A Systematic Literature Review of Model-Driven Engineering using Machine Learning

Ana C. Marcén, Antonio Iglesias, Raúl Lapeña, Francisca Pérez, Carlos Cetina

Abstract—Model-driven engineering (MDE) is a software engineering paradigm based on the systematic use of models. Over the past few years, engineers have significantly increased the use of MDE, which has been reported as a successful paradigm for developing industrial software. Recently, there have also been remarkable advancements in the Artificial Intelligence (AI) domain, with a significant increase in advanced Machine Learning (ML) techniques. The advances in both fields have led to a surge in works that dwell within the intersection of ML and MDE. This work places the focus on systematically reviewing works that leverage ML to solve MDE problems. We have reviewed a total of 9,194 papers, selecting 98 studies for further analysis. The results of our Systematic Literature Review (SLR) bring light to the current state of the art and trends in the field, discussing the drift in the usage of the different available ML techniques along with the remaining research gaps and open challenges. Our SLR has the potential to produce a positive impact in the research community by steering it towards ML techniques that have been successfully applied to solve MDE challenges.

Index Terms—Model-Driven Engineering, Machine Learning, Systematic Literature Review

1 Introduction

Model-driven engineering (MDE) is a software engineering paradigm based on the systematic use of models as primary artifacts throughout the software development cycle [1]. With the MDE paradigm, models are used to capture and design the characteristics of software systems: models can be run and interpreted at run-time, automatically transformed into code, or used to design and derive other software artifacts. Major players in the software engineering field and in the requirements engineering field foresee a broad adoption of MDE techniques [2, 3], since they improve the productivity, quality, and performance of software in industrial scenarios that require more abstract approaches than mere coding [1].

While models have not replaced source code as a means of software development so far, MDE has been reported as a successful paradigm to develop industrial software [1, 4]. Real-world examples can be found in the BSH group, where models are used to generate the C++ firmware that controls their induction hobs (sold under the brands of Bosch and Siemens, among others) [5], and in CAF (www.caf.net/en), where models serve as a means for developing the software that controls the trains they manufacture [6]. Another example can be found in the area of video games, where models are used to obtain the software in the video game engines (for example, Unreal Engine blueprint models).[1]

This new discipline is gaining traction within the software engineering field and becoming a popular alternative for industrial software development under the commercial names of low-code and no-code. While engineers have placed the focus on evolving model-based technologies, decreasing the costs for the maintenance and documentation of MDE-based software systems in the process [7]–[10], several organizations and companies (including the Object Management Group, the Eclipse community, IBM, and Microsoft) are currently proposing a wide variety of different techniques and environments claiming to support MDE. However, MDE practices vary depending on the characteristics of the systems and sectors in which they are applied. Some stakeholders use MDE for requirements elicitation and to communicate with colleagues and peers, while others use them for automated code generation. Different sections of the same company might use different MDE artifacts and approaches for different purposes within the software cycle [11]. Moreover, MDE suffers from largely unresolved linguistic challenges and search complexity issues caused by the characteristics inherent to MDE artifacts.

Increasingly, MDE researchers are looking to Artificial Intelligence (AI) as an opportunity to improve or optimize their approaches. Various Machine Learning (ML) techniques have been applied with greater or lesser success in MDE. Decision trees, regression trees, classification trees, nearest neighbour algorithms, neural nets, bayesian belief networks, bayesian nets, association rules, support vector machines, or support vector regression algorithms are just some of the examples that make up the main current trends in the literature.

In fact, the number of publications applying ML techniques in MDE has tripled in the last five years. This rapid growth means that it is still unclear to what extent MDE problems can be solved with the help of machine learning techniques. Moreover, MDE is a very broad concept, and researchers and practitioners focus their work on various

- *Ana C. Marcén, Antonio Iglesias Raúl Lapeña, Francisca Pérez, and Carlos Cetina are with the SVIT Research Group of Universidad San Jorge, Zaragoza, Spain.*
  E-mail: {acmarcen, aiglesias rlapena, mfperez, ccetina@usj.es.
- *Carlos Cetina is also with the Computer Science Department of University College London, London, United Kingdom.*
- *Antonio Iglesias is also with the Research Center on Software Production Methods (PROS), Universitat Politècnica de València, Valencia, Spain.*

Manuscript received June X, 2020; revised .

1. https://www.unrealengine.com/

models (UML, domain models, conceptual models, etc.) and diverse tasks using these models (documentation, simulation, verification and validation, testing, code generation, etc.). While a specific machine learning technique may provide a remarkable solution for one type of model and task, a similar solution may not be effectively applied on other models and tasks due to the way the models are encoded to apply ML techniques, the presence or absence of text in the models, or even the size of the models. Also, a holistic view on what MDE problems are solved using ML techniques, the ML techniques applied, the types of models, and the challenges in using ML techniques for MDE is still missing.

The goal of this work is to identify, classify, and understand approaches that support MDE using ML techniques. To achieve our goal, we have applied a well-established methodology from the software engineering research communities called Systematic Literature Review (SLR) [12]. In accordance with the SLR steps, we examined 6,255 research studies (9,194 with the snowballing). Of these, 98 were selected based on inclusion and exclusion criteria and analyzed, in depth.

The main contributions of this study are:

- the identification of the MDE problems that have been solved using ML techniques. Although we do not explicitly use the term taxonomy, we classify the studies based on diverse categories (ML technique, Type of model, MDE problem, etc.). This can be a reusable classification framework for understanding, classifying, and comparing present and future work on the application of ML techniques in MDE.
- the presentation of progress in terms of quality, acceptance, and frequency of use of MDE approaches leveraging ML techniques.
- the analysis of current limitations and challenges as well as trends in publications related to MDE approaches using ML techniques.

This paper presents a systematic literature review, building towards a complete, comprehensive, and replicable picture of MDE research leveraging ML techniques and helping researchers and practitioners identify the state, limitations, and trends of current research on the topic.

The remainder of this paper is structured as follows: Section 2 motivates our research and presents the background for our study. Section 3 presents the related work in this area. Section 4 describes the research method used. Section 5 presents the results of the SLR performed based on the research questions. Section 6 and Section 7 discuss the results and the threats to validity, respectively. Finally, Section 8 concludes the paper.

2 Background

This section presents the necessary definitions to describe the framework of our SLR. Furthermore, it also includes a brief description of pre-processing models for applying ML techniques.

Model is a key term for this work, and is ambiguous when combining MDE and ML research areas. In MDE, a model is an abstraction of a system that is often used to replace the system under study [13]–[15]. On the other hand, in ML, a model is a rule-set that is learned when comparing feature vectors automatically using a ML technique [16]. In this work, we use the term *model* to refer to MDE models, and we adopt a synonym (i.e., *classifier*) to refer to ML models.

MDE stands for Model-Driven Engineering, which is a broader methodology that emphasizes the systematic use of models in both the development phase and throughout the entire engineering process. MDE aims to provide a systematic and efficient way to manage the complexity of software systems by using models to capture and express the design and behavior of the system at various levels of abstraction. While MDE focuses on the entire engineering process, Model-Driven Development (MDD) is an approach within MDE that emphasizes the use of models as a primary artifact throughout the development lifecycle. It includes activities such as model creation, transformation, analysis, and code generation. Model-Driven Architecture (MDA), which is related to MDD, is an initiative by the Object Management Group (OMG) to provide guidelines and standards for software development based on models.

Therefore, models are turned into primary artifacts throughout engineering processes, specifically throughout the development lifecycle. Although some modeling assets use a textual syntax (e.g., Object Constraint Language or ATL model transformations), most of models mainly use different shapes (e.g., rectangles, ellipses, etc.) to graphically represent objects, classes, relationships, etc. These shapes are complemented with text, lines, and arrowheads, forming diagrams that effectively illustrate the structure, behavior, or design of software systems. These models have two main advantages. First, the level of abstraction provided by the models helps the researchers or developers to easily understand the system. Second, transforming the models into source code facilitates the development of new software systems.

However, the use of models also brings with it certain problems, for example, how to transform a model into code or how to correctly specify the models. Some of the existing problems are addressed by ML techniques. Part of the contribution of this paper is to identify which MDE problems are currently being addressed by ML techniques.

ML is a subset of artificial intelligence (AI) that focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed. ML systems learn from data patterns and experiences, allowing them to improve their performance over time. ML is widely used in tasks such as image recognition, natural language processing, recommendation systems, and predictive analytics across diverse industries. For example, MDE large systems require complexity, size, and several models for their specifications. Therefore, engineers need approaches to find elements within these models, locate errors, validate the models, or even compare them. An ML classifier can be trained from correct and incomplete models so that the classifier can validate other models. Similarly, an ML classifier can be trained from errors, so that given a part of the model, the classifier can report whether it contains an error or is correct.

There are various types of ML techniques, including supervised learning, unsupervised learning, and reinforcement learning. There are different ways to apply ML techniques depending on these types, the hyperparameters of the technique, and the input data. We are not going to detail how to apply each specific technique or how to tune their hyperparameters. However, we do include a brief description of the input data for the context of this work.

In our case, the input data is composed of models. These models must be pre-processed in order to apply ML techniques. For example, some ML techniques (e.g., support vector machine) require feature vectors. A feature vector contains variables to represent the relevant information in predicting the output [17]. This means that, instead of the graphical representation of the model, we need to encode the model as a feature vector. There are two main ways to achieve this [SLR56]:

- Based on the **shape of the model elements** (e.g., the number of connecting lines, the rectangle size, or the vertical/horizontal alignment of shapes in the model).
- Based on the **type of the model elements** (e.g., the number of associations, aggregations, compositions, and generalizations in a class diagram).

Note that, although the examples are based on counting the number of occurrences, there are other ways of encoding features. For example, we could check whether a certain element is present or not in the model using a Boolean variable.

In addition, to encode a model into a feature vector, we can use both the semantics and text of the elements in the model. For example, we can count the occurrences of a relationship between an element of type X and an element of type Y (i.e., semantics), or count the occurrences of a specific term in the labels of the model elements (i.e., text).

3 Related Work

Our survey contributes to the available Software Engineering literature by comprehensively examining the landscape of ML techniques applied to MDE problems. Regarding the available literature, there is one work in the field that is close to ours in the form of a 2023 book chapter by Davide Di Ruscio et. al. [18] summarizing recent applications of ML approaches to support modeling ecosystems and envisioning a roadmap for the deployment of ML techniques in the MDE domain. While the book chapter puts the focus on the deployment of ML in MDE ecosystems, our survey goes beyond by identifying MDE problems addressed through ML techniques, presenting progress in the field and analyzing the current trends, limitations, and challenges in a comprehensive manner.

There are plenty of technical differences between both studies. Unlike the book chapter, which covers the period from 2012 to 2022, our survey incorporates a broader temporal perspective (with no limit towards the past and up to December 2023), allowing for a more complete assessment of recent developments. Additionally, while the book chapter relies only on Scopus for data retrieval, our survey extends the literature coverage by including data from multiple repositories, including Scopus, Web of Science (WOS), ACM, and IEEE. Furthermore, our survey employs a complete and formally defined query methodology through PICO, as opposed to the more generic nature of the query used in the book chapter, which includes broader terms such as ’Artificial Intelligence’, ’bot’, or ’Machine Learning’. The generic nature of the query causes the book chapter to omit older papers that did not include ’Machine Learning’ as a term in their title, abstract, or keywords. As an example, the book chapter found 2 papers regarding Bayesian Networks where we found 8 through our query. In contrast to the book chapter, we also utilize snowballing techniques, both backwards and forwards, to ensure a thorough identification of relevant literature. Moreover, the book chapter does not present research questions, whereas our survey defines research questions through the Goal-Question-Metric approach, enabling a systematic and structured analysis of the field. In terms of classification, the book chapter considers only MDE problems and ML techniques, where our survey categorizes findings according to types of models, ML techniques, MDE problems, work maturity, type of tools, SE activities, and work limitations. Finally, our survey presents a detailed analysis of threats to validity, tailored specifically for Systematic Literature Reviews (SLRs), a feature that is also absent in the book chapter.

Beyond the technical aspects, there are also key differences in the obtained results and conclusions between the two works. In the book chapter, less papers are obtained and analyzed than in our work (34 vs. 98), and the venues under scrutiny vary significantly from those of our study. While the book chapter only indicates the number of papers in conferences and in journals, we detail the number of papers per conference and per journal. While De Ruscio et. al. only consider the number of papers regarding tasks and techniques, we also show the yearly trends of both, and provide additional information by considering the types of models in use. Our results align with the book chapter statement that most of the papers regard modeling assistance in the form of the specification of models, but differ in the number of papers and research trends in other key MDE categories: for instance, we consider the classification of models and metamodels as separate tasks, and found evidence that supports model analysis as a cornerstone research task for the community. Overall, we consider that our work provides a more extensive view of the field and its development and maturity, revealing works and trends that the previously existing literature omitted.

Apart from the aforementioned book chapter, to the best of our knowledge, there are no other systematic reviews that focus on the application of ML to solve MDE problems. There is a recent work that explores the intersection of MDE with ML [19] which focuses on reviewing studies that apply MDE to systems with ML components. However, our paper explores the intersection in the opposite direction, reviewing studies that apply ML techniques to solve MDE problems.

Other works closely related to ours can be found in systematic reviews that focus on the application of MDE in Software Engineering. Since MDE is a software engineering methodology, it is applied to solve different software tasks. For example, it is used to support the development of secure systems, to collect the requirements of a system, or to systematize the software development. Specifically, there are some reviews that have tackled topics of this kind.

Loniewski et. al. [3] studied how to manage requirements during the MDE process while simultaneously stressing the benefits of automation, presenting a systematic review of the use of requirements engineering techniques in MDE processes and their automation level. Their results showed that, although MDE techniques are used to a great extent in platform-independent models, platform-specific models, and at code level, most MDE approaches use only partially defined requirements models or even natural language at the requirements level. They additionally identified several research gaps such as a need for more efforts to explicitly deal with requirements traceability and the provision of better tool support.

Santiago et. al. [20] place the focus on dealing with traceability in software development, analyzing the state of the art in traceability management in the context of Model-Driven Engineering. Their results highlighted that the most addressed operations at the time were storage, CRUD, and visualization, while the most immature operations were the exchange and analysis of traceability information.

Uzun and Tekinerdogan [21] analyzed Model-Driven Architecture-Based Testing (MDABT) approaches. Their SLRs show that although MDABT is a generic process, the available approaches differed in various ways with different goals, modeling abstractions, and results, and they concluded that the potential of MDABT had not been fully exploited yet.

Raibulet et. al. [22] explored and described the state of the art for model-driven approaches that support reverse engineering. The solutions were classified according to the models in use, the transformations applied to the model, the tools used for model definition, extraction, and transformation, the level of automation that the tools reached, their genericity, extensibility, automation of the reverse engineering process, and coverage of the full or partial source artifacts.

Nguyen et. al. [23] focused on Model-Driven Security (MDS), a specialized Model-Driven Engineering research area for supporting the development of secure systems. The results of this SLR showed the overall status and limitations of the key approaches of MDS, finding that developing domain-specific languages plays a key role in many MDS approaches. Their results suggested the need for addressing multiple security concerns more systematically and simultaneously, for tool chains supporting the MDS development cycle, and for more empirical studies on the application of MDS methodologies.

Alfraihi and Lano [24] combined agile development and Model-Driven Engineering, identifying the main characteristics of Agile Model-Driven Development (Agile MDD) approaches, as well as the benefits and the problems of adopting those approaches. Their results showed that agile development and MDE can coexist and benefit from their integration.

Tufail et. al. [25] considered Model-Driven Engineering (MDE) as a significant method to cope with the problems of mobile application development for cross platform and chose to analyze and summarize the trends, tools, and techniques in that particular context, identifying 11 models and 21 tools in the process.

Finally, Araújo Silva et. al. [26] conducted an SLR on Model-Driven Engineering (MDE) for systematizing the software development of robotics. This work aims to provide a comprehensive overview of existing model-based approaches in robotics. Their study highlighted that the self-adaptation of robots has been poorly explored and suggested using the advances in ML approaches for attaining greater autonomy of robotic software.

With the exception of the work by Araújo Silva et. al. [26], none of the works presented above mention ML or its potential application for solving MDE problems. Araújo Silva et. al. [26] mention that research in the field of ML might impact the intersection of MDE and robotics, but it does not study the application of ML in MDE. In contrast, ML is fundamental for this review where the goal consists of providing a comprehensive overview of ML approaches and their usage in MDE.

Since MDE is a software engineering methodology [27, 28], we considered related works that focus on the application of ML in Software Engineering. We found out that there are fewer works available, but none of them consider MDE the main artifact at play.

Subahi [29] places the focus on program synthesis, defined as a software development task that aims at achieving an automatic process of code generation that is satisfactory given high-level specifications. Subahi studied ML, Natural Language Processing (NLP), and Artificial Intelligence (AI) approaches, concluding that the rise in advanced ML techniques has been remarkable and arguing that there is a need to gain greater benefits from these approaches in order to cognify synthesis processes for building next-generation model-driven engineering (MDE) approaches.

Rigou et. al. [30] studied ML approaches that are able to draft a PIM model that describes the functional requirements of a system from a textual specification, which is a prerequisite in the context of MDE and which is used to automatically or semi-automatically derive the source code of a system. As a result of the paper, the authors outlined a deep learning approach to achieve the extraction of a PIM from the textual specification of a system.

Elmidaoui et. al. [31] reviewed software product maintainability prediction (SPMP) techniques for improving software maintainability. In their SLR, the authors analyzed and summarized the empirical evidence on the prediction accuracy of SPMP techniques, revealing that most studies were solution proposals using a history-based empirical evaluation approach and that the most used techniques were ML techniques. They concluded that developing more accurate techniques may facilitate their use in industry and they provided guidelines for improving the maintainability of software.

These reviews focus on the application of ML techniques for software engineering. In contrast, our review focuses on the application of ML techniques for a specific software engineering paradigm: MDE. Even though one of our research questions considers the software engineering tasks, MDE concepts are the primary artifacts of study in our review. In fact, our review provides results that would be hard to synthesize from processing other studies with orthogonal or adjacent scope, such as the ones previously described.

4 Research Method

The Systematic Literature Review (SLR) method describes the formal and reproducible steps to identify, evaluate, and interpret the scientific studies that are related to the desired subject [12]. Our review method was conducted using the best practices and guidelines for SLR research [12, 32, 33]. This section describes the review method and how it was followed to address the research questions raised.

4.1 Research questions

We formulate the goal of this research by using the Goal-Question-Metric perspectives (i.e., purpose, object, issue, viewpoint) [34]. The purpose is *identify, classify, and understand*, the object is *existing approaches that support MDE using ML techniques*, the issue is *for the publication of the state, limitations, and trends*, and the viewpoint is *from the researcher's viewpoint*. To achieve the goal of this work, the SLR aims to answer the following research questions (RQs):

RQ1: *What are the existing approaches that solve MDE problems using ML techniques?*

RQ1.1: What ML techniques are most commonly used by these approaches?

RQ1.2: What types of models do these approaches handle?

RQ1.3: What activities in the software development process are affected by these approaches?

RQ1 is motivated by the need to provide an overview of the approaches that solve MDE problems by applying ML techniques. RQ1 is detailed using three sub-questions that aim to know what the most common ML techniques are, what types of models are handled, and what software development activities are addressed by current approaches. The main goal of RQ1 is to identify trends in approaches to solve MDE problems by applying ML techniques. RQ1 also serves to identify open problems and possible improvements.

RQ2: *What is the current maturity level of approaches that use ML to solve MDE problems?*

RQ2 is motivated by the need to measure the level of maturity of current approaches that solve MDE problems by applying ML techniques. We defined the maturity of an approach as its level of development, acceptance, and usefulness in the scientific community. Maturity cannot be assessed by just one indicator, and therefore, we focus on four specific indicators: quality score, citation count, type of studies, and evaluation context. The quality score, the citation count, and the type of study are common indicators in reviews related to software engineering [35]–[37]. The evaluation context was included considering the importance of transferring research advancements to the industrial world. The main goal of RQ2 is to identify the extent to which the approaches could be used in practice and to learn about the maturity of studies published in the research area.

RQ3: *What are the limitations of the existing approaches?*

RQ3 is motivated by the interest in identifying future lines of work. The application of ML techniques to solve MDE problems is an area of research that has grown a lot in recent years. Therefore, researchers and practitioners would benefit greatly from an overview, especially for current problems and future research directions. The main objective of RQ3 is to identify trends in MDE approaches using ML techniques, open problems, and possible future research directions for improvement.

4.2 Search String

To collect all of the available published literature that is relevant for the research questions, a database search was adopted as search strategy. Specifically, database searches consist of conducting systematic searches in databases using well-defined search strings to find relevant literature.

In this work, the search string was extracted following the steps suggested by Kitchenham and Charters [12]. First, we used PICO (Population, Intervention, Comparison, and Outcomes) criteria to derive the major terms from the research questions.

- Population: In software engineering, population may refer to a specific software engineering role, category of software engineer, an application area, or an industry group [12]. In our work, the population consists of MDE studies.
- Intervention: In software engineering, intervention refers to a software methodology, tool, technology, or procedure that addresses a specific issue [12]. In our work, the intervention corresponds to ML techniques.
- Comparison: In software engineering, the comparison is the software engineering methodology, tool, technology, or procedure against which the intervention is being compared [12]. In our work, the comparison is not applied. The goal of this SLR is to characterize the MDE studies that apply ML techniques.
- Outcomes: In software engineering, the outcomes should relate to factors of importance to practitioners [12]. In our work, the outcomes are approaches, methods, tools, frameworks, processes, and guidelines presented by MDE studies that apply ML techniques.

Taking into account the research questions, the identified search terms were *Model Driven Engineering*, *MDE*, *Machine Learning*, and *Approach*. These terms were grouped into three sets:

- MDE terms: those search terms directly related to the population (i.e., *Model Driven Engineering* and *MDE*).
- ML terms: those search terms directly related to the intervention (i.e., *Machine Learning*).
- Approach terms: those search terms directly related to the outcomes (i.e., *Approach*).

Then, we found alternative spellings and/or synonyms. In the case of MDE terms, we included *model-driven* and *model driven*. In the case of ML terms, no alternative spellings or synonyms were included. In the case of approach terms, we included *method* and *process*.

Then, we verified the search terms in relevant works. The MDE terms were extended based on the search terms used in several works [3, 11, 38, 39]. However, we avoided including the terms *model* and *model-based*, because these two terms are fundamental and frequently used in both research areas: MDE and ML. However, the meaning of these terms is different depending on the area. Therefore, when considering any of these terms as a search term for MDE [3, 38], a study would only need to include the term *model* or the term *model-based* and one of the terms for ML to be selected. Since most of the studies related to ML contain either the term *model* or the term *model-based*, we would obtain a lot of studies that are only related to ML with no relation to MDE. In fact, using the term *model* as part of the search string in Scopus, 594,166 studies were found. In contrast, 6,255 studies were found without this term. Similarly, using the term *model-based* as part of the search string in Scopus, 47,027 studies were found. In contrast, 6,255 studies were found without this term. Therefore, to avoid misunderstandings and to simplify the search, the terms *model* and *model-based* were not included among the MDE terms.

In the case of ML terms, the search terms were extended based on the search terms used in a different set of papers [40, 41]. Both Ali et. al. [40] and Wen et. al. [41] propose a wide range of terms that are not focused only on ML but also on other related topics such as artificial intelligence and data mining. In artificial intelligence, ML is the sub-branch that allows the system to learn without being explicitly programmed [42, 43]. On the other hand, data mining combines statistics with ideas, tools, and methods from computer science, ML, database technology, and other classical data analytical technologies to discover interesting, unexpected, or valuable structures in large datasets [44]. Although ML is closely related to artificial intelligence and data mining, the terms *artificial intelligence* or *data mining* cannot be considered synonyms of *Machine Learning* nor valid terms within the ML sub-branch. For this reason, these terms were not included among the ML terms. We also discarded the terms *genetic algorithms* and *genetic programming* because genetic algorithms and genetic programming are artificial intelligence algorithms, but they are not considered ML techniques. Most recent taxonomies do not include these terms as ML techniques [45]–[48]. In contrast, we included the names of the ML techniques used as search terms in the aforementioned papers [40, 41], hence including terms such as decision tree or neural net.

With regard to approach terms, the search terms were extended based on the search terms used in [49]. However, we avoided including the terms *technique* and *model* because these two terms are frequently used in both ML and MDE.

Table 1 shows the search terms for each one of the sets: MDE terms, ML terms, and approach terms. From this table, we used boolean operators to construct the search string. Specifically, all MDE terms were combined by using the boolean 'OR' operator; all ML terms were combined by using the boolean 'OR' operator; and all approach terms were combined by using the boolean 'OR' operator. Then, we combined the MDE terms, the ML terms, and the approach iterms by using the Boolean 'AND' operator, which implies that, for a study to appear on our search, it needs to include one of the terms for MDE, one of the terms for ML, and one of the terms for approach. The search string is reported below:

(('model-driven engineering' OR 'MDE' OR 'model-driven' OR 'model driven' OR 'MDD' OR 'MDA' OR 'model-driven architecture' OR 'model-driven development' OR 'UML' OR 'DSL' OR 'DSML') AND ('Machine learning' OR 'decision tree' OR 'regression tree' OR 'classification tree' OR 'nearest neighbo*' OR 'neural net*' OR 'bayesian belief network' OR 'bayesian net*' OR 'association rule*' OR 'support vector machine' OR 'support vector regression' OR 'support vector*') AND ('approach' OR 'method' OR 'tool' OR 'framework' OR 'process' OR 'guidelines'))*

4.3 Selection Strategy

To search all of the available published studies that are relevant to our research questions, we followed the study selection process depicted in Fig. 1. Specifically, the selection strategy was composed of four steps: (1) Initial search, (2) Exclusion criteria, (3) Inclusion criteria, and (4) Snowballing.

4.3.1 Initial Search

The search string was used to collect the studies that are present in multiple sources. Specifically, the sources considered were IEEE, ACM Digital Library, Scopus, and Web of Science. These databases were selected based on the experience reported by Dyba et al. [50] and Kitchenham and Brereton [32]. Specifically, these works reported that the use of IEEE and ACM as well as two indexing databases is sufficient [49].

This search covers from any time in the past through December 31, 2023. Specifically, we found 6,255 studies using the search string. Table 2 shows the total number of studies found in the search per database and the number of studies selected at the end of the selection process.

Note that this table does not reflect the duplicity of papers: a paper can be found in several sources, or even in all four sources. Due to the search process followed in this work, Scopus papers were selected first, discarding duplicate papers from the other sources. The search then continued by orderly selecting Web of Science papers, IEEE papers, and ACM papers. With regard to the duplicity of papers, 83% of the 78 selected studies can be found in Scopus, 46% of the 71 selected studies can be found in Web of Science, 40% of the 71 selected studies can be found in IEEE, and 4% of the 71 selected studies can be found in ACM. Hence, although most of the selected studies were found in Scopus, it is also possible to find them in other sources.

4.3.2 Exclusion Criteria

After the search, the retrieved studies were evaluated based on a series of exclusion criteria. The studies that met one of the exclusion criteria were discarded for our review. The following criteria state the conditions for the exclusion of a study:

- studies that are duplicates of other studies.
- studies presenting summaries of conferences or editorials.
- studies that were not written in English.
- studies presenting peer-reviewed research that were not published in journals, conferences, or workshops (e.g., PhD theses or review studies).
- studies presenting non-peer-reviewed material.

The entire list of retrieved studies was filtered to exclude non-relevant studies. This step was conducted by one author of this paper, who applied the exclusion criteria taking into account the title, abstract, and keywords of the retrieved studies. As a result, 2,700 studies were discarded to satisfy the exclusion criteria. The remaining studies, 3,555 studies were selected for the following step.

TABLE 1: Search terms

MDE terms	ML terms	Approach terms
Model-driven engineering, MDE, model-driven, model driven, MDD, MDA, model-driven architecture, model-driven development, UML, DSL, DSML	Machine learning, decision tree, regression tree, classification tree, nearest neighbo*, neural net*, bayesian belief network, bayesian net*, association rule*, support vector machine, support vector regression, support vector*	Approach, method, process, tool, framework, guidelines

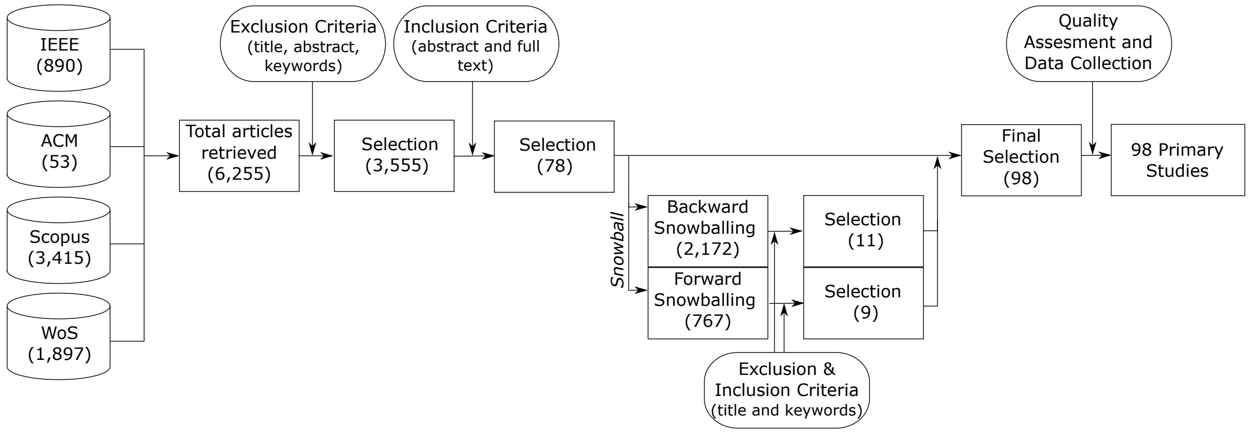

IEEE (890)
ACM (53)
Scopus (3,415)
WoS (1,897)
Total articles retrieved (6,255)
Exclusion Criteria (title, abstract, keywords)
Selection (3,555)
Inclusion Criteria (abstract and full text)
Selection (78)
Snowball
Backward Snowballing (2,172)
Forward Snowballing (767)
Selection (11)
Selection (9)
Exclusion & Inclusion Criteria (title and keywords)
Final Selection (98)
Quality Assesment and Data Collection
98 Primary Studies

Fig. 1: Overview of the search process

TABLE 2: Comparison between the total studies found concerning each source (i.e., IEEE, ACM, Scopus, and Web of Science) or using snowballing and the number of studies selected at the end of the search process

Source	# Total studies found	# Primary studies
IEEE	890	8
ACM	53	0
Scopus	3,415	68
Web of Science	1,897	2
Backward Snowballing	2,172	11
Forward Snowballing	767	9

4.3.3 Inclusion Criteria

After the exclusion criteria, the inclusion criteria were applied to the remaining studies. We define a set of inclusion criteria to be taken into account for the three ways that ML and MDE can be combined.

First, MDE artifacts (e.g., models or metamodels) are used to solve or improve the solution to a challenge stemming from ML (MDE_in_ML). For instance, the work in [51] analyzes the architecture of ML software systems. To do this, their authors apply reverse engineering to build the UML class diagram of the systems, and then use these diagrams to study the characteristics of the architectures. Second, ML techniques are applied to solve MDE problems (ML_in_MDE). For example, the work in [SLR17] applies a long short-term memory neural network to automatically infer model transformations from sets of input-output model pairs. Third, MDE artifacts and ML techniques are used to solve a research problem (MDE_and_ML). For example, the work in [52] introduces the notion of context into rules and decision trees that are used inside conceptual models allowing to incorporate context as important information for personalized web applications. The goal of this work is to provide travelers on public transport with personalized information at the right time. The solution proposed combines conceptual models (MDE) and rules and decision trees (ML).

Since this work focuses on the second perspective (ML_in_MDE), the inclusion criteria are defined to select the studies that solve MDE problems using ML techniques. The following criteria state the conditions for the inclusion of a study:

- studies that were in the field of computer science.
- studies that were focused on software models that describe how to develop software systems.
- studies that were focused on applying ML techniques to solve MDE problems.

The studies that met all of the inclusion criteria are the ones selected for our review (primary studies). This step was conducted by two authors of this paper, who applied the inclusion criteria taking into account the abstract and full-text of the selected studies. The 3,555 studies that were selected in the previous step were filtered to include only the relevant studies to answer the research questions. As a result, 78 studies were included to satisfy the inclusion criteria, which is about 1.25% of the papers found in the initial search.

4.3.4 Snowballing

To search for possible missing papers, we applied snowballing to identify additional sources [53]. Specifically, from the 78 selected studies, we performed as many backward and forward snowballing iterations as necessary until no

TABLE 3: Questions for Assessing Study Quality

#	Question
Q1	Is there a rationale provided for why the study was undertaken?
Q2	Is there an adequate description of the context (industry, laboratory settings, product used, etc.) in which the research was carried out?
Q3	Is there a justification and description for the research design?
Q4	Is there a clear statement of findings, including data that supports findings?
Q5	Did the researcher(s) critically examine his/her (their) own role, potential bias, and influence during the study?
Q6	Are the limitations and credibility of the study discussed explicitly?

new studies were found. For each iteration, two authors reapplied the exclusion/inclusion criteria based on the title of all of the studies. As a result, 11 studies were found using backward snowballing, and 9 studies were found using forward snowballing (see Table 2). In total, 20 studies were found for further analysis.

To ensure accuracy, the selection process was double-checked by another author of this paper. In case of disagreement, the first and second authors held a discussion in order to reach a consensus. Table 2 shows the number of primary studies selected through snowballing.

At the end of the search process, 98 papers (78 papers from the initial search and 20 studies from snowballing) make up the primary studies of this work. Therefore, there are a total of 98 primary studies in this review, where 79.6% of the studies were found in databases and 20.4% of the studies were found by applying snowballing.

4.4 Quality Assessment

This section describes the quality assessment, which is critical for SLRs [12]. Since we expected to find both empirical and non-empirical studies, using a study design hierarchy (e.g., experimental studies, case control studies) would not apply to all studies. Therefore, instead of using the study design hierarchy for software engineering proposed in [54], the papers were assessed using questions as proposed in [35]. Specifically, each paper was assessed using the six questions of Table 3, which placed the focus on the overall quality of a paper.

Similarly to Galster et al. [35] and the three point scale instrument described in [55], each question was answered by assigning responses on a three point scale: *yes*, *to some extent*, or *no*. Each point in the scale is replaced by a numerical value in order to avoid neglecting the papers with limited information [35]: *yes* is equal to 1, *to some extent* is equal to 0.5, and *no* is equal to 0. The quality score of a study is calculated as the sum of the scores that the study has obtained for each question. The quality scores are used for quality assessment as an indicator of the maturity level of the approaches. It was not used for filtering purposes, so none of the primary studies were included or excluded due to their quality score.

4.5 Data Collection

Table 4 shows the information collected for each study. To collect this data, two researchers read the full papers for the selected studies in parallel. The data was extracted by the one of the researchers and checked against the paper by the other researcher. In the case of disagreement, the researchers discussed the data or consulted an additional researcher.

TABLE 4: Data Collection Form

#	Field	Research question
F1	Author	n/a
F2	Year	n/a
F3	Title	n/a
F4	Venue	n/a
F5	Keywords	n/a
F6	Abstract	n/a
F7	ML technique	RQ1.1
F8	Type of Models	RQ1.2
F9	Type of UML diagrams	RQ1.2
F10	MDE problem	RQ1.3
F11	Transformation type	RQ1.3
F12	Activities addressed	RQ1.3
F13	Quality score	RQ2
F14	Citation count	RQ2
F15	Type of studies	RQ2
F16	Evaluation context	RQ2
F17	Level of tool support	RQ3
F18	Limitations	RQ3

The data collection was divided into four parts: Demographic information (F1 to F6), Data for RQ1 (F7 to F12), Data for RQ2 (F13 to F16), and Data for RQ3 (F17 to F18). The following subsections describe the fields for each of these parts.

4.5.1 Demographic information

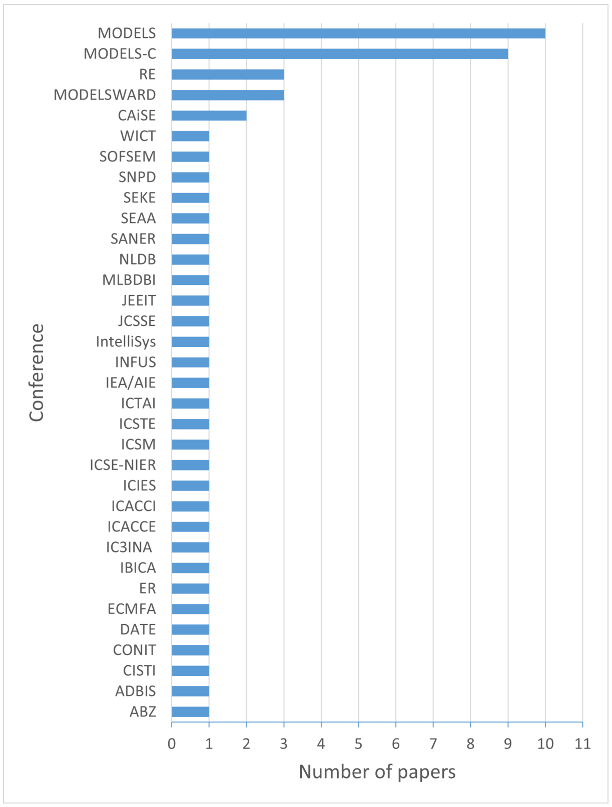
F1 to F6 record the meta-information of papers, such as the names of the authors, the publication year, the title of the paper, the venue (i.e., workshop, conference, or journal), the keywords, and the abstract. This information is collected for both documentation purposes and for the demographic study.

Taking into account this information, 14.61% of the primary studies are published in workshops, 56.18% of the primary studies are published in conferences, and 29.21% of the primary studies are published in journals.

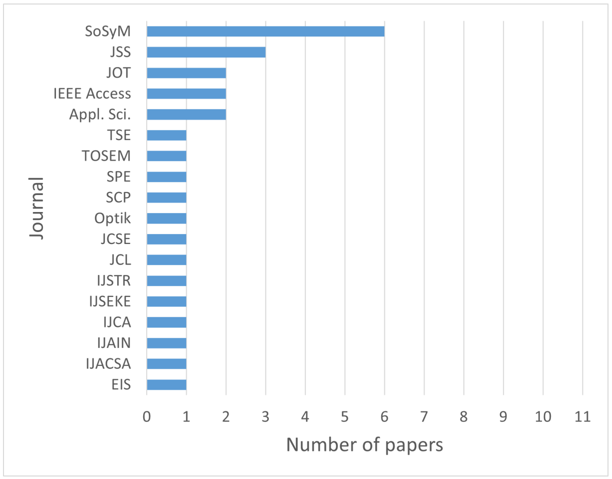
Fig. 2a and Fig. 2b show the distribution of primary studies concerning conferences and journals, respectively. The conferences and journals with the highest number of primary studies are, in fact, relevant publication venues for the MDE community. *International Conference on Model-Driven Engineering Languages and Systems* (MODELS) and *International Conference on Model-Driven Engineering Languages and Systems Companion* (MODELS-C) have been the two most prominent venues for publishing ML-based papers for MDE, with MODELS being the main conference and MODELS-C being a collection of satellite events that accompany the conferences (mainly, associated workshops). *Software and Systems Modeling* (SoSyM) is the most prominent journal for publishing ML-based papers for MDE.

Moreover, Fig. 3 shows the distribution of the primary studies regarding type and year of publication. This figure shows an increasing trend for publishing ML-based papers in MDE. While the number of studies published in work-

Fig. 2: Distribution of primary studies for venues



(a) Distribution of primary studies for conferences



(b) Distribution of primary studies for journals

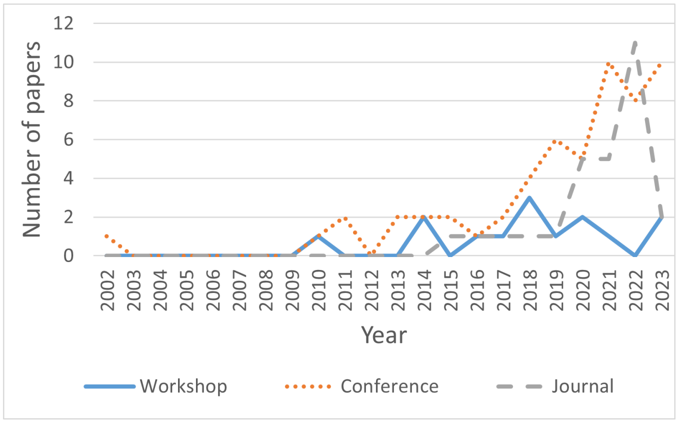


Fig. 3: Distribution of primary studies for type and year of publication

shops remained stable until 2022, the number of papers published in conferences consistently increased from 2013 onwards. In the case of journals, the number of publications has consistently been lower than the number of papers published in conferences. However, an increase in the number of papers published in journals from 2019 onwards can be observed with journals surpassing conferences for the first time in 2022.

4.5.2 Data for RQ1

F7 to F12 extract the data to answer RQ1. Each of these fields and their categories are detailed below.

F7 records the type of ML technique applied on MDE. We used the terms that were identified during the search process (see Table 1). This field was used to answer RQ1.1.

F8 reports the type of models in use. If the type of model is Unified Modeling Language (UML), F9 documents the types of UML diagrams used in the paper. While the categories for F8 are identified using natural language, the categories for F9 are the ones defined by the Object Management Group (OMG) in the last UML specification (Version 2.5.1) [56]. These fields were used to answer RQ1.2.

F10 documents the MDE issue addressed by the method. In the cases where the MDE problem deals with model transformations, F11 captures the specific transformation (e.g., from code to model). As for F12, it logs the activities targeted by the approach, tool, or methodology. While F10 and F11 were recorded using natural language, F12 is based on the following activities proposed by Galster et al. [35]: requirements engineering, architecture/design, implementation, testing and verification, and maintenance. These fields were used to answer RQ1.3.

4.5.3 Data for RQ2

F13 to F16 extract the data to answer RQ2. Specifically, the answer to this question is based on four metrics: the quality score, the citation count, the type of study, and the evaluation context.

F13 records the quality score. This quality score is measured using the quality assessment described in Section 4.4. F14 records the citation count based on Google Scholar (as of April 2024). F15 records whether a study is empirical or not.

F16 records the evaluation context, which is used to study the maturity of the evaluation performed by each primary study. We adapted the revised classification that is described in [35], which is based on the levels of study design proposed by [54]. The classification has five levels: no evaluation (0), evaluation based on a demonstration or toy examples (1), evaluation based on expert opinions or observations (2), evaluation based on academic cases (3), evaluation based on industrial cases (4), and evaluation from industrial practices (5). The weakest level is 0 where the paper does not present any evaluation. The strongest level is 5 where the approach, tool, or method is approved and adopted by industrial organizations [57].

4.5.4 Data for RQ3

F17 to F18 extract the data to answer RQ3. Specifically, F17 records the level of tool support, considering that the paper can present an *automatic* tool, a *semi-automatic* tool, a *manual* tool, or *no tool* at all. F18 records the limitations of the works as text.

5 Results

This section presents the final results of the SLR. The results of the whole search process, step by step, are available at https://doi.org/10.5281/zenodo.10678163. Specifically, this website contains the results of the selection process as well as the results of the quality assessment and data collection.

5.1 RQ1. What are the existing approaches that solve MDE problems using ML techniques?

In this section, we outline the results of the studies on solving MDE problems using ML. The studies are listed in the Systematic Review References section that comes after the bibliography of this work.

5.1.1 RQ1.1. What ML techniques are most commonly used by these approaches?

The answer to this question is based on F13 data from the data extraction form regarding ML techniques in use. Table 5 shows the number of papers that apply each ML technique. It also includes the references of the papers.

Overall, neural networks and decision, regression, and classification trees are the most commonly used techniques in MDE. Then, we find bayesian networks, followed by clustering and nearest neighbors. The rest of the ML techniques (e.g., association rules or support vector machine (SVM)) are less used to solve MDE problems.

Fig. 4 shows the distribution of the most used ML techniques applied on MDE for years. Note that, neural networks, which are the most widely used ML technique, were practically not applied until 2020. However, there was a pronounced increasing interest in 2021. In contrast, the other four most applied techniques (i.e., regression and classification trees, bayesian networks, clustering, and nearest neighbors) have a similar behavior. They were first used between 2009 and 2012 and their application remains more or less uniform. Notably, the recent growth of neural networks in 2021 coincides with a decline in the application of the other four techniques. Despite the continuous growth of neural networks in 2022, these techniques saw a recovery to their previous levels in 2022. However, this recovery does not appear to be stable given the 2023 results.

Furthermore, more than 14% of the studies use more than one ML technique. Generally, studies that apply multiple ML techniques compare those techniques to determine which one is best for the proposed approach of a study [SLR64, SLR67, SLR14, SLR23, SLR38, SLR66, SLR34, SLR77, SLR54, SLR12, SLR41, SLR40, SLR48]. However, one of the primary studies uses several ML techniques to address

TABLE 5: ML techniques applied by studies

Techniques	#	Papers
Neural Networks	40	[SLR3], [SLR12], [SLR13], [SLR17], [SLR18], [SLR23], [SLR33], [SLR37], [SLR38], [SLR41], [SLR50], [SLR51], [SLR62], [SLR63], [SLR64], [SLR72], [SLR76], [SLR81], [SLR85], [SLR84], [SLR86], [SLR89], [SLR95], [SLR96], [SLR49], [SLR24], [SLR61], [SLR36], [SLR19], [SLR22], [SLR70], [SLR43], [SLR92], [SLR40], [SLR52], [SLR47], [SLR80], [SLR79], [SLR16], [SLR91]
Decision, regression, and classification tree	24	[SLR12], [SLR14], [SLR15], [SLR23], [SLR29], [SLR31], [SLR32], [SLR38], [SLR39], [SLR41], [SLR44], [SLR46], [SLR45], [SLR53], [SLR64], [SLR66], [SLR65], [SLR67], [SLR77], [SLR88], [SLR97], [SLR98], [SLR40], [SLR54]
Bayesian Networks	13	[SLR14], [SLR34], [SLR35], [SLR38], [SLR41], [SLR64], [SLR66], [SLR67], [SLR77], [SLR90], [SLR94], [SLR40], [SLR54]
Clustering	12	[SLR7], [SLR6], [SLR21], [SLR27], [SLR42], [SLR73], [SLR83], [SLR48], [SLR8], [SLR5], [SLR4], [SLR20]
Nearest Neighbors	11	[SLR12], [SLR14], [SLR34], [SLR38], [SLR41], [SLR59], [SLR64], [SLR66], [SLR67], [SLR77], [SLR40]
SVM	6	[SLR23], [SLR38], [SLR41], [SLR67], [SLR48], [SLR40]
Association rules	2	[SLR26], [SLR32]
Others	16	[SLR1], [SLR30], [SLR58], [SLR60], [SLR69], [SLR2], [SLR93], [SLR25], [SLR11], [SLR54], [SLR10], [SLR28], [SLR82], [SLR55], [SLR56], [SLR57]
Not specified	7	[SLR9], [SLR68], [SLR71], [SLR74], [SLR75], [SLR78], [SLR87]

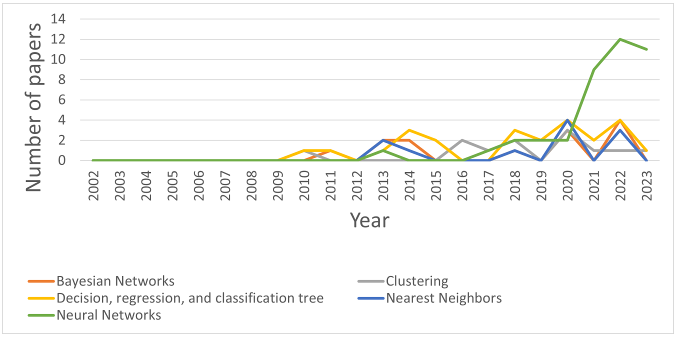


Fig. 4: Most used ML techniques in MDE per year

different parts of the proposed approach. In [SLR32], the proposed approach uses association rules for obtaining features that are then used by a decision tree algorithm to classify classes, whether it predicts the proneness to change or not.

5.1.2 RQ1.2. What types of models do these approaches handle?

To answer this question, we drew on data that was extracted based on F9 (type of models) and F10 (type of UML diagrams) from the data extraction form. Table 6 groups and counts the papers by type of model.

Most of the primary studies apply ML techniques on Unified Modeling Language (UML) diagrams. Specifically, more than 61% of the primary studies applied ML techniques on UML (i.e., 60 papers). To a lesser extent, ML techniques are also applied on other types of models. Mod-

TABLE 6: Type of models by studies: Unified Modeling Language diagrams (UML), Component State Transition Diagram (CSTD), models based on Domain Specific Language (DSL), etc.

Type of model	#	Papers
UML	60	[SLR3], [SLR1], [SLR12], [SLR13], [SLR17], [SLR18], [SLR26], [SLR29], [SLR31], [SLR32], [SLR33], [SLR34], [SLR35], [SLR37], [SLR38], [SLR39], [SLR42], [SLR46], [SLR53], [SLR59], [SLR60], [SLR64], [SLR66], [SLR65], [SLR67], [SLR68], [SLR69], [SLR71], [SLR72], [SLR75], [SLR81], [SLR83], [SLR85], [SLR84], [SLR86], [SLR88], [SLR89], [SLR90], [SLR94], [SLR95], [SLR96], [SLR24], [SLR93], [SLR36], [SLR19], [SLR22], [SLR25], [SLR43], [SLR92], [SLR8], [SLR5], [SLR11], [SLR54], [SLR10], [SLR20], [SLR82], [SLR80], [SLR79], [SLR16], [SLR91]
DSL	13	[SLR58], [SLR74], [SLR97], [SLR98], [SLR48], [SLR25], [SLR70], [SLR5], [SLR47], [SLR28], [SLR55], [SLR56], [SLR57]
Domain models	11	[SLR6], [SLR9], [SLR14], [SLR15], [SLR27], [SLR41], [SLR77], [SLR76], [SLR78], [SLR4], [SLR40]
Metamodels	10	[SLR7], [SLR44], [SLR45], [SLR59], [SLR62], [SLR63], [SLR73], [SLR49], [SLR24], [SLR61]
CSTD	3	[SLR50], [SLR51], [SLR52]
Design models	2	[SLR2], [SLR54]
Relational schema	2	[SLR17], [SLR16]
Others	5	[SLR30], [SLR23], [SLR30], [SLR1], [SLR2]
Not specified	2	[SLR21], [SLR87]

els defined by Domain Specific Language (DSL) are used in 13.3% of the primary studies, domain models are used in 11.2% of the primary studies, metamodels are used in 10.2% of the primary studies, component state transition diagrams are used in 3.1% of the primary studies, design models are used in 2.0% of the papers, relational schemas are used in 2.0% of the papers. Other kinds of models, such as entity-relationship diagrams or conceptual models are used in less than 1.0% of the papers. We also found two works where the type of model is not specified [SLR21, SLR87]. The work in [SLR21] proposes the use of graph kernels as a general framework for approaching problems in Model-Driven Engineering. However, this framework will be evaluated with different graph kernel and model types in the future. The work in [SLR87] presents ideas on taking the first steps towards cultivating synergy between MDE, ML, and software clones. Therefore, none of these two works focus on one specific type of model.

Since most of the primary studies focus on UML diagrams, we also collected the type of UML diagram in use. To do this, we used the taxonomy of structure and behavior diagrams defined by the OMG in the last UML specification (Version 2.5.1) [56]. According to this taxonomy, there are two major kinds of diagram types: structure diagrams and behavior diagrams. In Fig. 5, the structure diagrams are pointed out with striped bars: profile diagrams, class diagrams, composite structure diagrams, component diagrams, deployment diagrams, object diagrams, and package diagrams. The behavior diagrams are pointed out with dotted bars: activity diagrams, interaction diagrams, use case diagrams, and state machine diagrams. Fig. 5 shows the number of papers for each type of diagram. Most of

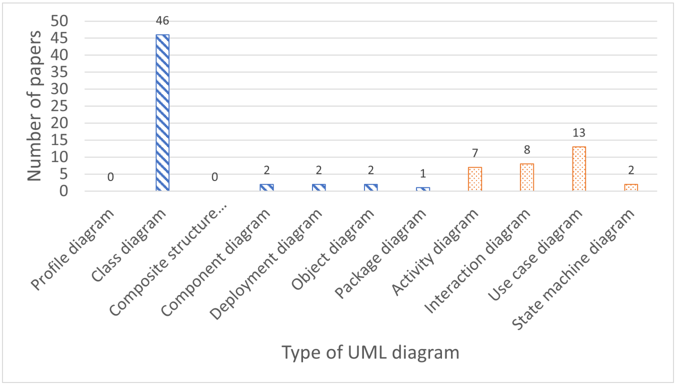


Fig. 5: Distribution of the primary studies on the type of UML diagram

the primary studies applied ML techniques on structure diagrams, and more specifically, on class diagrams.

Note that, of the 60 primary studies that apply ML techniques in UML diagrams, 46 primary studies focus on class diagrams, i.e., 54.1% of the primary studies applying ML techniques in MDE focus on class diagrams. However, some of these primary studies do not focus only on class diagrams but support other UML diagrams as well (usually sequence diagrams). It is also remarkable that some primary studies cite the use of a single type of model as a limitation of their work or propose to extend their study to other types of models as future work [SLR24, SLR60, SLR26, SLR7].

As we commented in the background section, models are based on the shape of the model elements or the type of model elements. Although both methods are used independently of the type of model, shape is always used in research dealing with images [SLR33, SLR60, SLR89, SLR84, SLR85, SLR19, SLR43, SLR92]. These works focus on the recognition of diagrams from images or on the classification of images by model type. Moreover, all of these works use UML diagrams.

5.1.3 *RQ1.3. What activities in the software development process are affected by these approaches?*

To answer this question, we drew on data extracted based on F11 (MDE problem), F12 (transformation type), and F16 (activities addressed), with the aim of identifying the software development activities and MDE problems addressed by the approaches, enabling the identification of future lines of research as well as the application of the approaches to other software development activities or MDE problems.

TABLE 7: Software engineering (SE) activities addressed by studies

SE activities	#	Papers
Requirements engineering	20	[SLR26], [SLR29], [SLR37], [SLR42], [SLR44], [SLR46], [SLR45], [SLR50], [SLR51], [SLR58], [SLR68], [SLR69], [SLR72], [SLR90], [SLR94], [SLR96], [SLR2], [SLR78], [SLR36], [SLR52]
Architect./design	4	[SLR18], [SLR63], [SLR89], [SLR95]
Implementation	2	[SLR32], [SLR20]
Testing	3	[SLR21], [SLR34], [SLR74]
Maintenance	6	[SLR31], [SLR53], [SLR86], [SLR93], [SLR4], [SLR40]

Table 7 shows the number of papers per addressed software engineering activity. This table only categorizes the primary studies that focus on a specific SE activity. However, only 39.3% of the primary studies focus on a specific activity, while 64.3% of the primary studies are classified as *General*.

On the other hand, we use F11 (MDE problem) to identify MDE problems addressed by the primary studies of this review. The following are the main MDE problems tackled by them:

- Specification of models, which is focused on the design of models.
- Model analysis, which is focused on evaluating models to determine whether a model is valid, detecting defects on models, or grading models according to specific quality criteria.
- Model transformation, which is focused on converting models between different abstraction levels (e.g., model-to-model or model-to-code).
- Model-driven reverse engineering, which is focused on generating relevant model-based artifacts from legacy systems.
- Classification of models, which is focused on the categorization of models.
- Model comparison, which is focused on identifying similarities and differences between models.
- Identification of model elements, which is focused on finding specific elements in a model.
- Model recognition, which is focused on identifying models in different sources (e.g., images).
- Classification of metamodels, which is focused on the categorization of metamodels.

Table 8 shows the distribution of the primary studies over the identified MDE problems. Model specification is the most common MDE problem. Nevertheless, there is a considerable number of studies focusing on model analysis, classification of models, and model transformation. In addition, considering the type of model transformation, most of the primary studies, that focus on model transformation deal with model-to-model transformation [SLR1, SLR17, SLR27, SLR30, SLR44, SLR45, SLR2, SLR28]. However, Burgueño et. al address both the transformation from model to model and the transformation from model to code [SLR16].

When comparing these MDE problems over the years, we observed three waves of emergence. In the first wave, between 2010 and 2012, the first papers applying ML techniques for model transformations, model-driven reverse engineering, model analysis, and specification of models were published. In the second wave, between 2016 and 2017, the first papers applying ML techniques for model comparison, classification of models, and identification of model elements were published. In the third wave, between 2019 and 2020, the first studies applying ML techniques for classification of metamodels and model recognition were published.

Fig. 6 shows the distribution of the most common MDE problems concerning the three waves described. Fig. 6a shows that there is a higher number of publications on the MDE problems belonging to the first wave compared to the two later waves, with one exception classification of models (Fig. 6b). The classification of models is a special case. The first primary study that addressed this problem using ML dates back to 2002. However, it was not until 2017 that this problem was addressed again using ML. Furthermore, although it has only been six years since this problem resurfaced, there are already more primary studies on model classification using ML than primary studies addressing model transformations or model-driven engineering, both of which emerged in the first wave.

TABLE 8: MDE problems addressed by studies

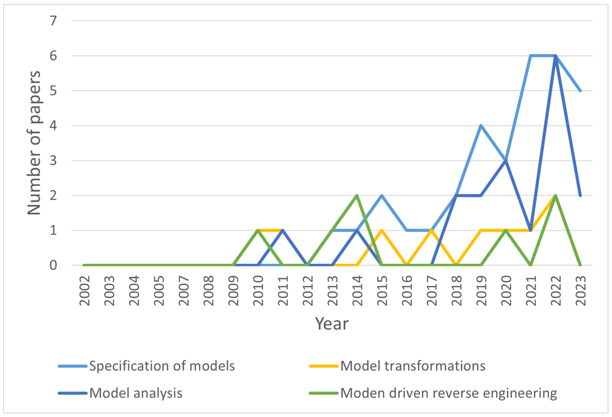
MDE problem	#	Papers
Specification of models	32	[SLR26], [SLR29], [SLR31], [SLR32], [SLR34], [SLR37], [SLR42], [SLR46], [SLR50], [SLR68], [SLR69], [SLR71], [SLR72], [SLR74], [SLR75], [SLR77], [SLR76], [SLR81], [SLR83], [SLR90], [SLR94], [SLR96], [SLR97], [SLR98], [SLR24], [SLR78], [SLR25], [SLR70], [SLR87], [SLR47], [SLR80], [SLR79]
Model analysis	18	[SLR6], [SLR9], [SLR12], [SLR14], [SLR15], [SLR18], [SLR23], [SLR35], [SLR38], [SLR53], [SLR86], [SLR88], [SLR22], [SLR11], [SLR54], [SLR10], [SLR82], [SLR57]
Model transformations	9	[SLR1], [SLR17], [SLR27], [SLR30], [SLR44], [SLR45], [SLR2], [SLR28], [SLR16]
Moden driven reverse engineering	7	[SLR39], [SLR64], [SLR66], [SLR65], [SLR95], [SLR55], [SLR56]
Classification of models	14	[SLR3], [SLR13], [SLR21], [SLR41], [SLR67], [SLR85], [SLR84], [SLR89], [SLR49], [SLR93], [SLR5], [SLR40], [SLR20], [SLR91]
Model comparison	5	[SLR7], [SLR21], [SLR59], [SLR8], [SLR4]
Identification of model elements	4	[SLR51], [SLR58], [SLR36], [SLR52]
Model recognition	7	[SLR33], [SLR60], [SLR85], [SLR84], [SLR19], [SLR43], [SLR92]
Classification of metamodels	5	[SLR62], [SLR63], [SLR73], [SLR61], [SLR48]

In addition, there is also some relationship between the emergence of the MDE problems and the ML techniques used to solve them. To address MDE problems that arose in the first wave (i.e., model transformations, model-driven reverse engineering, model analysis, and specification of models), the main techniques used were decision, regression, and classification tree, bayesian networks, and nearest neighbors. However, when the next wave emerged, these same problems began to be addressed using mostly neural networks. Similarly, to address the MDE problems that arose in the second wave (i.e., model comparison, classification of models, and identification of model elements) mainly clustering was used, but when the next wave arose, they started to be addressed using mainly neural networks. In contrast, the MDE problems that emerged in the third wave (i.e., classification of metamodels and model recognition) have been addressed from the beginning and almost exclusively by neural networks.

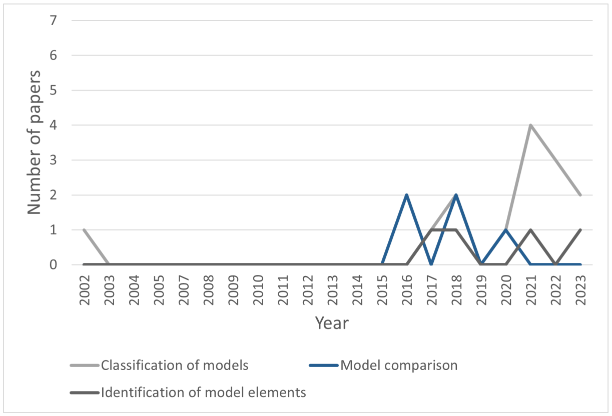
5.2 RQ2. What is the current maturity level of approaches that use ML to solve MDE problems?

In this section, we study the maturity of the approaches that apply ML techniques in MDE based on four metrics: the quality score (F13), the citation count (F14), the type of study (F15), and the evaluation context (F16). We also study the correlation between these metrics.

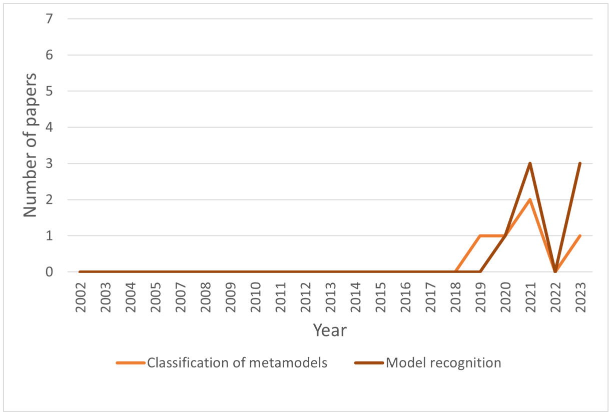
Fig. 6: Distribution of MDE problems over the years



(a) First wave of MDE problems using ML



(b) Second wave of MDE problems using ML



(c) Third wave of MDE problems using ML

5.2.1 *Maturity metrics*

With regard to quality scores, Fig. 7a shows the distribution of the primary studies for the total quality scores. The total quality score of each study is calculated as the sum of its scores for each quality question. A study achieves the maximum total quality score (equal to 6) when it reaches a 1 in the six quality questions. Most studies received a score between 3.5 and 6. The mean quality score of all papers is 4.33, so, although not perfect, this finding indicates that the authors strive to present research with some rigor. In Table 9, we list all of the studies with the highest total quality scores.

Fig. 7a also shows the total quality scores based on the type of publication venue. The number of papers published in workshops is outlined using solid bars, the number of papers published in conferences is outlined using dotted bars, and the number of papers published in journals is outlined using striped bars. The mean of the total quality score is 3.50 for studies published in workshops, 3.65 for studies published in conferences, and 4.70 for studies published in journals. In Table 9, we also include the citation count, the type of study, and the evaluation context for the studies with the highest total quality scores. Most of the studies that achieve high total quality scores (5.5 and 6) are empirical papers whose evaluation is based on academic cases and are published in journals.

TABLE 9: Studies with the highest total quality score

Paper	Venue	Quality score	Citation Count	Type of Study	Evaluation context
[SLR14]	Conf.	6	6	Emp.	3
[SLR19]	Journal	6	1	Emp.	3
[SLR26]	Journal	6	12	Emp.	3
[SLR33]	Journal	6	10	Emp.	3
[SLR51]	Conf.	6	2	NonE.	3
[SLR62]	Journal	6	16	Emp.	3
[SLR63]	Journal	6	1	Emp.	3
[SLR65]	Work.	6	11	NonE.	3
[SLR97]	Conf.	6	13	Emp.	3
[SLR98]	Journal	6	3	Emp.	3

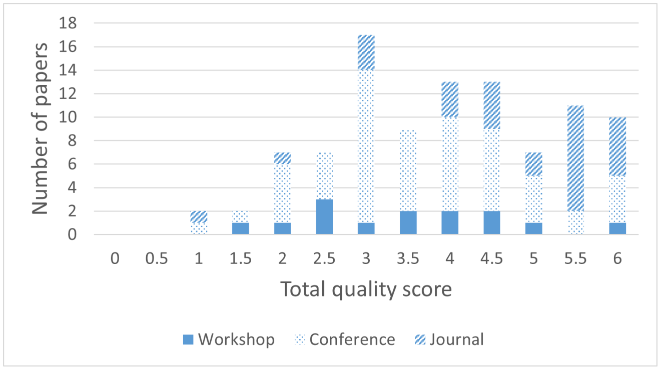
Venue: Work.-Workshop, Conf.-Conference, Journal-Journal
Type of Study: Emp.-Empirical, NonE.-Non-Empirical
Evaluation Context: 0-No evaluation, 1-Evaluation based on a demonstration or toy examples, 2-Evaluation based on expert opinions or observations, 3-Evaluation based on academic cases, 4-Evaluation based on industrial cases, 5-Evaluation from industrial practices.

With regard to citation counts, Fig. 7b shows the distribution of the primary studies for the number of citations. The paper with the highest number of citations has 90 citations. The mean citation count of all of the papers is 13.33.

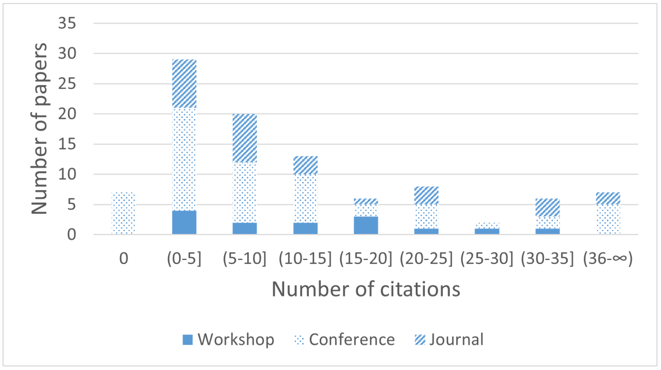
In Table 10, we list the studies with at least 16 citations. This threshold for the citation count is chosen to select the top 15% papers as in [35]. This table also includes the quality score, the type of study, and the evaluation context for the studies with the highest citation count. Most of the studies that achieve a high number of citations are evaluated using academic cases. There is no clear relation between the number of citations and the other two metrics (quality score and type of study).

Nevertheless, only 23.47% of the primary studies have more than 20 citations. Most of the studies have less than 10 citations. There are even seven papers that have no citations. Six of these seven papers are of recent publication (2022 onwards), while the remaining paper is a non-empirical study published at a conference in 2018 [SLR29]. This study presents a preliminary approach for a system in which the requirements taken as input are processed using advanced methods and decision trees to determine the requirements efficiently. The authors have not yet presented further work to continue this preliminary approach. We acknowledge that any direct conclusion regarding citation count may not be completely fair because the citation count of more recently

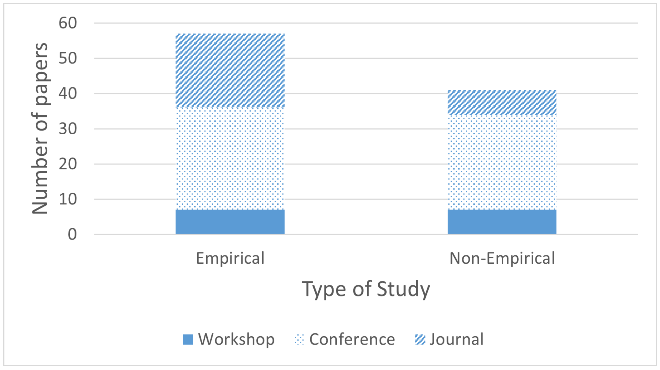
Fig. 7: Distribution of the primary studies over the four metrics and the venue.



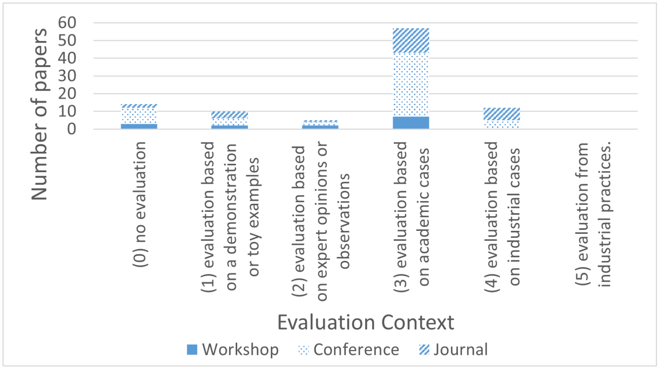
(a) Distribution of primary studies for Quality Score



(b) Distribution of primary studies for Citation Count



(c) Distribution of primary studies for Type of Study



(d) Distribution of primary studies for Evaluation Context

TABLE 10: Studies with the highest number of citations

Paper	Venue	Citation Count	Quality Score	Type of Study	Evaluation Context
[SLR12]	Journal	32	5.5	NonE.	2
[SLR17]	Conf	44	5	Emp.	3
[SLR21]	Work.	26	2.5	NonE.	0
[SLR27]	Work.	34	1.5	NonE.	1
[SLR33]	Journal	32	6	Emp.	3
[SLR39]	Conf	52	2.5	NonE.	3
[SLR62]	Journal	34	6	Emp.	3
[SLR64]	Conf	53	4.5	Emp.	3
[SLR69]	Journal	53	5.5	Emp.	4
[SLR86]	Journal	38	4.5	NonE.	1
[SLR24]	Conf	31	4	Emp.	3
[SLR61]	Conf	39	5	Emp.	3
[SLR93]	Conf	33	3	NonE.	3
[SLR5]	Conf	30	4	Emp.	3
[SLR54]	Conf	90	3.5	Emp.	3

Venue: Work.-Workshop, Conf.-Conference, Journal-Journal
Type of Study: Emp.-Empirical, NonE.-Non-Empirical
Evaluation Context: 0-No evaluation, 1-Evaluation based on a demonstration or toy examples, 2-Evaluation based on expert opinions or observations, 3-Evaluation based on academic cases, 4-Evaluation based on industrial cases, 5-Evaluation from industrial practices.

TABLE 11: Studies classified by type of study: Non-Empirical and Empirical

Non-Empirical	Emprirical
[SLR3], [SLR1], [SLR6], [SLR12], [SLR13], [SLR15], [SLR18], [SLR21], [SLR23], [SLR27], [SLR29], [SLR31], [SLR32], [SLR35], [SLR37], [SLR38], [SLR39], [SLR41], [SLR44], [SLR45], [SLR51], [SLR65], [SLR68], [SLR71], [SLR72], [SLR74], [SLR75], [SLR81], [SLR86], [SLR90], [SLR94], [SLR95], [SLR96], [SLR93], [SLR36], [SLR22], [SLR4], [SLR40], [SLR11], [SLR87], [SLR80]	[SLR7], [SLR9], [SLR8], [SLR5], [SLR2], [SLR14], [SLR17], [SLR26], [SLR30], [SLR33], [SLR34], [SLR42], [SLR46], [SLR50], [SLR53], [SLR58], [SLR59], [SLR60], [SLR62], [SLR63], [SLR64], [SLR66], [SLR67], [SLR69], [SLR73], [SLR77], [SLR76], [SLR83], [SLR85], [SLR84], [SLR88], [SLR89], [SLR97], [SLR98], [SLR49], [SLR24], [SLR78], [SLR61], [SLR19], [SLR48], [SLR25], [SLR70], [SLR43], [SLR92], [SLR52], [SLR54], [SLR10], [SLR20], [SLR47], [SLR28], [SLR82], [SLR79], [SLR16], [SLR91], [SLR55], [SLR56], [SLR57]

published papers will always be significantly lower than the citation count of older primary studies.

With regard to type of study, Fig. 7c shows the distribution of the primary studies for the type of study. Most of the primary studies are empirical studies. Specifically, there is a similar number of non-empirical and empirical studies published in workshops and conferences. In contrast, the number of empirical studies is somewhat greater than the number of non-empirical studies published in journals. In Table 11, we list the studies by type of study.

With regard to evaluation context, Fig. 7d shows the distribution of the primary studies for the evaluation contexts. Most of the studies have an evaluation that is based on academic cases. Moreover, 13.72% of the studies (14 papers) do not evaluate the work presented. All of these works are non-empirical studies that present preliminary ideas for a system, a method, or an approach. Their total quality

TABLE 12: Studies with the highest level of evaluation context (i.e., Evaluation based on industrial cases)

Paper	Venue	Evaluation context	Quality score	Citation Count	Type of Study
[SLR42]	Conf.	4	5	4	Emp.
[SLR46]	Conf.	4	4.5	0	Emp.
[SLR58]	Conf.	4	5.5	8	Emp.
[SLR69]	Journal	4	5.5	43	Emp.
[SLR96]	Conf.	4	3	1	NonE.
[SLR49]	Conf.	4	4.5	2	Emp.
[SLR25]	Journal	4	4	6	Emp.
[SLR47]	Journal	4	4.5	4	Emp.
[SLR16]	Journal	4	5	17	Emp.
[SLR55]	Journal	4	5.5	24	Emp.
[SLR56]	Journal	4	5.5	6	Emp.
[SLR57]	Journal	4	5.5	1	Emp.

Venue: Work.-Workshop, Conf.-Conference, Journal-Journal
Type of Study: Emp.-Empirical, NonE.-Non-Empirical
Evaluation Context: 0-No evaluation, 1-Evaluation based on a demonstration or toy examples, 2-Evaluation based on expert opinions or observations, 3-Evaluation based on academic cases, 4-Evaluation based on industrial cases, 5-Evaluation from industrial practices.

scores are around 2.5, and, surprisingly, most of them are published in conferences instead of workshops.

On the other hand, although 12.24% of the primary studies are evaluated using industrial cases, none of the primary studies evaluate the work directly in industry. Therefore, the presented approaches, methods, or tools are not approved and adopted by industrial organizations. Since evaluations based on industrial cases are the highest level of evaluation context, these studies are potentially the most relevant works from researchers. Therefore, we list these papers in Table 12 with their quality score, citation count, and type of study. All of these studies were published from 2016 to 2023, particularly in the last three years. Most of them are empirical studies that are published in both conferences and journals.

5.2.2 Correlation between metrics

In the previous section, we determined which primary studies achieve the best quality score (i.e., six points), obtain the highest number of citations (i.e., more than 20 citations), present a specific type of study (i.e., empirical or non-empirical), and evaluate the work based on the highest level of context (i.e., based on industrial cases). Next, we analyze whether the metrics are related to each other. For example, if the number of citations is high, the main study could have a high-quality score or empirical studies could achieve better quality scores than non-empirical studies.

To check the correlation among the four metrics, we applied different statistical analyses depending on the metrics and the normality of the data. It should be noted that all of the normality tests performed indicate that the data do not follow a normal distribution. Therefore, the methods applied were Spearman's rank correlation coefficient (when the two variables do not follow a normal distribution, such as the quality score and the citation count), the Mann-Whitney U test (when the two variables do not follow a normal distribution and one of this variables has two groups, such as the type of study), and Kruskal Wallis (when the two variables do not follow a normal distribution and one of the variables has more than two groups, such as the evaluation context).

The citation count is not related to the other metrics. When analyzing the correlation between citation count and quality score, Spearman's rank correlation coefficient (Spearman's ρ) is equal to 0.25. That means that there is no correlation between the citation count and the quality score, or that this correlation is weak. Similarly, there are no significant differences between the citation count for empirical and non-empirical studies, and there are no significant differences between the citation count for the different evaluation contexts.

In contrast, the quality score is related to the other two metrics: the type of study and the evaluation context. Therefore, we performed a deeper analysis of these metrics. With regard to the type of study, the Mann-Whitney U test does not show a significant difference between empirical and non-empirical studies for two of the questions (i.e., Q1 and Q3). Therefore, there are no differences between the rationale for why the work was undertaken (Q1) in empirical and non-empirical primary studies. There are no significant differences between the way to report the research design (Q3) in empirical and non-empirical primary studies.

However, there are significant differences between the quality scores of empirical and non-empirical studies for the other questions (i.e., Q2, Q4, Q5, and Q6). For the way to describe the context in which the research was carried out (Q2), the empirical studies significantly achieved better quality scores than the non-empirical studies. For Q2, 40 of the 57 empirical papers achieved a score equal to 1 (70.2% of the empirical papers), while only 15 of the 41 non-empirical papers achieved the same score (36.6% of the non-empirical papers). For presenting data that supports their findings (Q4), the empirical studies significantly achieved better quality scores than the non-empirical studies. For Q4, 51 of the 57 empirical papers achieved a score equal to 1 (89.5% of the empirical papers), while only 19 of the 41 non-empirical papers achieved the same score (46.3% of the non-empirical papers). For critically examining their own role, potential bias, and influence during the study (Q5), the empirical studies also significantly achieved better quality scores than the non-empirical studies. For Q5, 16 of the 57 empirical papers achieved a score equal to 1 (28.1% of the empirical papers), while only 3 of the 41 non-empirical papers achieved the same score (7.3% of the non-empirical papers). For describing the limitations and credibility of the work (Q6), the empirical studies also significantly achieved better quality scores than the non-empirical studies. For Q6, 15 of the 57 empirical papers achieved a score equal to 1 (26.3% of the empirical papers), while only 4 of the 41 non-empirical papers achieved the same score (9.8% of the non-empirical papers).

Therefore, empirical studies achieved significantly better quality scores than the non-empirical studies for Q2, Q4, Q5, and Q6. Furthermore, independently of the type of study, most of the papers scored the lowest on Q5 and a medium-quality score on Q6.

With regard to the evaluation context, the performed analysis shows that the quality scores are related to the evaluation contexts but that not all of the evaluation contexts have significant differences. Specifically, the differences are only significantly important when we compare the most distant contexts, i.e., when no evaluation (0) is compared to an evaluation based on academic cases (3), when no evaluation (0) is compared to industrial cases (4), when an evaluation based on a demonstration or toy examples (1) is compared to an evaluation based on academic cases (3), and when an evaluation based on a demonstration or toy examples (1) is compared to industrial cases (4). Specifically, the primary studies with more basic evaluation contexts (i.e., no evaluation and an evaluation based on demonstration or toy examples) achieve worse quality scores than the superior evaluation contexts (i.e., evaluation based on academic cases and an evaluation based on industrial cases). Primary studies with no evaluation achieve a mean quality score of 2.54, and primary studies with an evaluation based on demonstration or toy examples achieve a mean quality score of 3.15. In contrast, primary studies with an evaluation based on academic cases achieve a mean quality score of 4.19, and primary studies with an evaluation based on industrial cases achieve a mean quality score of 4.83.

Finally, the analysis also indicates that there is a correlation between the type of study and the evaluation context. Specifically, there is a higher number of non-empirical studies that do not have an evaluation or have an evaluation based on a demonstration or toy examples. Of the 24 primary studies that do not have an evaluation or have an evaluation based on a demonstration or toy examples, 21 are non-empirical studies (87.5% are non-empirical). In contrast, there is a higher number of empirical studies that have an evaluation based on academic cases or industrial cases. Of the 69 primary studies that have an evaluation based on academic cases or industrial cases, 52 are empirical studies (75.4% are empirical).

5.3 RQ3. What are the limitations of the existing approaches?

In this section, we outline the limitations of the existing approaches based on F17 of the data collection form. We also identified the existence or non-existence of support tools (F14). If they do not exist, this would be a limitation for the practical applicability of the approach. Based on F17, the following topics are recurrently reported as limitations:

- The size and quality of the datasets is not always enough or can be improved. To use supervised ML techniques, a dataset with labelled examples is needed to perform the training and testing of a classifier. However, a dataset with the appropriate number of examples in the datasets is not always available. Moreover, there is a need for the examples in the datasets to cover all possible scenarios (balanced datasets). Therefore, both the size and the quality of the datasets are critical to successfully apply ML techniques of this kind. If an appropriate dataset is not available, researchers cannot use the approaches proposed by some of the primary studies. Some authors highlight that part of their training or testing datasets is synthetic. They consider this as a possible point of future study by comparing synthetic datasets with real datasets. Furthermore, many authors point out the need for finding larger balanced datasets in their studies.
- The vocabulary can lead to limitations when approaches use the terms in models. Some works take advantage of the text in model elements to apply ML techniques. These studies, which usually use techniques to process texts, have to deal with the ambiguities of natural language. For example, the authors in [SLR77, SLR76] point out the need for a larger set of vocabulary and word embeddings of higher dimensions. In [SLR66], the author points out that some words do not carry any meaning and lead to inaccuracy when counting the occurrence of the words. In [SLR48], the authors highlight the out-of-vocabulary (OOV) problem as a problem for embedding the model terms.
- Different types of models and ML techniques can be explored. Most of the works focus on one or at most two types of models. Therefore, many authors propose extending their research to other models. Similarly, most authors propose approaches that are based on a specific ML technique. In this case, in addition to exploring other ML techniques, some authors highlight the need to compare their works with other ML-based approaches.
- The generalization and capacity to scale the approaches are difficult to guarantee. The authors usually test the approaches in specific domains, so it is necessary to test their approaches in other domains in order to verify the generalization and scalability of their approaches. More than 20% of the primary studies explicitly highlight generalization as a limitation or future work.

Other limitations are related to specific problems, such as the use of a single configuration to tune the parameters, the presentation of a domain-dependent approach, the use of few features or features that are unrepresentative for the training and testing process, or the lack of a support tool to evaluate the proposed approach. In fact, regarding tool support, 36 approaches of the primary studies are evaluated using automatic tools, and 25 approaches are evaluated using semi-automatic tools. However, most studies do not make their tools available to other researchers and practitioners, although in some cases they do publish their datasets. Furthermore, 24 studies do not present a tool.

Note that the collected data highlights issues that are not limited to solving MDE problems using ML. These include the availability of appropriate datasets, the selection and comparison of different techniques, or the generalization of the findings. In addition, two necessary avenues of research have been identified: the use of texts in models (vocabulary) and the design of approaches that do not depend on the type of model (different types of models). While execution time is reported as a limitation, almost no study provides information on execution time costs.

6 Discussion

Through a thorough analysis of the results obtained by our SLR, we have raised a series of discussion points regarding the state of the art on MDE using ML. The following subsections bring forth such a discussion.

Note that most of the studies are not associated to a specific software engineering activity. Moreover, some of the studies tackle more than one MDE problem. These results confirm that an approach does not necessarily have to have a single purpose. Therefore, designing approaches that are independent of the domain, the type of model, or even the ML technique in use could allow authors to exploit the approaches for other research areas.

6.1 Usage of ML techniques

From the gathered results, it becomes clear that the last few years have seen a rise in the usage of neural networks. There is a lone appearance of the use of neural networks in the first half of the period under study, a single paper in 2013. Then, there is an appreciable increase in the number of works that comprise neural networks from 2017 onwards. However, there seems to be no tangible evidence regarding the reasons for this increment in the usage of neural networks. Within the works that utilize them, we were unable to find a coherent rationale behind the selection of neural networks over other techniques, nor a comprehensive guide that presents the pros and cons of using neural networks instead of other techniques.

Why is it, then, that neural networks have increased in popularity in such a short period of time? We theorize that this is due to recent advances in hardware (with increases in GPU and CPU performance), the widespread access to neural network libraries, and recent developments in dataset creation and availability. Additionally, while it is true that the results obtained by neural networks have been below expectations for many years in different fields of knowledge, it is also reasonable to say that the hype surrounding these techniques has not diminished. Many current research projects and job positions are centered on the application of neural networks to real-world challenges, and researchers and developers alike have kept working on the development of neural network and the necessary technologies to support them. These factors have led to the appearance of specific applications that enable the creation and management of neural networks and the creation of didactic materials that have made them more accessible to the general public.

In fact, the rise of neural networks is not a strange or unusual occurrence. Many recent review articles highlight the proliferation of neural networks in different research areas [58]–[60]. However, as in our case, there does not seem to be a specific cause or motivation for this. Our hypothesis is that in 2013, the technology (i.e., tools and datasets for modeling) had not yet reached the necessary maturity to produce the inertia that we have seen in our results. In contrast, from 2017 onwards, the combination of widespread access, ease of use, and the maturity of neural network techniques has pushed more researchers to explore the application of neural networks in their works. In turn, this has translated into an increase in the number of success stories, producing a call effect and coaxing more practitioners towards neural networks. It is our belief that this has produced a virtuous circle that will govern the direction of research in the next few years.

Finally, the results also show that a percentage of the community continues to focus on decision, regression, and classification trees. The number of papers through time has remained relatively stable, with 24 papers in total over the full period of time under study. This situation defies the paradox of the peak of neural networks. It is clear that an important part of the community is still devoted to improving and applying other ML techniques to the full extent of their capabilities. Nonetheless, it remains to be seen whether the virtuous circle of neural networks produces a shift in the usage of these techniques in future years.

6.2 Models and their characteristics

Based on the above discussion, we consider ML techniques to be in vogue. As a result, we automatically think of Large Language Models (LLMs). Although there are already several papers investigating the use of LLMs in MDE [61]–[64], it is not unreasonable to expect more exploration of LLMs for solving MDE problems in future studies. This would be a different line of research that could even eclipse neural networks.

However, we should not lose sight of a differentiating characteristic of the field under study: in MDE, the main artifacts in use are software models. The consensus in the primary studies is that ML techniques understand models better than other methods due to their innate ability to detect patterns. However, this not only depends on the ML technique but also on the type of model.

There are different models and metamodels in the market (UML, BPMN, models based on DSL, etc.) with different characteristics. Some of them contain a high percentage of elements that only count with graphical representations and that do not provide textual patterns to be exploited. Contradictorily, in most studies, models are encoded using the text of their elements (if they exist).

Perhaps, one way to advance in the application of ML techniques in models could be to consider the models' own characteristics as the semantic information, instead of encoding the models based only on the text. Although there are some studies on this subject [SLR56, SLR3], this way of encoding models is still in its early stages.

It is also up to debate whether models present less noise (i.e., less irrelevant or contradictory information) than other kinds of input software artifacts, or other models. Models can be inherently different in nature depending on the role they play in software projects: while some models are nothing more than diagrams to help organize ideas on paper, other models are used as blueprints by the developers or even executed directly as code. In the more informal models, ML techniques struggle to retrieve information since the noise level in the models is comparable to and even greater than that of other artifacts.

This factor is not considered or discussed in the reviewed literature. We have determined that this is due to the fact that we are facing a first wave of works that, while successful, are not considering the particularities of the models and metamodels in use, but rather treating them as code or other documents with standard natural language in them. Future research efforts should explore these aspects of models and take these particularities into consideration.

6.3 Datasets and replication packages

According to the limitations and future lines of research, it is still difficult to find datasets to evaluate the approaches proposed in the research work. It is common knowledge that certain ML techniques require a larger number of examples to perform successful training (e.g., neural networks). However, this seems to be a general problem regardless of the ML technique.

To minimize this problem, we have compiled a list of datasets from the reviewed studies. Specifically, this list contains the public datasets that are being used to apply ML techniques in MDE (Table 13). Although this information is relevant for both researchers and practitioners in the field, we would like to mention a couple of points that have caught our attention.

First, most of these repositories do not provide information about their datasets. In other words, it is complicated to determine the type of model, the number of models available, or even the characteristics of these models (i.e., their provenance, their size, etc.). Therefore, the researcher has to browse through the repository to get an idea of its contents. Perhaps a future opportunity for improvement would be to improve the way datasets are reported.

Second, some of the primary studies indicate that they have extended some of the datasets in the list to evaluate their approaches. Among these studies, some studies have even used model generators such as RandomEMF [65] or VIATRA-Generator [66]. However, they have not published the final dataset, so it may not be possible to replicate their work or obtain similar results using only the base dataset.

In addition to the previous list of datasets, some studies use their own datasets. In fact, some studies provide replication packages containing both the dataset used and other complementary materials to their work (e.g., the developed tool). Table 14 presents the replication packs of the reviewed studies. Specifically, this table highlights the reference study, the availability of its dataset (i.e., Yes or No), the availability of complementary artifacts (i.e., Yes or No), and the website address of the repository where the pack is available.

It is worth mentioning that some researchers publish complementary material but do not publish the datasets due to ethical considerations [SLR14]. Others publish only a sample of their datasets due to legal considerations because their datasets are industrial and belong to a company [SLR55, SLR56]. Of course, it is important that the knowledge generated through research can be applied in industry. In addition, it is clear that legal and ethical considerations must be taken into account. Perhaps the community could discuss a way to find a balance between publishing materials to replicate results and exploiting data or materials that come from industry.

Finally, another problem worth mentioning is outdated or expired links. Several of the reviewed studies reference replication packages or datasets in repositories that are no longer available [SLR52, SLR8, SLR5, SLR61, SLR89, SLR67, SLR64, SLR66, SLR37, SLR35, SLR30, SLR3]. This problem leads us to wonder whether in 10 or 20 years the current references will no longer be available. If this could well be the case, we have noticed that materials in recent years have been published in repositories that allow identification by DOI rather than using personal or university repositories. From our point of view, this could be a great initiative to minimize this problem from happening in the future.

6.4 Maturity level of current research

The collected results show that most studies report their evaluations, data, and results to a greater or lesser extent. However, there is still room for improvement. Apart from the fact that many studies do not make their material available to other researchers and professionals, we also found papers that do not specify the ML techniques or the type of models used. This complicates the replication of the experiments in future academic works and makes their use in industrial environments impossible.

Specifically, we used four metrics to analyze the maturity level of current research: quality score, citation count, type of study (i.e., empirical or non-empirical), and evaluation context. However, the results suggest that citation counts may not be fair for studies published in recent years. Therefore, we recommend researchers take this into account before drawing conclusions. The only thing that we find relevant to highlight from this metric is that ML application in MDE seems to be an attractive topic to investigate, given the considerable number of citations that continues to increase.

With regard to the quality score, we noted that authors typically describe the motivation, context, and research design. However, supporting conclusions and critical reflections on the role of the author and the limitations of the study are usually not sufficiently described. The total quality score of the studies is moderate, although studies published in journals usually present a higher quality.

With regard to the type of study, the results indicate that there is a relevant difference between the empirical and non-empirical studies. In general, the maturity of empirical studies is better than the maturity of non-empirical studies. Overall, the non-empirical studies usually provide preliminary ideas that most of the time are poorly described, and they are not evaluated or evaluated using toy examples or expert observations. For these reasons, most of the non-empirical studies achieve lower quality scores (around 3.1) than empirical studies (around 4.5). As repeatedly mentioned by several researchers, systematic evaluations are needed [35, 67, 68]. Our results confirm that this observation also applies to MDE problems solved using ML, especially for non-empirical studies.

With regard to the evaluation context, our study of the available literature also brings light to the current state of the applicability of research results to practice and real-world scenarios. The gathered distribution of the primary studies for the evaluation context reveals that a majority of the works in the field (55.9%) evaluate their approaches based on academic cases. Remarkably, 14.7% of the works evaluated their approaches with toy examples or expert observations, and 13.7% of the works presented no evaluation whatsoever. In other words, a total of 84.3% of the works do not consider the applicability in industry. It is also noticeable that all of the works that evaluate their approaches based on industrial cases do not provide evidence to suggest or support their factual standardization and daily usage in their

TABLE 13: Public datasets with their names, URLs where a brief description of the dataset is published, and the primary studies that use them

Datasets	URL	Description	Primary Studies
Alloy Models	https://github.com/AlloyTools/models	This repository holds public Alloy models to be used as entertainment, examples, tutorials, utilities, and proofs.	[SLR20], [SLR24]
Occiware	https://github.com/occiware/ecore/	This repository contains the OCCI meta-model and examples described with EMF plus OCCI extensions.	[SLR11]
amlMeta-Model	https://github.com/amlModeling/amlMetaModel	This repository contains EMF Meta-Models for AutomationML.	[SLR11]
EMF-Fragments	https://github.com/markus1978/emf-fragments	This is a framework for EMF-models, thorugh sets of larger fragments and not based on an object relational mappings.	[SLR11]
MDEForge	https://github.com/MDEGroup/MDEForge	This platform is to foster a community-based repository for developing, analyzing, and reusing modeling artifacts.	[SLR11]
AtlanMod Zoo	https://github.com/atlanmod/zoo	This is a set of Eclipse Modeling Framework (EMF) Ecore models.	[SLR94], [SLR35], [SLR44], [SLR7], [SLR6], [SLR5]
Lindholmen dataset	http://models-db.com/	The dataset of this initiative includes links to more than 93,000 UML files (spread across more than 24,000 GitHub repositories).	[SLR71], [SLR13], [SLR12], [SLR67], [SLR85], [SLR84], [SLR89], [SLR43]
ModelSet	https://modelset.github.io/	ModelSet is a dataset composed of 5,460 Ecore and 5,120 UML labelled models, making a total of 10,580 models.	[SLR71], [SLR15], [SLR48], [SLR25]
A dataset for clustering	https://zenodo.org/records/2585432 https://zenodo.org/records/2585456	Manually labeled 555 metamodels mined from GitHub in April 2017.	[SLR73], [SLR41], [SLR6], [SLR25]
OntoUML/ UFO Catalog	https://github.com/OntoUML/ontouml-models	This Catalog is a collaborative, structured and open-source catalog of OntoUML and UFO ontology models.	[SLR3]

TABLE 14: Replication packages for the primary studies: reference of the study, URL where the pack is published, the availability of its dataset (DS), the availability of complementary artifacts (CA)

Study	URL	DS	CA
[SLR49]	https://github.com/Antolin1/TCRMG-GNN	Yes	Yes
[SLR24]	https://github.com/MDEGroup/MORGAN	Yes	Yes
[SLR81]	https://github.com/MeMartijn/text2uml	Yes	Yes
[SLR73]	https://github.com/Models-Lucene2021/Metamodel_Clustering_Classification	Yes	Yes
[SLR62]	https://github.com/MDEGroup/AURORA	Yes	Yes
[SLR17]	https://github.com/modelia/ann-for-mts/tree/master	Yes	Yes
[SLR14]	https://github.com/YounesB-McGill/uml-grader	No	Yes
[SLR26]	https://s-case.github.io/publications/eis2017/	Yes	Yes
[SLR55]	https://bitbucket.org/svitusj/flame/src/master/	Yes*	Yes
[SLR57]	https://goo.gl/YRrXSp	Yes	Yes
[SLR56]	https://bitbucket.org/svitusj/flame/src/master/FLiM_ML/	Yes*	Yes
[SLR9]	https://github.com/MagMar94/ParmorelRunnable	Yes	Yes
[SLR12]	https://zenodo.org/records/6645685	Yes	Yes
[SLR82]	https://github.com/MSharbaf/CoReRL	No	Yes
[SLR16]	https://github.com/modelia/ai-for-model-manipulation	Yes	Yes
[SLR79]	https://zenodo.org/records/7007647	Yes	Yes
[SLR63]	https://github.com/MDEGroup/AURORA/	Yes	Yes
[SLR84]	https://zenodo.org/records/4595957	Yes	No
[SLR94]	https://github.com/songyang-dev/uml-classes-and-specs	Yes	No
[SLR48]	https://github.com/models-lab/worde4mde	No	Yes
[SLR25]	https://github.com/MDEGroup/MORGAN	Yes	Yes
[SLR70]	https://github.com/AbbasRahimi/netgan/tree/Ecore_model_generator	Yes	Yes
[SLR43]	https://gitlab.univ-lille.fr/emmanuel.renaux/neural-uml	Yes	Yes
[SLR47]	https://github.com/Antolin1/M2	Yes	Yes

*Only part of the dataset is available due to legal industry considerations.

respective industrial settings. Overall, we can conclude that, while a part of the community has converged towards industry, current technology in general is not mature enough to be applicable in the day-to-day of industrial scenarios.

Nevertheless, the usage of academic cases is not a novel issue first identified by our work. Rather, it is a common occurrence in current technological research in general: the evaluation of approaches and techniques on academic problems and datasets has been identified as an endemic problem, mentioned by all previous SLR and survey papers referenced in the Related Works section. However, in our particular field of study, the results indicate that the technology is mature enough to start getting closer to industry and to achieve success with industrial datasets. While the technology may not be mature enough to be used on a daily basis by industrial practitioners, our recommendation is for the research community to break the current trend and to shift their progress towards the industrial application of their approaches and techniques.

7 THREATS TO VALIDITY

In this section, we use the classification of threats to validity of [69] and the study of threats to validity of systematic literature reviews in [70] to acknowledge the limitations of our survey.

Construct validity: This aspect of validity reflects the extent to which the operational measures that are studied represent what the researchers have in mind. Examples of issues are whether the concepts are defined clearly enough before measurements are defined, and interaction of different treatments when persons are involved in more than one study [71].

- To avoid the threat of non-specification of survey settings and sufficient details, we describe the search string, the digital sources, the inclusion and exclusion criteria, and the data collection process.

- To minimize the threat of inappropriate or incomplete search terms in automatic search, we used the steps suggested by Kitchenham and Charters [12]. First, we used PICO (Population, Intervention, Comparison, and Outcomes) criteria to derive the major terms from the RQs. Then, we identified synonyms for these terms. Finally, we verified the search terms that other relevant surveys used in their search strings. Despite minimizing this validation threat, we find two limitations that impact our search. First, some acronyms proved to harm the search. None of the primary studies were included because of the terms: MDE, MDD, MDA. They only increased the noise of the search. For example, 1,414 studies were found due to the acronym MDD. However, all of them were discarded because they were not related to Model Driven Engineering. Most of them are related to Major Depressive Disorder (MDD) or Maximal Dry Density (MDD). Second, some terms are present in both the Population terms (MDE terms) and the Intervention terms (ML terms). For example, we can use the term 'model' to describe the ML classifier generated from a set of labelled data or we can talk about the UML model designed for a system. For filtering, the terms must be identifiers of a single group. If 'model' is an MDE term, any paper with an ML technique and the term model will be selected, even if that paper is not related to MDE in any way (e.g., supervised learning model). If 'model' is an ML term, any paper from MDE and the term model will be selected, even if that paper is not related to ML in any way (e.g., model-driven development). To avoid this problem, we have avoided including the terms 'technique' and 'model' because these two terms are frequently used in both the ML and MDE terms. However, this also means that some papers might be lost in the process. To minimize this threat, backward and forward snowball iterations were applied until no new papers were found.
- To avoid the threat of an incorrect search method, our search strategy is based on the guidelines provided in [32, 33].
- To avoid the threat of inappropriate exclusion and inclusion criteria, the exclusion criteria were defined according to criteria used commonly in surveys, and the inclusion criteria were defined by domain experts (reducing the probability of not including studies whose conclusion could be relevant for our research field).
- To minimize the threat of inappropriate research questions, specific sessions were held with all of the co-authors to discuss and assess which research questions were most appropriate in order to provide useful information about the research field to both researchers and practitioners.

Internal Validity: This aspect of validity is of concern when causal relations are examined. There is a risk that the factor being investigated may be affected by other neglected factors. In this work, the outcomes could be affected by how the primary studies are selected.

- To minimize the threat of misclassifying primary studies, the studies were classified by two authors of the paper independently. Then, they compared their results, and, in the case of disagreement, they discussed the classification in order to reach a consensus.
- To avoid the threat of primary study duplication, the duplication was included within the exclusion criteria.
- To minimize the threat of bias in study selection, the selection was conducted by two authors of the paper. Therefore, the selection process was double-checked. Moreover, they strictly followed the search strategy defined for the selection.
- To minimize the threat of bias in data extraction, the extraction was conducted by two authors of the paper. Moreover, the features for data extraction were identified before starting the extraction process.

External Validity: This aspect of validity is concerned with to what extent it is possible to generalize the findings and to what extent the findings are of relevance for other cases. There is a risk that the papers recovered by the search are not representative of the target population.

- To minimize the threat of incomplete research information or conclusions by primary studies, we used snowballing to avoid missing relevant studies.
- To minimize the threat of generalizability of primary studies, we consider fields (i.e., MDE and ML) whose research is sufficiently advanced to draw interesting implications for researchers and practitioners. In fact, this work is based on 98 primary studies. However, it is necessary to keep in mind that the conclusions are limited to the application of ML in MDE problems. There may be more MDE problems that have not been manually identified in this review, either because they are not being solved by ML or because they are problems that have begun to be addressed at a later date than this review.
- To minimize the threat of a restricted timespan, in our search method, we included papers from a timespan longer than a decade. For our study, we considered papers up to the date on which the search was performed in the digital sources.

Conclusion validity: This aspect is concerned with to what extent the process can be replicated with the same results. Researchers may influence the result by looking for a specific outcome.

- To minimize the threat of a subjective interpretation about the extracted data, we clearly separate the results from the discussion. The results show the data that was extracted to answer the proposed RQs. The discussion provides different points with regard to the state of the art in the research field based on the interpretation of the results made by the authors of the study.
- To avoid the threat of the replication of the study, the documentation for all of the steps of the whole search process is publicly available.

8 Conclusions

MDE is a software engineering methodology based on the systematic use of models throughout the software development cycle to capture and design the characteristics of software systems. Major players in the field have significantly increased their use of model-based technologies to successfully develop industrial software. Recently, there have also been remarkable advancements in the AI domain, especially ML techniques, which have been successfully applied to solve MDE challenges.

For this work, we have systematically reviewed the works in the intersection of ML and MDE. We have reviewed a total of 9,194 papers, selecting 98 studies for further analysis. The results of the SLR shed light on the current state of the art. Through a detailed analysis of the results, we discuss the drift in the usage of the different available ML techniques, the models and their characteristics as a future line of research, the availability of public datasets and other materials to replicate and generalize the results, the trends regarding the study of different MDE problems, the maturity of the ML-based approaches for solving those MDE problems, and the remaining open challenges considering the maturity level of the current research.

The general view provided by this paper can be useful for practitioners and researchers alike when studying the current state of affairs in the field. Additionally, through the recommendations provided along with the discussion of the results, our SLR has the potential to produce a positive impact on the research community and support its transition towards ML. This can be beneficial, since compared to traditional approaches, ML techniques are better equipped to deal with the innate intricacies of MDE. Nonetheless, our study places the focus on enhancing the understanding of those works that leverage ML to solve MDE problems. This paper could be complemented by analyzing related fields of research that are left open as interesting opportunities for future literature exploration in the form of reviews that tackle either the usage of MDE to improve ML approaches or the combination of ML and MDE to solve a problem within other areas of knowledge.

Acknowledgements

This work was supported in part by the Ministry of Economy and Competitiveness (MINECO) through the Spanish National R+D+i Plan and ERDF funds under the Project VARIATIVA under Grant PID2021-128695OB-I00, and in part by the Gobierno de Aragón (Spain) (Research Group S05_20D). This work was partially supported by the Spanish Ministry of Science, Innovation and Universities under the Project VARNETICA (CNS2023-145422), and the Excellence Network AI4Software (Red2022-134647-T).

References

[1] M. Brambilla, J. Cabot, and M. Wimmer, "Model-driven software engineering in practice," *Synthesis Lectures on Software Engineering*, vol. 1, no. 1, pp. 1–182, 2012.

[2] S. Winkler and J. Pilgrim, "A survey of traceability in requirements engineering and model-driven development," *Software and Systems Modeling (SoSyM)*, vol. 9, no. 4, pp. 529–565, 2010.

[3] G. Loniewski, E. Insfran, and S. Abrahão, "A systematic review of the use of requirements engineering techniques in model-driven development," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2010, pp. 213–227.

[4] D. Di Ruscio, R. F. Paige, and A. Pierantonio, "Guest editorial to the special issue on success stories in model driven engineering," *Science of Computer Programming*, vol. 89, no. PB, pp. 69–70, 2014.

[5] J. Font, L. Arcega, Ø. Haugen, and C. Cetina, "Achieving feature location in families of models through the use of search-based software engineering," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 363–377, 2017.

[6] F. Pérez, R. Lapeña, A. C. Marcén, and C. Cetina, "Topic modeling for feature location in software models: Studying both code generation and interpreted models," *Information and Software Technology*, vol. 140, p. 106676, 2021.

[7] R. France and B. Rumpe, "Model-driven development of complex software: A research roadmap," in *Future of Software Engineering (FOSE'07)*. IEEE, 2007, pp. 37–54.

[8] B. Baudry, T. Dinh-Trong, J.-M. Mottu, D. Simmonds, R. France, S. Ghosh, F. Fleurey *et al.*, "Model transformation testing challenges," in *ECMDA workshop on Integration of Model Driven Development and Model Driven Testing.*, 2006.

[9] R. V. D. Straeten, T. Mens, and S. V. Baelen, "Challenges in model-driven software engineering," in *International conference on model driven engineering languages and systems*. Springer, 2008, pp. 35–47.

[10] G. Mussbacher, D. Amyot, R. Breu, J.-M. Bruel, B. H. Cheng, P. Collet, B. Combemale, R. B. France, R. Heldal, J. Hill *et al.*, "The relevance of model-driven engineering thirty years from now," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2014, pp. 183–200.

[11] D. Akdur and O. Demirörs, "Systematic reviews in model-driven engineering: A tertiary study," *Journal of Aeronautics and Space Technologies*, vol. 13, no. 1, pp. 57–68, 2020.

[12] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[13] J.-M. Favre, "Megamodelling and etymology," in *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2006.

[14] J. Ludewig, "Models in Software Engineering–An Introduction," *Software and Systems Modeling*, vol. 2, no. 1, pp. 5–14, 2003.

[15] T. Kühne, "Matters of (meta-) modeling," *Software & Systems Modeling*, vol. 5, no. 4, pp. 369–385, 2006.

[16] A. Shabtai, R. Moskovitch, Y. Elovici, and C. Glezer, "Detection of malicious code by applying machine learning classifiers on static features: A state-of-the-art survey," *information security technical report*, vol. 14, no. 1, pp. 16–29, 2009.

[17] S. Walker, W. Khan, K. Katic, W. Maassen, and W. Zeiler, "Accuracy of different machine learning algorithms and added-value of predicting aggregated-level energy performance of commercial buildings," *Energy and Buildings*, vol. 209, p. 109705, 2020.

[18] D. D. Ruscio, P. T. Nguyen, and A. Pierantonio, "Machine learning for managing modeling ecosystems: Techniques, applications, and a research vision," in *Software Ecosystems: Tooling and Analytics*. Springer, 2023, pp. 249–279.

[19] H. Naveed, C. Arora, H. Khalajzadeh, J. Grundy, and O. Haggag, "Model driven engineering for machine learning components: A systematic literature review," *Information and Software Technology*, p. 107423, 2024.

[20] I. Santiago, A. Jiménez, J. M. Vara, V. De Castro, V. A. Bollati, and E. Marcos, "Model-driven engineering as a new landscape for traceability management: A systematic literature review," *Information and Software Technology*, vol. 54, no. 12, pp. 1340–1356, 2012.

[21] B. Uzun and B. Tekinerdogan, "Model-driven architecture based testing: A systematic literature review," *Information and Software technology*, vol. 102, pp. 30–48, 2018.

[22] C. Raibulet, F. A. Fontana, and M. Zanoni, "Model-driven reverse engineering approaches: A systematic literature review," *IEEE Access*, vol. 5, pp. 14 516–14 542, 2017.

[23] P. H. Nguyen, M. Kramer, J. Klein, and Y. Le Traon, "An extensive systematic review on the model-driven development of secure systems," *Information and Software Technology*, vol. 68, pp. 62–81, 2015.

[24] H. A. A. Alfraihi and K. C. Lano, "The integration of agile development and model driven development: A systematic literature review," *The 5th International Confrence on Model-Driven Engineeing and Software Development*, 2017.

[25] H. Tufail, F. Azam, M. W. Anwar, and I. Qasim, "Model-driven development of mobile applications: A systematic literature review," in *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 2018, pp. 1165–1171.

[26] E. de Araújo Silva, E. Valentin, J. R. H. Carvalho, and R. da Silva Barreto, "A survey of model driven engineering in robotics," *Journal of Computer Languages*, vol. 62, p. 101021, 2021.

[27] D. C. Schmidt *et al.*, "Model-driven engineering," *Computer-IEEE Computer Society-*, vol. 39, no. 2, p. 25, 2006.

[28] S. Kent, "Model driven engineering," in *International conference on integrated formal methods*. Springer, 2002, pp. 286–298.

[29] A. F. Subahi, "Cognification of program synthesis—A systematic feature-oriented analysis and future direction," *Computers*, vol. 9, no. 2, p. 27, 2020.

[30] Y. Rigou, D. Lamontagne, and I. Khriss, "A sketch of a deep learning approach for discovering UML class diagrams from system's textual specification," in *2020 1st International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*. IEEE, 2020, pp. 1–6.

[31] S. Elmidaoui, L. Cheikhi, A. Idri, and A. Abran, "Empirical studies on software product maintainability prediction: A systematic mapping and review," *E-Informatica Software Engineering Journal*, vol. 13, no. 1, 2019.

[32] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Information and software technology*, vol. 55, no. 12, pp. 2049–2075, 2013.

[33] B. A. Kitchenham, D. Budgen, and P. Brereton, *Evidence-based software engineering and systematic reviews*. CRC press, 2015, vol. 4.

[34] V. R. B. G. Caldiera and H. D. Rombach, "The goal question metric approach," *Encyclopedia of software engineering*, pp. 528–532, 1994.

[35] M. Galster, D. Weyns, D. Tofan, B. Michalik, and P. Avgeriou, "Variability in software systems—A systematic literature review," *IEEE Transactions on Software Engineering*, vol. 40, no. 3, pp. 282–306, 2013.

[36] J.-D. Kasher, S. Riedle, and N. J. Sardarabady, "Digitization technologies in transport logistics: A systematic literature review protocol," in *2022 IEEE 28th International Conference on Engineering, Technology and Innovation (ICE/ITMC) & 31st International Association For Management of Technology (IAMOT) Joint Conference*. IEEE, 2022, pp. 1–13.

[37] N. Ozkan, K. Eilers, and M. Ş. Gök, "Back to the essential: A literature-based review on agile mindset," in *2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 2023, pp. 201–211.

[38] F. D. Giraldo, S. España, and O. Pastor, "Analysing the concept of quality in model-driven engineering literature: A systematic review," in *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*. IEEE, 2014, pp. 1–12.

[39] M. Goulão, V. Amaral, and M. Mernik, "Quality in model-driven engineering: A tertiary study," *Software Quality Journal*, vol. 24, no. 3, pp. 601–633, 2016.

[40] A. Ali and C. Gravino, "A systematic literature review of software effort prediction using machine learning methods," *Journal of Software: Evolution and Process*, vol. 31, no. 10, p. e2211, 2019.

[41] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Information and Software Technology*, vol. 54, no. 1, pp. 41–59, 2012.

[42] C. Virmani, T. Choudhary, A. Pillai, and M. Rani, "Applications of machine learning in cyber security," in *Handbook of Research on Machine and Deep Learning Applications for Cyber Security*. IGI Global, 2020, pp. 83–103.

[43] S. Mirjalili, H. Faris, and I. Aljarah, "Introduction to evolutionary machine learning techniques," in *Evolutionary Machine Learning Techniques*. Springer, 2020, pp. 1–7.

[44] D. J. Hand, "Principles of data mining," *Drug safety*, vol. 30, no. 7, pp. 621–622, 2007.

[45] G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido, and M. Marchetti, "On the effectiveness of machine and deep learning for cyber security," in *2018 10th international conference on cyber Conflict (CyCon)*. IEEE, 2018, pp. 371–390.

[46] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *applied sciences*, vol. 9, no. 20, p. 4396, 2019.

[47] R. Shyam and R. Singh, "A taxonomy of machine learning techniques," *J. Adv. Robot*, vol. 8, no. 3, pp. 18–25, 2021.

[48] S. M. Asaad, K. Z. Ghafoor, H. Sarhang, A. Mulahuwaish, and A. M. Ali, "Fingerprinting based positioning techniques using machine learning algorithms principles, approaches and challenges," *Trust, Security and Privacy for Big Data*, pp. 112–128, 2022.

[49] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and software technology*, vol. 64, pp. 1–18, 2015.

[50] T. Dyba, T. Dingsoyr, and G. K. Hanssen, "Applying systematic reviews to diverse study types: An experience report," in *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*. IEEE, 2007, pp. 225–234.

[51] M. Alenezi and M. Akour, "Open-source machine learning software systems: Architectural analysis," *ICIC express letters. Part B, Applications: an international journal of research and surveys*, vol. 12, no. 11, pp. 1019–1026, 2021.

[52] A. Brossard, M. Abed, and C. Kolski, "Taking context into account in conceptual models using a model driven engineering approach," *Information and Software Technology*, vol. 53, no. 12, pp. 1349–1369, 2011.

[53] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.

[54] B. Kitchenham, "Procedures for performing systematic reviews," *Keele, UK, Keele University*, vol. 33, no. 2004, pp. 1–26, 2004.

[55] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and software technology*, vol. 50, no. 9-10, pp. 833–859, 2008.

[56] OMG, "OMG Unified Modeling Language (OMG UML)," https://www.omg.org/spec/UML/2.5.1/PDF, December 2012.

[57] V. Alves, N. Niu, C. Alves, and G. Valença, "Requirements engineering for software product lines: A systematic literature review," *Information and Software Technology*, vol. 52, no. 8, pp. 806–820, 2010.

[58] N. Chattopadhyay, C. S. Y. Viroy, and A. Chattopadhyay, "Re-markable: Stealing watermarked neural networks through synthesis," in *Security, Privacy, and Applied Cryptography Engineering: 10th International Conference, SPACE 2020, Kolkata, India, December 17–21, 2020, Proceedings 10*. Springer, 2020, pp. 46–65.

[59] S. Chaudhuri, D. Ritchie, J. Wu, K. Xu, and H. Zhang, "Learning generative models of 3d structures," in *Computer Graphics Forum*, vol. 39, no. 2. Wiley Online Library, 2020, pp. 643–666.

[60] A. Aswath, A. Alsahaf, B. N. Giepmans, and G. Azzopardi, "Segmentation in large-scale cellular electron microscopy with deep learning: A literature survey," *Medical image analysis*, p. 102920, 2023.

[61] V. Kulkarni, S. Reddy, S. Barat, and J. Dutta, "Toward a symbiotic approach leveraging generative ai for model driven engineering," in *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 2023, pp. 184–193.

[62] K. Chen, Y. Yang, B. Chen, J. A. H. López, G. Mussbacher, and D. Varró, "Automated domain modeling with large language models: A comparative study," in *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 2023, pp. 162–172.

[63] M. B. Chaaben, L. Burgueño, and H. Sahraoui, "Towards using few-shot prompt learning for automating model completion," in *2023 IEEE/ACM 45th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. IEEE, 2023, pp. 7–12.

[64] J. Cámara, J. Troya, L. Burgueño, and A. Vallecillo, "On the assessment of generative ai in modeling tasks: an experience report with chatgpt and uml," *Software and Systems Modeling*, vol. 22, no. 3, pp. 781–793, 2023.

[65] M. Scheidgen, "RandomEMF," https://github.com/markus1978/RandomEMF/, December 2015.

[66] O. Semeráth, "VIATRA-Generator," https://github.com/viatra/, April 2022.

[67] C. Zannier, G. Melnik, and F. Maurer, "On the success of empirical studies in the international conference on software engineering," in *Proceedings of the 28th international conference on Software engineering*, 2006, pp. 341–350.

[68] D. Weyns, M. U. Iftikhar, S. Malek, and J. Andersson, "Claims and supporting evidence for self-adaptive systems: A literature study," in *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 2012, pp. 89–98.

[69] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Springer Science & Business Media, 2012.

[70] X. Zhou, Y. Jin, H. Zhang, S. Li, and X. Huang, "A map of threats to validity of systematic literature reviews in software engineering," in *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2016, pp. 153–160.

[71] C. Wohlin, M. Höst, and K. Henningsson, "Empirical research methods in software engineering," in *Empirical methods and studies in software engineering*. Springer, 2003, pp. 7–23.

SYSTEMATIC REVIEW REFERENCES

[SLR1] H.A. Al-Jamimi and M.A. Ahmed. Knowledge acquisition in model driven development transformations: An inductive logic programming approach. volume 2015-January, 2015.

[SLR2] H.A. Al-Jamimi and M.A. Ahmed. Model driven development transformations using inductive logic programming. *International Journal of Advanced Computer Science and Applications*, 8(11):531–541, November 2017.

[SLR3] S.J. Ali, G. Guizzardi, and D. Bork. Enabling representation learning in ontology-driven conceptual modeling using graph neural networks. In *International Conference on Advanced Information Systems Engineering*, pages 278–294. Springer, 2023.

[SLR4] Ö. Babur. Clone detection for ecore metamodels using n-grams. In *MODELSWARD*, pages 411–419, 2018.

[SLR5] Ö. Babur and L. Cleophas. Using n-grams for the automated clustering of structural models. In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 510–524. Springer, 2017.

[SLR6] Ö. Babur, L. Cleophas, and M. Van den Brand. SAMOS-A framework for model analytics and management. *Science of Computer Programming*, 223:102877, 2022.

[SLR7] Ö. Babur. Statistical analysis of large sets of models. pages 888–891, 2016.

[SLR8] Ö. Babur, L. Cleophas, T. Verhoeff, and M. Van Den Brand. Towards statistical comparison and analysis of models. pages 361–367, 2016.

[SLR9] A. Barriga, L. Iovino, A. Rutle, and R. Heldal. Model repair with quality-based reinforcement learning. *Journal of Object Technology*, 19(2), 2020.

[SLR10] A. Barriga, A. Rutle, and R. Heldal. Automatic model repair using reinforcement learning. In *MoDELS (Workshops)*, pages 781–786, 2018.

[SLR11] A. Barriga, A. Rutle, and R. Heldal. Personalized and automatic model repairing using reinforcement learning. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 175–181. IEEE, 2019.

[SLR12] G. Bergström, F. Hujainah, T. Ho-Quang, R. Jolak, S.A. Rukmono, A. Nurwidyantoro, and M.R.V. Chaudron. Evaluating the layout quality of uml class diagrams using machine learning. *Journal of Systems and Software*, 192:111413, 2022.

[SLR13] N. Bnouni Rhim, S. Cheballah, and M. Ben Mabrouk. Cross synergetic mobilenet-VGG16 for UML multiclass diagrams classification. In *International Conference on Innovations in Bio-Inspired Computing and Applications*, pages 24–30. Springer, 2022.

[SLR14] Y. Boubekeur, G. Mussbacher, and S. McIntosh. Automatic assessment of students' software models using a simple heuristic and machine learning. pages 84–93, 2020.

[SLR15] M. Bragilovski, F. Dalpiaz, A. Sturm, et al. From user stories to domain models: Recommending relationships between entities. In *Proceedings of the Workshop on Natural Language Processing in Requirements Engineering (NLP4RE'23)*, volume 3378, pages 1–11. CEUR Workshop Proceedings, 2023.

[SLR16] L. BurgueÑo, J. Cabot, S. Li, and S. Gérard. A generic LSTM neural network architecture to infer heterogeneous model transformations. *Software and Systems Modeling*, 21(1):139–156, 2022.

[SLR17] L. Burgueño, J. Cabot, and S. Gérard. An LSTM-based neural network architecture for model transformations. pages 294–299, 2019.

[SLR18] T. Capuano, H. Sahraoui, B. Frenay, and B. Vanderose. Learning from code repositories to recommend model classes. *Journal of Object Technology*, 21(3):3, 2022.

[SLR19] Alisha Sharma Chapai and Eric J Rapos. SkeMo: Sketch modeling for real-time model component generation. In *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 173–183. IEEE, 2023.

[SLR20] R. Clarisó and J. Cabot. Diverse scenario exploration in model finders using graph kernels and clustering. In *International Conference on Rigorous State-Based Methods*, pages 27–43. Springer, 2020.

[SLR21] R. Clarisó and J. Cabot. Applying graph kernels to model-driven engineering problems. pages 1–5, 2018.

[SLR22] Guilherme Dalcin, Willian Bolzan, Luan Lazzari, and Kleinner Farias. Recommendation of UML model conflicts: Unveiling the biometric lens for conflict resolution. In *Proceedings of the XXXVII Brazilian Symposium on Software Engineering*, pages 83–88, 2023.

[SLR23] A. Del Pino Lino and A. Rocha. Automatic evaluation of ERD in e-learning environments. volume 2018-June, pages 1–5, 2018.

[SLR24] J. Di Rocco, C. Di Sipio, D. Di Ruscio, and P.T. Nguyen. A GNN-based recommender system to assist the specification of metamodels and models. In *2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 70–81, 2021.

[SLR25] Claudio Di Sipio, Juri Di Rocco, Davide Di Ruscio, and Phuong T Nguyen. MORGAN: a modeling recommender system based on graph kernel. *Software and Systems Modeling*, 22(5):1427–1449, 2023.

[SLR26] T. Diamantopoulos and A. Symeonidis. Enhancing requirements reusability through semantic modeling and data mining techniques. *Enterprise Information Systems*, 12(8-9):960–981, 2018.

[SLR27] Xavier Dolques, Marianne Huchard, Clémentine Nebut, and Philippe Reitz. Learning transformation rules from transformation examples: An approach based on relational concept analysis. In *2010 14th IEEE International Enterprise Distributed Object Computing Conference Workshops*, pages 27–32. IEEE, 2010.

[SLR28] M. Eisenberg, H.P. Pichler, A. Garmendia, and M. Wimmer. Towards reinforcement learning for in-place model transformations. In *2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 82–88. IEEE, 2021.

[SLR29] K. El Dahshan, N. Aboalanin, A. Tomoum, and M. Sameh. Useinator: Requirements collection automation. pages 461–464, 2018.

[SLR30] M. Essaidi, A. Osmani, and C. Rouveirol. Transformation learning in the context of model-driven data warehouse: An experimental design based on inductive logic programming. pages 693–700, 2011.

[SLR31] D. Godara and R.K. Singh. Improving change proneness prediction in UML based design models using ABC algorithm. pages 1296–1301, 2014.

[SLR32] D. Godara and R.K. Singh. Enhancing frequency based change proneness prediction method using artificial bee colony algorithm. *Advances in Intelligent Systems and Computing*, 320:535–543, 2015.

[SLR33] B. Gosala, S.R. Chowdhuri, J. Singh, M. Gupta, and A. Mishra. Automatic classification of UML class diagrams using deep learning technique: Convolutional neural network. *Applied Sciences (Switzerland)*, 11(9), 2021.

[SLR34] A. Halim. Predict fault-prone classes using the complexity of UML class diagram. pages 289–294, 2013.

[SLR35] S.J.I. Herzig and C.J.J. Paredis. Bayesian reasoning over models. volume 1235, pages 69–78, 2014.

[SLR36] A.T. Imam. The automatic definition of the intuitive linguistic heuristics set to recognize the elements of UML analysis and design models in english. *IEEE Access*, 2023.

[SLR37] M. Jahan, Z. Shakeri H. Abad, and B. Far. Detecting use case scenarios in requirements artifacts: A deep learning approach. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 682–694. Springer, 2022.

[SLR38] R. Jebli, J. Elbouhdidi, and M.Y. Chkouri. Automatic evaluation of UML class diagrams using the xml schema matching and the machine learning algorithm. *Advances in Intelligent Systems and Computing*, 1105 AISC:149–156, 2020.

[SLR39] S. Kansomkeat, P. Thiket, and J. Offutt. Generating test cases from UML activity diagrams using the condition-classification tree method. In *2010 2nd International conference on software technology and engineering*, volume 1, pages V1–62. IEEE, 2010.

[SLR40] A. Khalilipour, F. Bozyigit, C. Utku, and M. Challenger. Categorization of the models based on structural information extraction and machine learning. In *International Conference on Intelligent and Fuzzy Systems*, pages 173–181. Springer, 2022.

[SLR41] A. Khalilipour, F. Bozyigit, C. Utku, and M. Challenger. Machine learning-based model categorization using textual and structural features. In *European Conference on Advances in*

Databases and Information Systems, pages 425–436. Springer, 2022.

[SLR42] T. Kochbati, S. Li, S. Gérard, and C. Mraidha. From user stories to models: A machine learning empowered automation. pages 28–40, 2021.

[SLR43] Aymeric Koenig, Benjamin Allaert, and Emmanuel Renaux. NEURAL-UML: Intelligent recognition system of structural elements in UML class diagram. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 605–613. IEEE, 2023.

[SLR44] K. Lano, S. Fang, M.A. Umar, and S. Yassipour-Tehrani. Enhancing model transformation synthesis using natural language processing. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, pages 1–10, 2020.

[SLR45] K. Lano, S. Kolahdouz-Rahimi, and S. Fang. Model transformation development using automated requirements analysis, metamodel matching, and transformation by example. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(2):1–71, 2021.

[SLR46] K. Lano, S. Yassipour-Tehrani, and M.A. Umar. Automated requirements formalisation for agile MDE. pages 173–180, 2021.

[SLR47] José Antonio Hernández López and Jesús Sánchez Cuadrado. Generating structurally realistic models with deep autoregressive networks. *IEEE Transactions on Software Engineering*, 2022.

[SLR48] José Antonio Hernández López, Carlos Durá, and Jesús Sánchez Cuadrado. Word embeddings for model-driven engineering. In *2023 ACM/IEEE 26th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 151–161. IEEE, 2023.

[SLR49] J.A.H. López and J.S. Cuadrado. Towards the characterization of realistic model generators using graph neural networks. In *2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 58–69, 2021.

[SLR50] K. Madala, D. Gaither, R. Nielsen, and H. Do. Automated identification of component state transition model elements from requirements. pages 386–392, 2017.

[SLR51] K. Madala, S. Piparia, E. Blanco, H. Do, and R. Bryce. Model elements identification using neural networks: A comprehensive study. *Requirements Engineering*, 26(1):67–96, 2021.

[SLR52] K. Madala, S. Piparia, H. Do, and R. Bryce. Finding component state transition model elements using neural networks: An empirical study. In *2018 5th international workshop on artificial intelligence for requirements engineering (AIRE)*, pages 54–61. IEEE, 2018.

[SLR53] M. Maddeh, S. Ayouni, S. Alyahya, and F. Hajjej. Decision tree-based design defects detection. *IEEE Access*, 9:71606–71614, 2021.

[SLR54] N. Maneerat and P. Muenchaisri. Bad-smell prediction from software design model using machine learning techniques. In *2011 Eighth international joint conference on computer science and software engineering (JCSSE)*, pages 331–336. IEEE, 2011.

[SLR55] A.C. Marcén, R. Lapeña, Ó. Pastor, and C. Cetina. Traceability link recovery between requirements and models using an evolutionary algorithm guided by a learning to rank algorithm: Train control and management case. *Journal of Systems and Software*, 163:110519, 2020.

[SLR56] A.C. Marcén, F. Pérez, Ó. Pastor, and C. Cetina. Enhancing software model encoding for feature location approaches based on machine learning techniques. *Software and Systems Modeling*, 21(1):399–433, 2022.

[SLR57] A.C. Marcén, F. Pérez, Ó. Pastor, and C. Cetina. Evaluating the benefits of empowering model-driven development with a machine learning classifier. *Software: Practice and Experience*, 52(11):2439–2455, 2022.

[SLR58] A.C. Marcén, F. Pérez, and C. Cetina. Ontological evolutionary encoding to bridge machine learning and conceptual models: Approach and industrial evaluation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10650 LNCS:491–505, 2017.

[SLR59] S. Martínez, M. Wimmer, and J. Cabot. Efficient plagiarism detection for software modeling assignments. *Computer Science Education*, 30(2):187–215, 2020.

[SLR60] V. Moreno, G. Génova, M. Alejandres, and A. Fraga. Automatic classification of web images as UML static diagrams using machine learning techniques. *Applied Sciences (Switzerland)*, 10(7), 2020.

[SLR61] P.T. Nguyen, J. Di Rocco, D. Di Ruscio, A. Pierantonio, and L. Iovino. Automated classification of metamodel repositories: A machine learning approach. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 272–282, 2019.

[SLR62] P.T. Nguyen, D. Di Ruscio, A. Pierantonio, J. Di Rocco, and L. Iovino. Convolutional neural networks for enhanced classification mechanisms of metamodels. *Journal of Systems and Software*, 172, 2021.

[SLR63] P.T. Nguyen, J. Di Rocco, L. Iovino, D. Di Ruscio, and A. Pierantonio. Evaluation of a machine learning classifier for metamodels. *Software and Systems Modeling*, 20(6):1797–1821, 2021.

[SLR64] M.H. Osman, M.R.V. Chaudron, and P. Van Der Putten. An analysis of machine learning algorithms for condensing reverse engineered class diagrams. pages 140–149, 2013.

[SLR65] M.H. Osman, M.R.V. Chaudron, and P. Van Der Putten. Interactive scalable abstraction of reverse engineered UML class diagrams. volume 1, pages 159–166, 2014.

[SLR66] M.H. Osman, M.R.V. Chaudron, P. Van Der Putten, and T. Ho-Quang. Condensing reverse engineered class diagrams through class name based abstraction. pages 158–163, 2014.

[SLR67] M.H. Osman, T. Ho-Quang, and M.R.V. Chaudron. An automated approach for classifying reverse-engineered and forward-engineered UML class diagrams. pages 396–399, 2018.

[SLR68] M.S. Osman, N.Z. Alabwaini, T.B. Jaber, and T. Alrawashdeh. Generate use case from the requirements written in a natural language using machine learning. pages 748–751, 2019.

[SLR69] A. Rago, C. Marcos, and J.A. Diaz-Pace. Identifying duplicate functionality in textual use cases by aligning semantic actions. *Software and Systems Modeling*, 15(2):579–603, 2016.

[SLR70] Abbas Rahimi, Massimo Tisi, Shekoufeh Kolahdouz Rahimi, and Luca Berardinelli. Towards generating structurally realistic models by generative adversarial networks. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 597–604. IEEE, 2023.

[SLR71] G.J. Ramackers, P.P. Griffioen, M.B.J. Schouten, and M.R.V. Chaudron. From prose to prototype: Synthesising executable UML models from natural language. pages 380–389, 2021.

[SLR72] Y. Rigou and I. Khriss. A deep learning approach to UML class diagrams discovery from textual specifications of software systems. In *Proceedings of SAI Intelligent Systems Conference*, pages 706–725. Springer, 2022.

[SLR73] R. Rubei, J.D. Rocco, D.D. Ruscio, P.T. Nguyen, and A. Pierantonio. A lightweight approach for the automated classification and clustering of metamodels. pages 477–482, 2021.

[SLR74] A. Sadovykh, G. Widforss, D. Truscan, E.P. Enoiu, W. Mallouli, R. Iglesias, A. Bagnto, and O. Hendel. VeriDevOps: Automated protection and prevention to meet security requirements in DevOps. volume 2021-February, pages 1330–1333, 2021.

[SLR75] R. Saini, G. Mussbacher, J.L.C. Guo, and J. Kienzle. Teaching modelling literacy: An artificial intelligence approach. pages 714–719, 2019.

[SLR76] R. Saini, G. Mussbacher, J.L.C. Guo, and J. Kienzle. A neural network based approach to domain modelling relationships and patterns recognition. pages 78–82, 2020.

[SLR77] R. Saini, G. Mussbacher, J.L.C. Guo, and J. Kienzle. Towards queryable and traceable domain models. volume 2020-August, pages 334–339, 2020.

[SLR78] R. Saini, G. Mussbacher, J.L.C. Guo, and J. Kienzle. DoMoBOT: An AI-empowered bot for automated and interactive domain modelling. In *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 595–599, 2021.

[SLR79] R. Saini, G. Mussbacher, J.L.C. Guo, and J. Kienzle. Machine learning-based incremental learning in interactive domain modelling. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems*, pages 176–186, 2022.

[SLR80] A. Sajji, Y. Rhazali, and Y. Hadi. A methodology of automatic class diagrams generation from source code using model-driven architecture and machine learning to achieve energy efficiency. In *E3S Web of Conferences*, volume 412, page 01002. EDP Sciences, 2023.

[SLR81] M.B.J. Schouten, G.J. Ramackers, and S. Verberne. Preprocessing requirements documents for automatic UML modelling. In *International Conference on Applications of Natural Language to Information Systems*, pages 184–196. Springer, 2022.

[SLR82] M. Sharbaf, B. Zamani, and G. Sunyé. Automatic resolution of model merging conflicts using quality-based reinforcement learning. *Journal of Computer Languages*, 71:101123, 2022.

[SLR83] N. Sharma and P. Yalla. A hybrid weighted probabilistic based source code graph clustering algorithm for class diagram and sequence diagram visualization. *International Journal of Scientific and Technology Research*, 9(4):3142–3158, 2020.

[SLR84] S. Shcherban, P. Liang, Z. Li, and C. Yang. Multiclass classification of four types of UML diagrams from images using deep learning. volume 2021-July, pages 57–62, 2021.

[SLR85] S. Shcherban, P. Liang, Z. Li, and C. Yang. Multiclass classification of UML diagrams from images using deep learning. *International Journal of Software Engineering and Knowledge Engineering*, 31(11-12):1683–1698, 2021.

[SLR86] B. Kaur Sidhu, K. Singh, and N. Sharma. A machine learning approach to software model refactoring. *International Journal of Computers and Applications*, 44(2):166–177, 2022.

[SLR87] Matthew Stephan. Towards a cognizant virtual software modeling assistant using model clones. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 21–24. IEEE, 2019.

[SLR88] D.R. Stikkolorum, P. Van Der Putten, C. Sperandio, and M.R.V. Chaudron. Towards automated grading of UML class diagrams with machine learning. volume 2491, 2019.

[SLR89] J.F. Tavares, Y.M.G. Costa, and T.E. Colanzi. Classification of UML diagrams to support software engineering education. pages 102–107, 2021.

[SLR90] K.V. Vineetha and P. Samuel. A multinomial naïve bayes classifier for identifying actors and use cases from software requirement specification documents. In *2022 2nd International Conference on Intelligent Technologies (CONIT)*, pages 1–5. IEEE, 2022.

[SLR91] F. Wang. UML diagram classification model based on convolution neural network. *Optik*, page 170463, 2022.

[SLR92] Kai Wang, Wei Liu, Yongan Mu, and Sheng Gao. Automatic extraction of sequence diagram semantic information. In *2023 5th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pages 315–318. IEEE, 2023.

[SLR93] H.G. Woo and W.N. Robinson. Reuse of scenario specifications using an automated relational learner: A lightweight approach. In *Proceedings IEEE Joint International Conference on Requirements Engineering*, pages 173–180. IEEE, 2002.

[SLR94] S. Yang and H. Sahraoui. Towards automatically extracting UML class diagrams from natural language specifications. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, pages 396–403, 2022.

[SLR95] X. Zhang, H. Washizaki, N. Yoshioka, and Y. Fukazawa. Detecting design patterns in UML class diagram images using deep learning. In *2022 IEEE/ACIS 23rd International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pages 27–32. IEEE, 2022.

[SLR96] R. Zhu, W. Li, and C. Jin. TAG: UML activity diagram deeply supervised generation from business textural specification. In *2023 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 956–961. IEEE, 2023.

[SLR97] A. Zolotas, N. Matragkas, S. Devlin, D.S. Kolovos, and R.F. Paige. Type inference in flexible model-driven engineering. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9153:75–91, 2015.

[SLR98] A. Zolotas, N. Matragkas, S. Devlin, D.S. Kolovos, and R.F. Paige. Type inference in flexible model-driven engineering using classification algorithms. *Software and Systems Modeling*, 18(1):345–366, 2019.