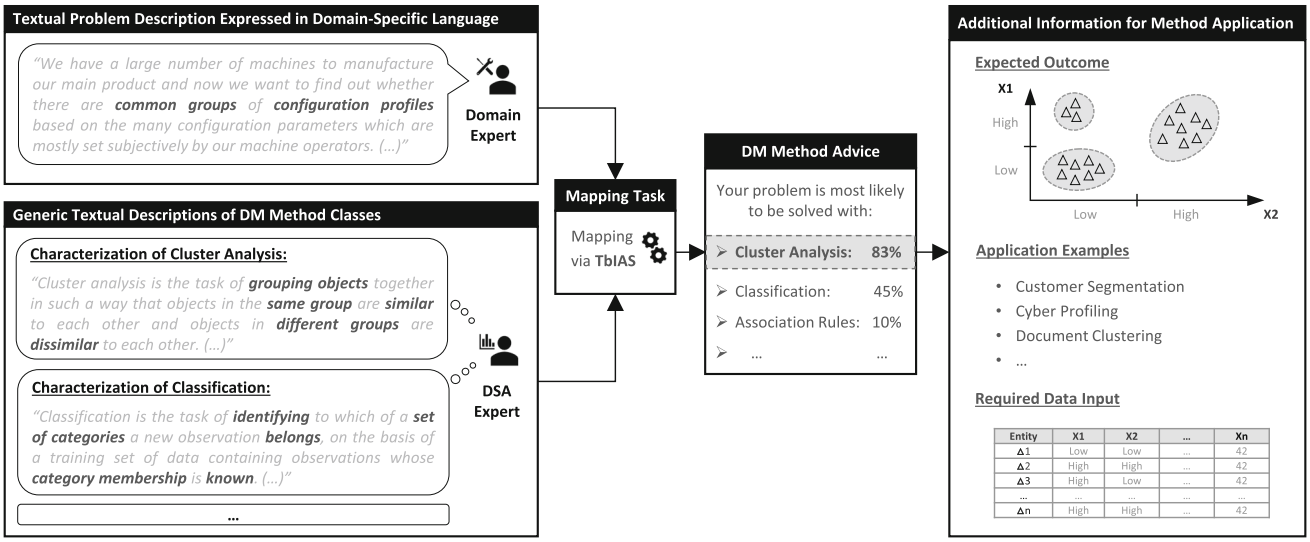

Fig. 3 Intended functionality of a TbIAS for DM method selection

The TbIAS receives a textual and domain-specific problem description from the DM novice and recognizes all relevant entities and relationships of interest. The specific domain problem is then translated into a more abstract problem class, which can be mapped to a certain class of DM methods. The mapping itself is based on a learning base consisting of structured and unstructured information of generic DM method descriptions, from which the TbIAS can infer which class of DM methods addresses the articulated problem. This results in advice for the user by determining the DM method class with the highest degree of suitability. On this basis, the DM novice receives further information for the application of the DM method as an entry point to the methodical field.

5 Design and Development of the System Design Artifact

5.1 Construction of a Learning Base

In the following, we describe the construction of the learning base as the foundation for our TbIAS according to the design principle DP3. For this purpose, we first define a sub-selection of DM methods as target classes and subsequently outline the data creation process of several relevant subsets for training purposes.

The main objective of the TbIAS is to recommend those DM methods out of a number of alternatives where the degree of equivalence with the given problem description is the highest. Generally, there is a broad set of DM methods available, such as classification, regression, cluster analysis, factor analysis, association rule mining, sequence mining, graph mining (Hogl 2003; Manyika et al. 2011). For this purpose, we currently focus on a predominantly employed subset of DM methods to keep the complexity manageable. In particular, we concentrate on the three classes of (i) *clustering* (*CL*), (ii) *prediction* (*PR*) (comprising classification and regression), and (iii) *frequent pattern mining* (*FPM*) (comprising association rule mining and sequence mining).

For the creation of our learning base, we used excerpts from academic articles in which the application and characteristic properties of the selected DM methods are described representatively (Vainshtein et al. 2018). This entails that there are two different data pools with syntactic divergence (Kulkarni et al. 2013) mainly expressed in the pursued intentions and applied terminologies (cf. Table 1 for their distinction).

Table 1 Divergence between academic articles and problem descriptions

	Academic articles	Problem descriptions
Usage	Training and test data	Validation data
Intention	Method definitions and technical problem solutions	Ambiguous problem descriptions from real-world scenarios
Terminology	Mainly method-centric, partly domain-specific	Highly domain-specific

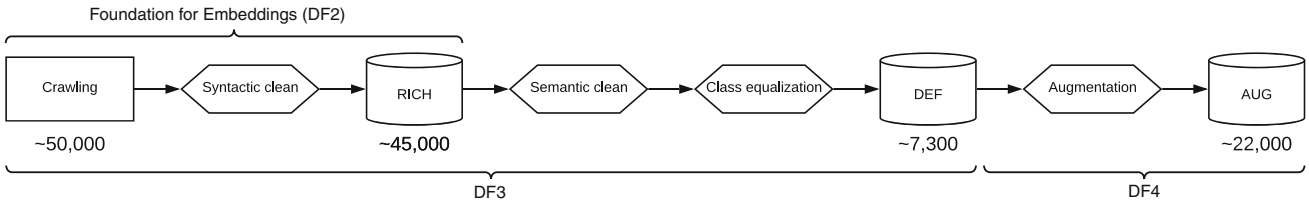

Fig. 4 Process of dataset generation for different training purposes

We preprocessed this data pool of academic articles in several preparation steps, resulting in three different subsets that were required for the subsequent training purposes according to the design features DF2–DF4. Figure 4 illustrates the overall creation process. Cf. Appendix C for exemplary sentences for each preparation step.

As a starting point, we crawled the scientific databases *Elsevier* and *arXiv* as well as the online journal *Medium* as they provide standardized crawling APIs. For the definition of search terms, we applied several synonyms of the three target classes within the fields *title*, *abstract* and *keywords* to obtain a large body of academic documents that are related to the application of the predefined DM methods. This resulted in about 50,000 documents as a basis for further processing. The syntactic clean then aimed to reduce noise and remove syntactically irrelevant elements like formulas, authors, numbers, or brackets as well as documents that did not reach a certain length. As a result, we obtained about 45,000 clean and context-rich documents as our RICH dataset, which were necessary to train our embedding models at a later stage. The RICH dataset serves as the basis for the TbIAS's design feature DF2.

In the next step, the richness of context was deliberately reduced to concentrate on phrases that show a definition-like character and are able to express the constituent properties of the individual DM methods. The result is our DEF dataset. For this purpose, the step of semantic cleaning aimed to find such definition-like phrases in the full dataset by applying two processing steps in which the advantages of vector representation were exploited.

First, the vectors were inferred by applying the pre-trained USE model as shown by Cer et al. (2018). Based on the distributional hypothesis, documents that internalize the meaning of a target class are closer to each other in vector space than those that represent the meaning of the target class only slightly. To make use of this assumption, we applied a noise clustering approach (Dave 1991) per target class using the DBSCAN algorithm (Ester et al. 1996). In this way, it was possible to assign semantic outliers to a single noise cluster to remove them from the dataset, as illustrated with the red dots in Fig. 5. Please note that, for visualization purposes, the vector representations were reduced to a two-dimensional space using principal component analysis. Second, to obtain phrases that are close to definitions, we preselected DM method definitions from literature and applied cosine similarities between all phrases in our data. The top 20% of the most similar sentences were selected and then used in the next steps.

Up to this point, the dataset contained an unequal distribution of documents among the three target classes with a strong bias towards FPM (PR: 2443, CL: 2862, FPM: 6127). To reduce bias, we equalized the class distribution according to the least represented class, resulting in about 7300 documents stored in the DEF dataset. This entire automated processing pipeline realizes the design feature DF3.

In order to assure a suitable level of variability within the text corpus, we implemented design feature DF4 with the purpose of expanding the learning base by means of data augmentation methods. Particularly, we applied synonym substitution based on the lexical online database WordNet[1] and a trigram Markov model to generate new sentences out of the existing DEF dataset. This resulted in about 22,000 documents stored in the augmented AUG dataset.

5.2 Development of Embedding Models

For the instantiation of design principle DP2, we implemented design feature DF2 by training multiple variants of embedding models. For this purpose, the RICH dataset was used due to the higher contextual information of the unfiltered documents. Overall, we chose three different embedding architectures for a comparison. The first was based on FastText as it is robust to small datasets and minimizes the problem of inferring vectors for words that were not part of or rarely represented in the training vocabulary using sub-word information in terms of character *n*-grams (Bojanowski et al. 2017). As a second approach, we used DANs as they apply an unordered composition function for paragraph vectors to counteract the syntactic divergence outlined above because of their ability to separate syntactically similar sentences with different meaning (Iyyer et al. 2015). Finally, we considered a third approach from the field of transfer learning by using Google's USE as a pre-trained embedding model

[1] https://wordnet.princeton.edu/.

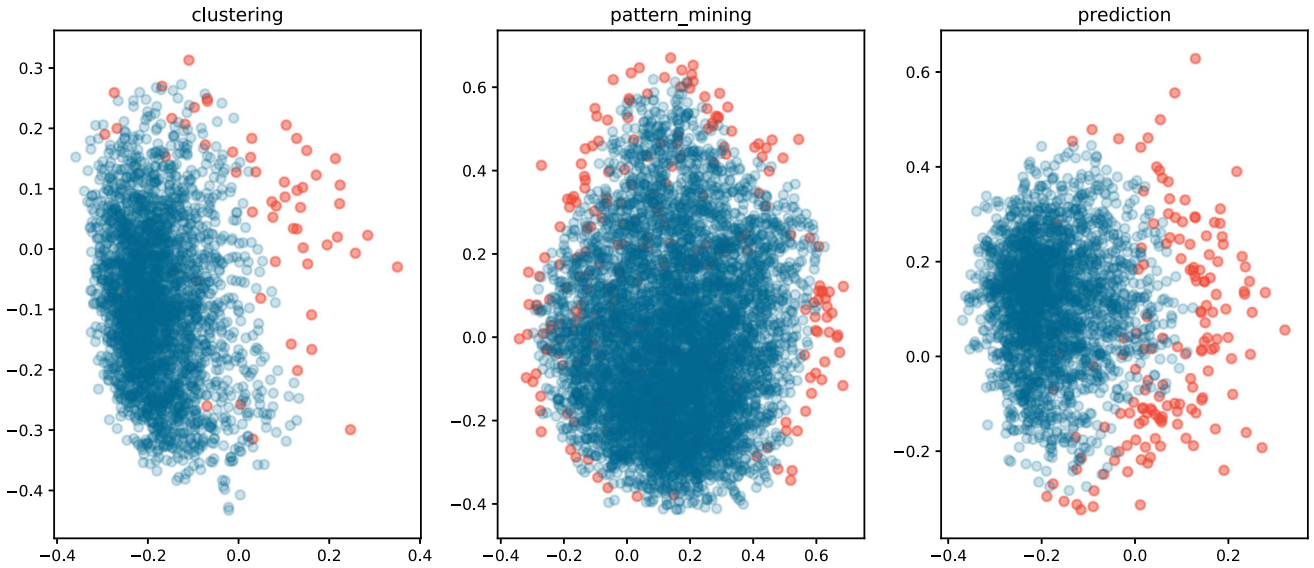

Fig. 5 Noise clustering for removal of semantic outliers

(Cer et al. 2018), which was updated with the RICH dataset to better incorporate the given context.

To determine the best performing architectures for both DAN and FastText models, we investigated multiple combinations of hyperparameter settings using a grid search approach. By doing so, we trained a total of 36 FastText and 480 DAN models to select the best candidates for further processing. As a preliminary result, we observed that the best performing DAN models outperform the FastText embeddings. Furthermore, we confirmed Iyyer et al. (2015) concerning the response to syntactic divergence by DAN models, since each additional layer of the network architecture contributes to an increasing discrimination among the three target classes as shown in Fig. 6. In contrast, however, a too high number of layers causes blending effects at the class boundaries, which underlines the importance of hyperparameter tuning during the training of embeddings. For further information on the training and evaluation of the embedding models, please refer to Appendix D.

5.3 Development of Text Classifiers

Finally, to realize design principle DP1, we developed a mixture of several text classification models based on alternative learning capabilities as outlined in Sect. 2.2. For this purpose, we applied the previously trained embedding models to receive vectorized representations of the two datasets DEF and AUG and used them separately to build the different text classifiers. Figure 7 provides an overview about the individual pipelines for classification model training that were performed to examine the best possible assignment of a target class.

In total, we trained nine types of classification models, whereas six of them directly used the vectorized representations derived from the embedding models as input features. Here, we specifically focused on SVM and KNN as rather simple classifiers and four recurrent neural networks as more sophisticated DL approaches to capture the sequential nature of natural language. The latter included the two networks LSTM and GRU, each of them trained with two different architecture variants, that is a one-to-one (1–1) and a many-to-one (N–1) architecture, to compare the different effects of word versus paragraph embeddings. Each of those classifiers was trained with different hyperparameter settings, resulting in 14,876 model variants among all classifiers, whereas for subsequent evaluation purposes, we only included the best performing models as assessed during the training stage.

The seventh type of classifier was built on a KNN model in combination with keyword extractions (Topic-KNN) as input features to retrieve a collection of the most relevant words that describe a respective target class. The final two classifiers were based on topic modeling using LDA, whereas a first variant uses three topics for directly classifying problem descriptions and the second variant is based on seven retrieved topics in combination with a subsequent SVM classifier. For more information on the Topic-KNN model and the examination of keyword extractions as well as both LDA models and the extracted topics, please refer to Appendix E.

Finally, we evaluated the different classifiers on an external validation dataset with realistic characteristics of real-world problem descriptions (cf. Appendix G) and combined them within an ensemble model (Sagi and

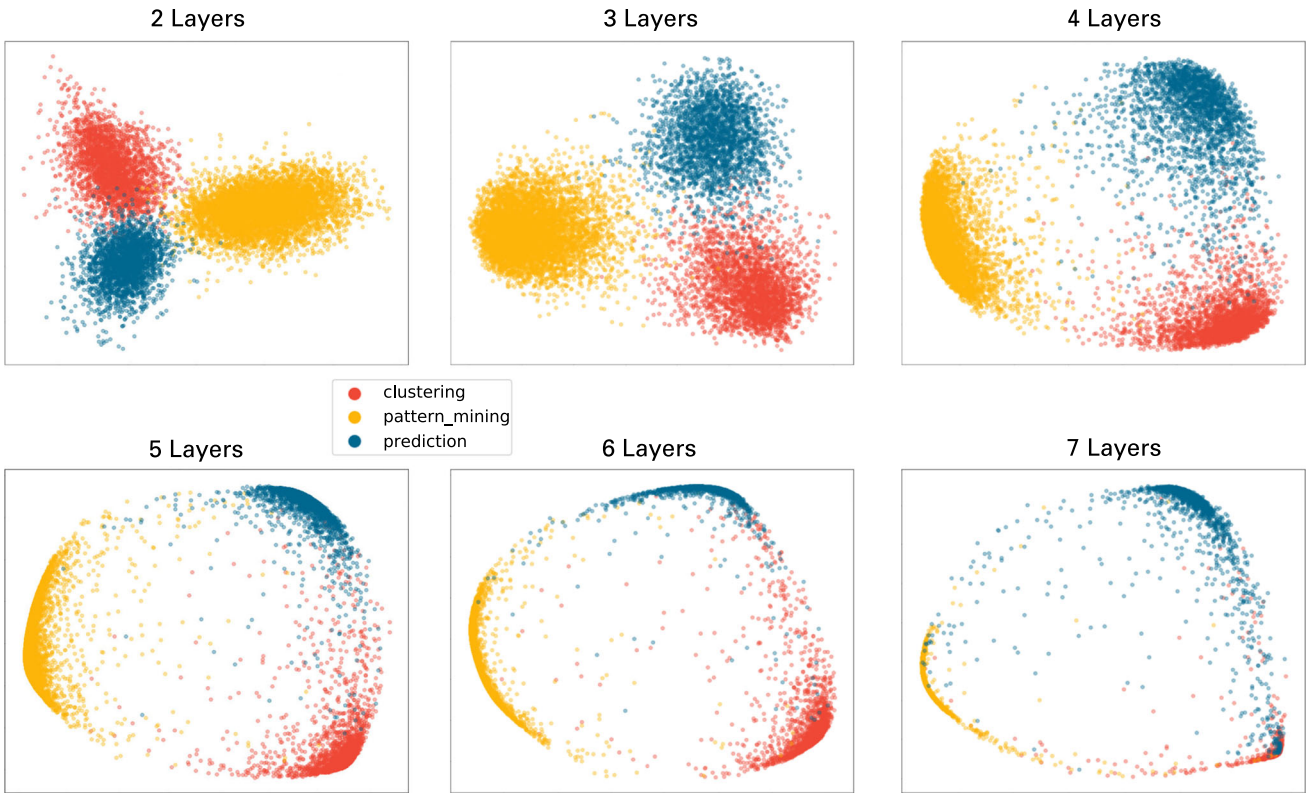

Fig. 6 Counteracting syntactic divergence with deep averaging networks

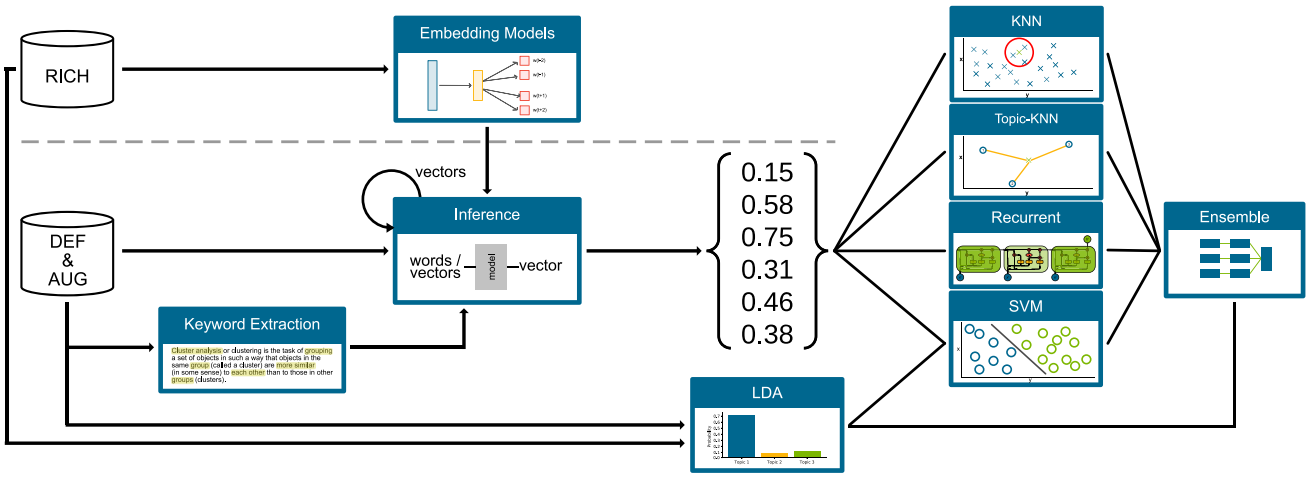

Fig. 7 Overview of pipelines for classification model training

Rokach 2018), where the weights were determined by the performance of the individual models.

5.4 Prototypical Implementation

In the following, we demonstrate the prototype's frontend and backend components and their technical realization. The frontend consists of two consecutive output pages. The first page serves as the landing page, where the user can enter a title and a description to express the problem statement (Fig. 8a). This information is then used as a concatenated text by the prototype to process the output. The results are displayed on the result page (Fig. 8b) visualizing the calculated assignment scores as well as an exemplary application of the recommended DM method class so that the novice user can gain a quick overview of the basic functionality.

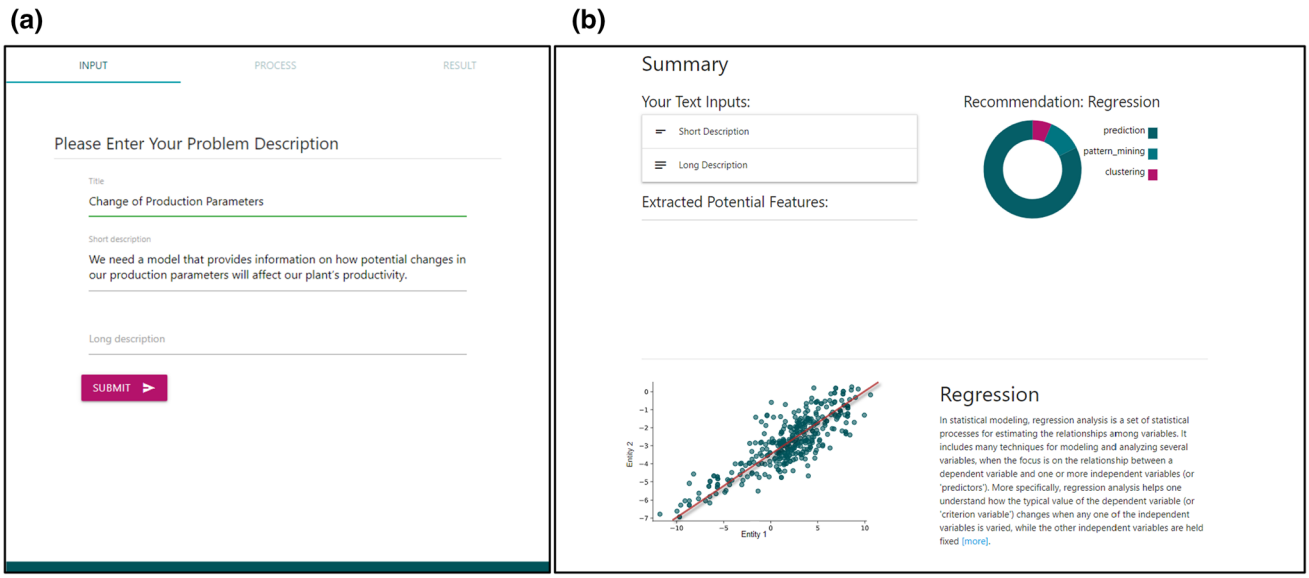

Fig. 8 Landing page (**a**) and result page (**b**) of the prototype

The entered information is processed in the backend by the analytical architecture of the prototype. This architecture is based on those feature implementations that performed best on external validation data during the quantitative evaluation studies (see subsequent evaluation section). The operating principle behind the composition of the best performing components works as follows: The input information is first transformed into both (i) word vectors using the FastText embedding model, and (ii) paragraph vectors using DAN vectorizations. Subsequently, the word vectors are used in combination with a many-to-one LSTM, whereas the paragraph embeddings are forwarded to a one-to-one GRU and an SVM. Finally, these predictions are combined by the ensemble approach for the final classification results.

We made the prototype available on GitHub[2] to guarantee transparency and provide an entry point for the community to participate in further research as well as testing and development activities. See Appendix F for more details.

6 Evaluation

6.1 Evaluation Design

For the evaluation, we collected out-of-sample data that is not linked to the constructed learning base and therefore was not part of the model training procedures. It served both as a basis for benchmarking purposes between alternative text classifiers and for the comparison of the different evaluation items. In particular, we collected 60 different real-world problem statements, which are equally distributed among the three target classes, based on problem descriptions derived from own industrial DSA projects as well as selected DM competitions from online platforms such as Kaggle.[3] When gathering the set of problem statements, we paid attention to ensure (i) that the underlying scenarios originated from a wide range of application domains, (ii) that the keywords and key phrases for signalizing a specific class of DM method contained sufficient degree of variability, and (iii) that the descriptions were provided with a varying degree of filling information and noise. The complete list of problem descriptions can be found in Appendix G.

To evaluate our system design artifact, we measured its advice quality to provide a correct mapping between problem statements expressed in domain-specific natural language and DM methods. We grounded the evaluation on a performance comparison of different evaluation items constituting test and reference elements for multiple design hypotheses. Table 2 provides an overview of these items.

According to the derived design requirements, the TbIAS should be able to provide advice that is of improved quality over random guessing assuming a discrete uniform distribution across all possible DM methods, which determines the lowest limit of any reference line. A second reference line for assessing the artifact's usefulness can be obtained by directly measuring the judgement capacity of a potential user group for whom the assistance system has

[2] https://github.com/rsmttud/Recommender-System.

[3] https://www.kaggle.com/.

Table 2 Reference and test items of the evaluation design

Evaluation items	Description	Activation of design principles	Role within hypotheses
Random guessing	Discrete uniform distribution	No DP	Reference item for H1
Novice assessment	DSA student survey	No DP	Reference item for H2
TbIAS baseline configuration	Constructed learning Base + Standard text classifiers	DP3 + DP1(*)	Reference item for H3
TbIAS full configuration	Constructed learning Base + Embeddings + Advanced text classifiers	DP3 + DP1 + DP2	Test item for H1 Test item for H2 Test item for H3

been designed (cf. Sect. 6.2). The third reference item is the baseline combination of instantiable design principles. Here, we followed the basic idea of incrementally activating individual design principles, resulting in different design configurations to measure their effects separately (Meth et al. 2015).

Note that in our case, design principles are sequentially interdependent. For example, without an underlying learning base (DP3), no text classifiers for automated NL request processing (DP1) can be applied and vice versa. Similarly, the use of embedding models for automated context extraction (DP2) allows to apply different types of text classifiers, which results in alternative feature instantiations of DP1 with and without (*) the use of embeddings. Therefore, our baseline configuration consists of the constructed learning base and some standard text classification models (cf. Sect. 6.3). By contrast, the test item represents the full configuration based on the entire system of design principles DP1 to DP3 and their associated design features (cf. Sect. 6.4). In this case, DP1 was instantiated with more advanced text classifiers, as already outlined in Sect. 5.3.

Based on the four evaluation items, we propose three hypotheses. First, at the very minimum we assume that the performance of the full configuration is better than pure guessing when signalizing a match between problem statements and DM method classes. Thus, we hypothesize:

H1 Using a TbIAS that is built on an automatically constructed learning base (DP3), supports automated natural language request processing (DP1), and allows automated context extraction (DP2) will result in higher advice quality for DM method class selection than a selection by random guessing.

Second, we expect that the full TbIAS configuration based on all three design principles is also able to outperform the judgement capacity of DM novices and therefore provides useful assistance when no sufficient DM experience is available. Consequently, we hypothesize:

H2 Using a TbIAS that is built on an automatically constructed learning base (DP3), supports automated natural language request processing (DP1), and allows automated context extraction (DP2) will result in higher advice quality for DM method class selection than a selection based on the judgement capacity of DM novices.

Lastly, we expect that the full TbIAS configuration based on all three design principles outperforms the basic configuration due to the additional capability of automatically extracting relevant context. Thus, we hypothesize:

H3 Using a TbIAS that is built on an automatically constructed learning base (DP3), supports automated natural language request processing (DP1), and allows automated context extraction (DP2) will result in higher advice quality for DM method class selection than a TbIAS that is built on an automatically constructed learning base (DP3) and supports automated natural language request processing (DP1(*)).

To measure and report the advice quality for each item, we rely on standard metrics for the evaluation of classification problems that are straightforward to interpret. Specifically, we use the *overall accuracy* as the proportion of correctly classified cases among the total number of cases and the *recall* as the proportion of correctly classified positive cases among the total number of positive cases (Metz 1978). Here, a positive class refers to a specific class of DM methods to be considered (e.g., cluster analysis) in contrast to the remaining method classes. Furthermore, to assess the inter-group differences between the items at a case level, we consider confidence scores instead of binary decisions to express how certain a case was assigned to a particular DM method.

6.2 Novice Assessment

To obtain a representative reference item reflecting the judgement capacity of DM novices, we collected survey data from master level students at a public research university in Germany. Specifically, we recruited 20 students attending an advanced DSA module as subjects, who were still at the beginning of their education with only little

experience in the application of DM methods. Hence, we consider them as representative DM novices. As the module is an elective, we assume that the subjects were intrinsically motivated to answer the questions conscientiously.

The survey consisted of two parts. In the first part, we introduced the survey and asked the subjects to rate their DM knowledge. Specifically, we asked to provide descriptive data about their study background, their general knowledge in DM, and their particular experience with the three method classes of interest (cf. Table 3).

To introduce the second part, the instructor provided a brief overview of the three method classes using an overview slide to ensure that the subjects have a basic understanding of all method classes. Finally, the subjects were asked to read the 60 randomly ordered problem statements carefully and select the DM method class that they consider to be suited best. To avoid any distortion of the results by pure guessing, we asked subjects to indicate whenever they were not sure about a selection, which was offered as a fourth response option. Answering the second part took between 25 and 38 min. The full questionnaire can be found in Appendix H.

The analysis of the survey data shows that on average the DM novices were able to assign 55% of the problem descriptions correctly. Considering the self-assessed method experience in Table 3, one could assume that the majority of incorrectly assigned problem statements belong to the class of FPM. However, there was no remarkable difference between the three classes when considering their individual average recall scores {CL: 0.62, PR: 0.48, FPM: 0.56}. As a pre-test, we provided the questionnaire to three graduate students with more advanced DM experiences, who achieved an average accuracy of 0.91. This ensures that with a certain level of DM experience, the mapping task based on the given validation data can be performed unambiguously.

6.3 Baseline Configuration

In order to compare the artifact with baseline text classification functionality, we implemented several standard classifier algorithms to represent activated DP3 and DP1(*) in the absence of DP2. For this purpose, we considered an SVM with a radial basis kernel and a multilayer perceptron (MLP) with three hidden layers, fifty neurons per hidden layer, a sigmoid activation function and dropout layer. We chose those algorithms to best resemble the text classifiers used in the full design. We omitted the LSTM and GRU architectures from the baseline configuration as it is only trained on a document level. Hence, there are no word or sentence level embeddings as sequential inputs due to the absence of DP2, which renders the use of a sequential model superfluous. Additionally, to represent out-of-the-box behavior, we only considered the algorithms in their most standard configuration without hyperparameter optimization. For model training, we built a vector representation of all documents by adding all the words from the corpora as features using term frequency-inverse document frequency (TF-IDF) weighting on each word for each document (Salton and Buckley 1988). We used only the non-augmented (DEF) and augmented (AUG) datasets for training and withheld the validation data for evaluation purposes.

Table 4 shows that the accuracies with the AUG dataset are superior to those with the DEF dataset for both classifiers. This affects especially the SVM, which falls back to a random guessing level with the non-augmented data, while slightly improving with the augmented data. The same effect can be observed for the MLP classifier with an

Table 3 Student subjects' descriptive data

Semester	Enrolled Studies	DM knowledge	CL experience	PR experience	FPM experience
6.3	Information Systems: 12	1.30 out of 3	2.95 out of 7	3.05 out of 7	1.40 out of 7
(SD = 1.93)	Industrial Engineering: 3	(SD = 0.40)	(SD = 0.86)	(SD = 0.97)	(SD = 0.73)
	Business Administration/Economics: 4				
	Business and Economics Education: 1				

Table 4 Evaluation results of the baseline models trained on different datasets

Classifier	Dataset	Recall CL	Recall PR	Recall FPM	Accuracy
SVM	DEF	0.33	0.33	0.33	0.33
	AUG	0.10	0.35	0.85	0.43
MLP	DEF	0.20	0.60	0.55	0.45
	AUG	0.20	0.60	0.95	0.58

even higher magnitude. We expected these results since the three hidden layers most likely generate abstract features that could better describe the documents compared to the SVM. We can also observe a skewed result distribution that leans towards the class of FPM. This effect would result in the TbIAS to favor the vote of one class over another when unsure. Additionally, we can see that the baseline models perform only very close to random guessing for the class of cluster analysis.

6.4 Full Configuration

Lastly, we evaluated the full configuration of our system design artifact based on the advanced text classifiers, which were trained on distinct embedding models and the two datasets DEF and AUG. Table 5 reveals that the SVM (on DAN:DEF) and the LSTM (N–1) (on FastText:AUG) show the best performances and produce accurate results. In contrast, the KNN and Topic-KNN models (on USE) exhibit the lowest accuracies and are not suitable to realize an adequate mapping. Generally, we observe that the results of the pre-trained USE model lag behind those of the other embedding models and that the concept of transfer learning produces no useful effect in this context. Further, we can see that methods using singular paragraph vectors as input achieve better results on DAN models, whereas the models with N–1-architectures perform better with FastText vectors. This underlines the usefulness of separate models for inference of word and paragraph vectors. Lastly, we see that the DL architectures generally perform better on the augmented data, whereas the classical approaches perform better on the non-augmented data.

In addition to the depicted text classifiers that are based on the embedding models, we also examined the two LDA topic models. The first approach, which is used for direct classification with three topics, reaches an accuracy of 0.7, whereas the second approach based on seven topics and a subsequent SVM classifier only shows poor accuracy of 0.46, demonstrating that the latter approach is not suitable for the given task.

Table 6 Evaluation results of the ensemble models based on weighted averaging

Ensembles	Accuracy
SVM + LSTM (N–1) + GRU (1–1)	0.90
LDA + SVM + GRU (1–1)	0.88
LDA + LSTM (N–1) + GRU (1–1)	0.88

Ultimately, to produce an even more accurate classifier, we built weighted averaging ensembles (Sagi and Rokach 2018) based on the best performing classifiers per each model type. The results of the three top ensembles are illustrated in Table 6. We can see that accuracies up to 90% can be achieved with the combination of an SVM, an LSTM (N–1) and a GRU (1–1). Consequently, this combination was implemented in the final prototype.

6.5 Performance Comparison and Hypothesis Testing

After the assessment of the individual evaluation items, a comparison of the scores reveals that the full configuration based on all three design principles dominates the other reference items. Table 7 summarizes the recall and accuracy results for all four items. We considered only the best-performing TbIAS configurations.

In order to provide even more reliable statements about the inter-group differences and test our design hypotheses H1–H3, we additionally considered the confidence scores for each classification decision. These scores express how certain an algorithm is about a decision. While, for a general evaluation, we want the algorithm to make the right decisions, we also want the algorithm to be sure about it. For example, confidences of {CL: 0.32, PR: 0.32, FPM: 0.36} produce the same decision as confidences of {CL: 0.01, PR: 0.01, FPM: 0.98}, the resulting decision to classify the problem as FPM, however, is less reliable.

For random guessing, we set equal confidences of 0.33 for each class, whereas the scores for the novice assessment were calculated using the relative frequency of subjects

Table 5 Evaluation results of the full design configuration trained on different embeddings and datasets

Classifier	Accuracy					
	FastText:DEF	DAN:DEF	USE:DEF	FastText:AUG	DAN:AUG	USE:AUG
SVM	0.73	0.85	0.67	0.75	0.83	0.77
KNN	0.78	0.82	0.60	0.78	0.80	0.57
Topic-KNN	0.62	0.72	0.50	0.65	0.70	0.55
LSTM (N–1)	0.77	0.73	0.82	0.85	0.83	0.78
GRU (N–1)	0.83	0.73	0.82	0.83	0.80	0.78
LSTM (1–1)	0.72	0.80	0.70	0.75	0.83	0.77
GRU (1–1)	0.78	0.82	0.70	0.73	0.83	0.77

Table 7 Overall performance comparison for the different evaluation items

Evaluation item	Recall CL	Recall PR	Recall FPM	Accuracy
Random guessing	0.33	0.33	0.33	0.33
Novice assessment	0.62	0.48	0.56	0.55
TbIAS baseline configuration	0.20	0.60	0.95	0.58
TbIAS full configuration	0.85	1.00	0.85	0.90

voting for the right DM method class. For both TbIAS configurations, we used the scores derived from the classifiers. An overview of all confidence scores for each problem statement can be found in Appendix G.

To test our hypotheses, we conducted a two-stage analysis. First, we performed an ANOVA with the evaluation item as the independent variable and the confidence as the dependent variable. We applied the Bartlett test, the Levene test, and the Brown–Forsythe test for unequal variances to check the prerequisites for the ANOVA (Blanca et al. 2018). The tests returned indication of unequal variances. We therefore applied a robust version of the standard ANOVA by Wilcox (1989) to adjust for these circumstances. The results of the ANOVA are depicted in Table 8.

After the ANOVA returned a significant result for the overall test that at least two evaluation items are different, we performed a post hoc independent *t* test with Bonferroni adjustment to compare them. The *t* tests returned significant results on H1 and H2 at the 0.01 level, and on H3 at the 0.05 level. This supports our three hypotheses and confirms that our design principles indeed increase the advice quality using natural language problem descriptions. Table 9 shows the results of the test.

6.6 Robustness Checks

In addition to the evaluation based on fixed validation data, we also conducted several robustness checks with the full TbIAS design configuration to ensure the transferability of the results to other circumstances than those given within the currently considered problem descriptions. Specifically, we investigated the impact on the confidence scores when (i) replacing method-centric keywords, (ii) replacing domain entities, and (iii) modifying the length of the problem descriptions. For each check, several examples can be found in Appendix I.

The first check revealed that the choice of keywords has a high impact on the confidence scores, as keywords with a stronger semantic connection to a certain DM method generally increased the confidence scores of the correct class, whereas weaker keywords resulted in a decrease. Likewise, keywords, which are associated with contrary DM methods, cause a problem statement to be assigned to another class.

For the second check, we systematically replaced characteristic domain entities. We found that the type of problem surroundings also has a certain impact on the confidence scores. For example, the TbIAS generally

Table 8 ANOVA results

Source	Degrees of freedom	Sum of squares	Mean square	*F* ratio	Prob > *F*
ITEM	3	4.28	1.426	26.45	< .0001*
Error	236	12.73	0.054		
Corrected total	239	17.01			

*The results are statistically significant

Table 9 Post-hoc *t* test results of hypotheses H1–H3

Hypothesis	Level	versus Level	Difference	*p* Value
H1	Full configuration	Random guessing	0.369	< .0001*
	Baseline configuration	Random guessing	0.264	< .0001*
	Novice assessment	Random guessing	0.219	< .0001*
H2	Full configuration	Novice assessment	0.147	0.0006*
H3	Full configuration	Baseline configuration	0.102	0.0165*
	Baseline configuration	Novice assessment	0.044	0.2968

*The results are statistically significant

tended to drift towards FPM whenever a problem statement contained sales-related terms, such as "client" or "selling". Presumably, this kind of distortion is caused by the fact that a predominant portion of academic articles (as the fundamental foundation of the learning base) investigate FPM mostly for sales problems. Thus, this bias is an apparent current limitation, as the system should be able to classify problems independently of the underlying domain.

In the third check, we iteratively modified the length of the problem descriptions by either reducing them to the central statement or adding noise. Hereby, we could also notice an influence of additional noise, but despite a decreasing ratio of keywords to total words, we could still obtain relatively stable confidence values.

7 Discussion

7.1 Theoretical and Practical Contributions

The field of DSA is continuously evolving by means of new and innovative assistance systems to further improve and simplify the execution of data analysis projects. For example, RapidMiner provides some illustrative examples, such as an "Auto Model" function that automatically suggests the best ML techniques based on a given data input or the "Wisdom of Crowds" function that recommends analysis operators and parameters derived from an internal best-practice knowledge base (RapidMiner 2019). However, despite broad assistance systems in existing DSA platforms (Serban et al. 2013), there is no IAS that suggests the best suitable class of analysis methods based on a problem description expressed in natural language. Our TbIAS is a novel solution which automatically selects suitable class of DM methods for a given problem space and therefore provides an improvement (Gregor and Hevner 2013) on the state-of-the-art.

Our study is a comprehensive technical investigation and evaluation of multiple text processing and classification method pipelines towards the creation of a system design artifact instantiation of a TbIAS. In an evaluation, we compared and incorporated a broad spectrum of approaches from disciplines like TM, NLP, and DL to determine the best performing approaches for the given mapping task. Simultaneously, we investigated several effects of associated procedures and concepts, such as data augmentation, transfer learning, ensemble learning, hyperparameter tuning, and the inference from word versus paragraph vectors, which all have their particular role within the system design and should therefore not be neglected. These analyses resulted in a working prototype, which increases advice quality in comparison to related approaches.

We have abstracted from the technical details of the concrete implementation to provide design principles as prescriptions for the design of a class of systems that assist novices in DM method selection. We did so by codifying facets of an effective solution entity informed by prior knowledge for entity realization. Our design requirements are a projection of prior design knowledge, which was particularly helpful for establishing a balanced selection of design requirements. While we do not claim that our resulting design principles should replace current practices or make them obsolete, we consider them an enrichment of current practices that offers an innovative and economical perspective on DM method selection for DSA projects. Our design features can be traced back to our design requirements via the design principles. They improve the conceptual understanding and the relevance of the system design artifact we propose.

Our design artifact can be considered as a composition of smaller artifacts. They for themselves as well as their sum can serve as a sound baseline for further development and research activities. According to the design knowledge map concept of vom Brocke et al. (2020), this can involve, for example, the artifact's conceptual and technical projection to other problem spaces for method selection (generalization) or the enhancement of fitness due to the current limitations detailed below (amplification).

Lastly, since the area of DSA is highly interdisciplinary, the TbIAS can generally help to bridge the gap between analytically oriented method knowledge and domain-specific expertise, especially at the initiation of a DSA project, where data may not be even available yet. Thus, on the one hand, our artifact can assist DM novices at the beginning of their DSA projects as an entry point to obtain a better understanding of possible solution directions as well as necessary foundations for the relevant DM methods. On the other hand, the TbIAS can serve as an efficient communication tool, which DSA experts with sufficient DM qualifications could use together with the respective domain experts to develop a common view on the peculiarities of upcoming DSA projects in order to create and discuss plausible solution blueprints for their realization. To this end, the artifact could either be used as a standalone application or as a novel add-on embedded into existing DSA platforms. It is also conceivable to combine it with other assistance systems, in which case the DM method is first determined using a provided problem statement and then a concrete DM algorithm with suitable parameterization is suggested on the basis of input data.

7.2 Limitations

As with any research, our work has limitations. First, we acknowledge that in practice problem descriptions for DSA can be inconsistent and superficial and must be pre-processed before they provide meaningful input to our TbIAS. However, this is a step that would be necessary in any setting also without our TbIAS. Further, we must acknowledge that DSA problems in industry can be more complex than those illustrated throughout the article and in the appendix. This includes for example that DSA problems are rarely straightforward but rather consist of numerous smaller sub-problems, which do not necessarily have to be expressed by means of a problem description based on ambiguous natural language. In addition, not every DSA problem can be addressed with an explicit DM method and an ultimate decision may not be possible without considering the data that is to be analyzed. Nevertheless, we still believe that a considerable amount of problems in practical environments can be supported by our TbIAS as long as the corresponding problem descriptions show an appropriate level of granularity. In addition, our prototype can only evaluate input in the English language. We assume its performance to be similar for other languages but have not yet evaluated this.

A second limitation is the missing general availability of real-world data from practice. Due to a lack of labelled problem descriptions, we extracted a large number of definitory and methodical statements from academic articles, which contained semantically relevant constructs of the selected DM method classes for training purposes. However, since those records do not truly reflect the structure of real-world problem descriptions, it was not reasonable to use them simultaneously for validation purposes, which is why the collection of real problem descriptions is still of central importance. To this end, we started our studies with a limited amount of collected examples to demonstrate the general feasibility of the approach, resulting in very promising classification results. Accordingly, a subsequent step is to collect a larger number of problem descriptions in cooperation with several industry partners to use them for a broader validation to ensure the generalizability of the results.

A third limitation is our focus on support for DM novices. While the system is superior to all other considered baseline models, it will most likely be inferior to a group of DM experts. Hence, the support gained from the TbIAS is limited to advice suitable for problems of low to medium complexity for inexperienced users. It is not a replacement for DM experts. Hence at this stage, for a comprehensive DSA project DM experts are still necessary. Furthermore, the evaluation scope of our study is focused on its technical aspects, in particular the validation of the analytical processing pipelines and the overall feasibility. Nevertheless, to ensure the artifact's suitability in practical settings for different application scenarios, it is also necessary to carry out further steps of evaluation by considering socio-technical aspects, such as usefulness, usability, comprehensibility or the range of applicability for different target groups.

A fourth limitation concerns the robustness of our current prototype. While the evaluation generally showed high accuracies using external out-of-sample data, our robustness checks revealed that in some cases specific domain entries can also have an impact on a DM method class's tendency. We suspect the cause to be an imbalance in the automatically constructed learning base. In subsequent research, we need to ensure that the learning data covering the individual DM method classes is distributed evenly across a broader variety of domains. In this context, we also plan to introduce an explanatory AI component to better trace and comprehend which entries are responsible for classifier decisions. In this way, we expect to incrementally construct an increasingly robust learning base that guarantees an even higher degree of domain independency. Likewise, we plan to further expand the number of target classes by including other method classes, such as anomaly detection, process mining, or network analysis.

8 Conclusion

Our research goal was to search for an artifact that assists DM novices to select DM methods during the initiation of DSA projects based on problem descriptions expressed in domain-specific language. We first drew a connection to related work and outlined how our approach is distinct from existing approaches in established fields such as meta-learning or QAS. Subsequently, we carried out a comprehensive study following a DSR methodology. As the main contributions, our research provides (i) a sound collection of design requirements, design principles, and design features for a TbIAS, (ii) the creation of several datasets to build a suitable learning base, (iii) the technical investigation and evaluation of multiple text classification pipelines using approaches from TM, NLP, and DL, and (iv) the concrete implementation of a TbIAS prototype. The resulting instantiation is based on the best performing pipelines and shows promising accuracies when applied to validation data.

With our research, we have not only provided a first functional prototype for DM method selection, but also offered conceptual guidance and methodical investigations as an incentive for other researchers and practitioners to participate in the joint development of a future IAS for DSA which support mapping problems that go beyond the

scope of traditional recommender tools, meta-learning approaches, or other conventional assistance systems.

Acknowledgement Open Access funding provided by Projekt DEAL.

References

Aggarwal CC, Zhai C (eds) (2012) Mining text data. Springer, Boston

Allahyari M, Pouriyeh SA, Assefi M, et al (2017) A brief survey of text mining: classification, clustering and extraction techniques. In: Proceedings of KDD bigdas, Halifax

Athenikos SJ, Han H (2010) Biomedical question answering: a survey. Comput Methods Programs Biomed 99(1):1–24. https://doi.org/10.1016/j.cmpb.2009.10.003

Baskerville R, Pries-Heje J (2019) Projectability in design science research. J Inf Technol Theory Appl 20(1):53–76

Bishop C (2006) Pattern recognition and machine learning. Springer, New York

Blanca MJ, Alarcón R, Arnau J et al (2018) Effect of variance ratio on ANOVA robustness: might 1.5 be the limit? Behav Res Methods 50:937–962. https://doi.org/10.3758/s13428-017-0918-2

Blei DM, Ng AY, Jordan MI (2003) Latent Dirichlet allocation. J Mach Learn Res 3:993–1022

Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans Assoc Comput Linguist 5:135–146. https://doi.org/10.1162/tacl_a_00051

Brodsky A, Shao G, Krishnamoorthy M, et al (2015) Analysis and optimization in smart manufacturing based on a reusable knowledge base for process performance models. In: 2015 IEEE international conference on big data. IEEE, Santa Clara, pp 1418–1427

Campos R, Mangaravite V, Pasquali A et al (2018) A text feature based automatic keyword extraction method for single documents. In: Pasi G, Piwowarski B, Azzopardi L, Hanbury A (eds) Advances in information retrieval. Springer, Cham, pp 684–691

Cer D, Yang Y, Kong S, et al (2018) Universal sentence encoder. arXiv:180311175 [cs]

Chandra L, Seidel S, Gregor S (2015) Prescriptive knowledge in IS research: conceptualizing design principles in terms of materiality, action, and boundary conditions. In: 2015 48th Hawaii international conference on system sciences. IEEE, pp 4039–4048

Cho K, van Merrienboer B, Bahdanau D, Bengio Y (2014) On the properties of neural machine translation: encoder–decoder approaches. In: Proceedings of the eighth workshop on syntax, semantics and structure in statistical translation. Association for Computational Linguistics, Doha, pp 103–111

Choinski M, Chudziak JA (2009) Ontological learning assistant for knowledge discovery and data mining. In: 2009 International multiconference on computer science and information technology. IEEE, Mragowo, pp 147–155

Dabab M, Freiling M, Rahman N, Sagalowicz D (2018) A decision model for data mining techniques. In: 2018 Portland international conference on management of engineering and technology. IEEE, Honolulu, pp 1–8

Danubianu M (2008) Design of an expert system for efficient selection of data mining method. Universitatea Tehnică Gheorghe Asachi, Iaşi

Dave RN (1991) Characterization and detection of noise in clustering. Pattern Recognit Lett 12(11):657–664. https://doi.org/10.1016/0167-8655(91)90002-4

Debortoli S, Müller O, vom Brocke J (2014) Comparing business intelligence and big data skills: a text mining study using job advertisements. Bus Inf Syst Eng 6:289–300. https://doi.org/10.1007/s12599-014-0344-2

Drechsler A, Hevner AR (2018) Utilizing, producing, and contributing design knowledge in DSR projects. In: Chatterjee S, Dutta K, Sundarraj RP (eds) Designing for a digital and globalized world. Springer, Cham, pp 82–97

Eckert S, Ehmke JF (2017) Classification of data analysis tasks for production environments. In: Abramowicz W, Alt R, Franczyk B (eds) Business information systems workshops. Springer, Cham, pp 399–407

Ester M, Kriegel H-P, Sander J, Xu X (1996) A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the second international conference on knowledge discovery and data mining. AAAI Press, pp 226–231

Fayyad U, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery in databases. AI Mag 17(3):37–54. https://doi.org/10.1609/aimag.v17i3.1230

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge

Gregor S, Hevner AR (2013) Positioning and presenting design science research for maximum impact. MIS Q 37(2):337–355. https://doi.org/10.25300/MISQ/2013/37.2.01

Guda V, Sanampudi SK, Manikyamba IL (2011) Approaches for question answering systems. Int J Eng Sci Technol 3(2):990–995

Gupta P, Gupta V (2012) A survey of text question answering techniques. Int J Comput Appl 53(4):1–8. https://doi.org/10.5120/8406-2030

Hesenius M, Schwenzfeier N, Meyer O, et al (2019) Towards a software engineering process for developing data-driven applications. In: Proceedings of the 7th international workshop on realizing artificial intelligence synergies in software engineering. IEEE Press, Piscataway, pp 35–41

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hogl OMJ (2003) Eine wissensbasierte Benutzerschnittstelle für das invisible data mining. Dissertation, Friedrich-Alexander-Universität Erlangen-Nürnberg

Hotho A, Nürnberger A, Paass G (2005) A brief survey of text mining. LDV Forum 20:19–62

Huber S, Wiemer H, Schneider D, Ihlenfeldt S (2019) DMME: data mining methodology for engineering applications – a holistic extension to the CRISP-DM model. Procedia CIRP 79:403–408. https://doi.org/10.1016/j.procir.2019.02.106

Iyyer M, Manjunatha V, Boyd-Graber J, Daumé III H (2015) Deep unordered composition rivals syntactic methods for text classification. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th

international joint conference on natural language processing. Association for Computational Linguistics, Beijing, pp 1681–1691

Jurafsky D, Martin JH (2008) Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd edn. Pearson Prentice Hall, Upper Saddle River

Kerschke P, Hoos HH, Neumann F, Trautmann H (2019) Automated algorithm selection: survey and perspectives. Evol Comput 27(1):3–45. https://doi.org/10.1162/evco_a_00242

Kowsari K, Jafari Meimandi K, Heidarysafa M et al (2019) Text classification algorithms: a survey. Information 10(4):150. https://doi.org/10.3390/info10040150

Kulkarni SB, Deshmukh PD, Kale KV (2013) Syntactic and structural divergence in English-to-Marathi machine translation. In: 2013 international symposium on computational and business intelligence. IEEE, New Delhi, pp 191–194

Kurgan LA, Musilek P (2006) A survey of knowledge discovery and data mining process models. Knowl Eng Rev 21(1):1–24. https://doi.org/10.1017/S0269888906000737

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444. https://doi.org/10.1038/nature14539

Lemke C, Budka M, Gabrys B (2015) Metalearning: a survey of trends and technologies. Artif Intell Rev 44:117–130. https://doi.org/10.1007/s10462-013-9406-y

Maedche A, Morana S, Schacht S et al (2016) Advanced user assistance systems. Bus Inf Syst Eng 58(5):367–370. https://doi.org/10.1007/s12599-016-0444-2

Manyika J, Chui M, Brown B et al (2011) Big data: the next frontier for innovation, competition, and productivity. McKinsey Global Institute, Amsterdam

Meth H, Mueller B, Maedche A (2015) Designing a requirement mining system. J Assoc Inf Syst 16(9):799–837. https://doi.org/10.17705/1jais.00408

Metz CE (1978) Basic principles of ROC analysis. Sem Nucl Med 8(4):283–298. https://doi.org/10.1016/S0001-2998(78)80014-2

Mihalcea R, Tarau P (2004) TextRank: bringing order into text. In: Proceedings of the 2004 conference on empirical methods in natural language processing. Association for Computational Linguistics, Barcelona, pp 404–411

Mikalef P, Krogstie J (2019) Investigating the data science skill gap: an empirical analysis. In: 2019 IEEE global engineering education conference (EDUCON). IEEE, Dubai, pp 1275–1284

Mikolov T, Corrado GS, Chen K, Dean J (2013) Efficient estimation of word representations in vector space. In: Proceedings of the international conference on learning representations (ICLR 2013). Scottsdale

Offermann P, Blom S, Schönherr M, Bub U (2010) Artifact types in information systems design science – a literature review. In: Winter R, Zhao JL, Aier S (eds) Global perspectives on design science research. Springer, Heidelberg, pp 77–92

Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2007) A design science research methodology for information systems research. J Manag Inf Syst 24(3):45–77. https://doi.org/10.2753/MIS0742-1222240302

Perone CS, Silveira R, Paula TS (2018) Evaluation of sentence embeddings in downstream and linguistic probing tasks. arXiv:180606259

RapidMiner (2019) Lightning fast unified data science platform | RapidMiner. In: RapidMiner. https://rapidminer.com/products/. Accessed 15 Jul 2019

Rupp C (2014) Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil, 6th edn. Hanser, München

Sagi O, Rokach L (2018) Ensemble learning: a survey. Wiley Interdiscip Rev Data Min Knowl Discov 8(4):e1249. https://doi.org/10.1002/widm.1249

Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. Inf Process Manag 24(5):513–523. https://doi.org/10.1016/0306-4573(88)90021-0

Salton G, Wong A, Yang CS (1975) A vector space model for automatic indexing. Commun ACM 18(11):613–620. https://doi.org/10.1145/361219.361220

Schumann C, Zschech P, Hilbert A (2016) Das aufstrebende Berufsbild des Data Scientist: Vom Kompetenzwirrwarr zu spezifischen Anforderungsprofilen. HMD Praxis der Wirtschaftsinformatik 53(4):453–466. https://doi.org/10.1365/s40702-016-0214-0

Serban F, Vanschoren J, Kietz J-U, Bernstein A (2013) A survey of intelligent assistants for data analysis. ACM Comput Surv 45(3):1–35. https://doi.org/10.1145/2480741.2480748

Vainshtein R, Greenstein-Messica A, Katz G, et al (2018) A hybrid approach for automatic model recommendation. In: Proceedings of the 27th ACM international conference on information and knowledge management. ACM Press, Torino, pp 1623–1626

vom Brocke J, Winter R, Hevner AR, Maedche A (2020) Accumulation and evolution of design knowledge in design science research – a journey through time and space. J Assoc Inf Syst (forthcoming)

Wang X, Huang C, Yao L et al (2018) A survey on expert recommendation in community question answering. J Comput Sci Technol 33(4):625–653. https://doi.org/10.1007/s11390-018-1845-0

Webster J, Watson RT (2002) Analyzing the past to prepare for the future: writing a literature review. MIS Q 26(2):13–23

Wilcox RR (1989) Adjusting for unequal variances when comparing means in one-way and two-way fixed effects ANOVA models. J Educ Stat 14(2):269–278. https://doi.org/10.3102/10769986014003269

Wirth R, Hipp J (2000) CRISP-DM: towards a standard process model for data mining. In: Proceedings of the fourth international conference on the practical application of knowledge discovery and data mining. pp 29–39

Zschech P (2018) A taxonomy of recurring data analysis problems in maintenance analytics. In: Proceedings of the 26th European conference on information systems. Portsmouth

Zschech P, Fleißner V, Baumgärtel N, Hilbert A (2018) Data science skills and enabling enterprise systems: Eine Erhebung von Kompetenzanforderungen und Weiterbildungsangeboten. HMD Praxis der Wirtschaftsinformatik 55(1):163–181. https://doi.org/10.1365/s40702-017-0376-4

Zschech P, Heinrich K, Horn R, Höschele D (2019) Towards a text-based recommender system for data mining method selection. In: Proceedings of the 25th Americas conference on information systems. Cancún