Combining Machine Learning and Semantic Web: A Systematic Mapping Study

ANNA BREIT, Semantic Web Company
LAURA WALTERSDORFER, TU Wien
FAJAR J. EKAPUTRA, Vienna University of Economics and Business (WU) and TU Wien
MARTA SABOU, Vienna University of Economics and Business (WU)
ANDREAS EKELHART, University of Vienna and SBA Research
ANDREEA IANA, HEIKO PAULHEIM, and JAN PORTISCH, University of Mannheim
ARTEM REVENKO, Semantic Web Company
ANNETTE TEN TEIJE and FRANK VAN HARMELEN, Vrije Universiteit (VU) Amsterdam

In line with the general trend in artificial intelligence research to create intelligent systems that combine learning and symbolic components, a new sub-area has emerged that focuses on combining Machine Learning components with techniques developed by the Semantic Web community—Semantic Web Machine Learning (SWeML). Due to its rapid growth and impact on several communities in thepast two decades, there is a need to better understand the space of these SWeML Systems, their characteristics, and trends. Yet, surveys that adopt principled and unbiased approaches are missing. To fill this gap, we performed a systematic study and analyzed nearly 500 papers published in the past decade in this area, where we focused on evaluating architectural and application-specific features. Our analysis identified a rapidly growing interest in SWeML Systems, with a high impact on several application domains and tasks. Catalysts for this rapid growth are the increased application of deep learning and knowledge graph technologies. By leveraging the in-depth understanding of this area acquired through this study, a further key contribution of this article is a classification system for SWeML Systems that we publish as ontology.

This work was supported in part by the research project OBARIS, which received funding from the Austrian Research Promotion Agency (FFG) under grant 877389. SBA Research (SBA-K1) is a COMET Centre within the framework of COMET—Competence Centers for Excellent Technologies Programme and funded by BMK, BMDW, and the federal state of Vienna; COMET is managed by FFG. Moreover, this work was supported by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital and Economic Affairs, and the National Foundation for Research, Technology and Development. M. Sabou was funded through the FWF HOnEst project (V 754-N).
Authors' addresses: A. Breit and A. Revenko, Semantic Web Company, Neubaugasse 1/8, 1070 Vienna, Austria; emails: {anna.breit, artem.revenko}@semantic-web.com; L. Waltersdorfer, TU Wien, Favoritenstrasse 9-11/194-01, 1040 Vienna, Austria; email: laura.waltersdorfer@tuwien.ac.at; F. J. Ekaputra, Vienna University of Economics and Business (WU), Institute for Data Process, and Knowledge Management (DPKM). Building D2/C, 2nd floor, Welthandelsplatz 1, 1020 Vienna, Austria; email: fajar.ekaputra@wu.ac.at; M. Sabou, University of Vienna, Kolingasse 14-16, 5. OG, 1090 Vienna, Austria; email: marta.sabou@wu.ac.at; A. Ekelhart, University of Vienna and SBA Research, Floragasse 7, 1040 Vienna, Austria; email: andreas.ekelhart@univie.ac.at; A. Iana, H. Paulheim, and J. Portisch, University of Mannheim, L 1, 1, 68131 Mannheim, Germany; emails: andreea.iana@uni-mannheim.de, {heiko, jan}@informatik.uni-mannheim.de; A. ten Teije and F. van Harmelen, Vrije Universiteit (VU) Amsterdam, Dept. of Computer Science, De Boelelaan 1111, 1085HV Amsterdam, Netherlands; emails: {annette.ten.teije, frank.van.harmelen}@vu.nl.
Author's current addresses: J. Portisch, SAP SE, Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany; email: jan.portisch@sap.com.

0360-0300/2023/07-ART313
https://doi.org/10.1145/3586163

CCS Concepts: • **Information systems → Semantic web description languages**; • **Computing methodologies → Knowledge representation and reasoning**; **Machine learning;**

Additional Key Words and Phrases: Semantic Web, Machine Learning, Artificial Intelligence, knowledge graph, Knowledge Representation and Reasoning, neuro-symbolic integration, Systematic Mapping Study

ACM Reference format:
Anna Breit, Laura Waltersdorfer, Fajar J. Ekaputra, Marta Sabou, Andreas Ekelhart, Andreea Iana, Heiko Paulheim, Jan Portisch, Artem Revenko, Annette ten Teije, and Frank van Harmelen. 2023. Combining Machine Learning and Semantic Web: A Systematic Mapping Study. *ACM Comput. Surv.* 55, 14s, Article 313 (July 2023), 41 pages.
https://doi.org/10.1145/3586163

1 INTRODUCTION

For a system to be perceived as "intelligent," it has to fulfill certain properties: it needs to be able to adapt and react to unknown situations, and it needs to have some understanding of the world in which it acts, subjected to constant refinement over time while it obtains access to new information. Although **Artificial Intelligence (AI)** and human intelligence are unarguably different, and especially the latter is still not fully understood, researchers have drawn parallels between the two in the past. A recent AAAI paper [6] relates the building blocks of AI to Daniel Kahneman's theory of human intelligence, which is divided into an (intuitive) system 1 and a (rational) system 2 [10]. The authors state that system 1 is comparable to **Machine Learning (ML)**, whereas system 2 rather resembles **Knowledge Representation and Reasoning (KR)**. They further argue that AI, just like human intelligence, needs the combination of both, also called *neuro-symbolic AI*.

This symbiotic use of ML and KR techniques is a strongly emerging trend in AI. Indeed, recent years have seen an increased interest and fast-paced developments in techniques that make use of this combination to build intelligent systems in the vein of neuro-symbolic AI. At the same time, the **Semantic Web (SW)** research community has popularized knowledge representation techniques and resources in the past two decades [15], leading to a great interest in and uptake of SW resources such as knowledge graphs, ontologies, thesauri, and linked datasets outside of the SW research community [17]. These two trends have led to the development of systems that rely on both SW resources and ML components, known as **Semantic Web Machine Learning (SWeML)** Systems.

This research area of SWeML has gained a lot of traction in the past few years, as shown in a rapidly growing number of publications in different outlets, as well as SWeML techniques being employed to solve problems in various domains. At the same time, this growth poses two main challenges that threaten to hamper further development of the field.

First, *keeping up with the main trends in the field* has become unfeasible, not only because of its fast pace and a large volume of published papers but also because papers require understanding techniques from the two diverse research sub-areas of AI. In an attempt to address this challenge, several works aimed to provide overviews of SWeML Systems and related systems (see Table 1). However, reviewing those, we conclude that existing work either (1) focuses rather on a wider or related field [34] or, on the contrary, (2) is scoped around a very specific sub-field of SWeML [25, 26, 29]. Additinally, none of the reviewed surveys adopts a principled and reproducible methodology that would guarantee unbiased and representative data collection. We therefore conclude that there is a need for a survey that adopts a solid review methodology to complement current insights with evidence-based findings.

The second challenge, which amplifies the first one, is the *lack of a standardized way to report SWeML Systems* that hampers understanding all key aspects of these systems. On the one hand, authors of SWeML Systems would benefit from a structured way to describe their system and

its key characteristics. Readers, on the other hand, would benefit from a structured way of interpreting such systems. This would not only facilitate the understandability for those coming from other communities but also improve the comparability of different systems. An early work in this direction was proposed by Van Harmelen and ten Teije [34] by introducing patterns for representing hybrid AI systems in terms of their components and information flows with the aim to facilitate a more schematic representation of the system. Although these patterns were derived from a large number of papers, there is currently no insight into their adoption in the field (e.g., about the completeness of the introduced system patterns) or their usage frequency.

To address the preceding challenges, in this article we investigate the following main research questions:

- What are the state of the art and trends related to systems that combine SW and ML components?
- How can these systems be classified into a systematic taxonomy?

To that end, in contrast to previous work, we perform a **Systematic Mapping Study (SMS)** [20] of the SWeML Systems field. SMS is an established method in evidence-based research because (1) it follows a well-defined paper selection process to identify a *representative set* of primary studies *reducing selection bias* in comparison to ad hoc study selection, and (2) it adopts a standard, well-documented process allowing for the study to be *reproduced*.

Based on this methodology, we provide two main contributions:

- *A trends landscape*, to capture the tendencies of SWeML Systems with respect to the level of adoption, maturity, and reproducibility, with an in-depth focus on structural aspects, their processing flows, and the characteristics of their ML and SW components, derived from a systematic survey.
- *A classification system for SWeML Systems* that can be used as a template for analyzing existing systems and describing new ones. This can be seen as a controlled vocabulary for the different building blocks of those systems. A key aspect of this classification system is manifested as a framework for documenting and classifying processing flows in SWeML Systems.

With these contributions, we aim to address a large and diverse audience and facilitate their understanding of this rapidly emerging area both within the scope of this article and beyond. To ensure *transparency and reproducibility*, we share our research material including the study protocol, the list of collected papers, and additional analysis.[1] Furthermore, we share the classification system in the form of an ontology to enable the creation of human-understandable yet machine-actionable documentation.

This article is structured as follows. Section 2 defines and positions SWeML Systems, whereas Section 3 reviews existing surveys and classification systems in related and neighboring fields. Section 4 describes the survey methodology including a refinement of the overall research question of this survey into more detailed ones, Section 5 presents the findings from the survey, and Section 6 derives a new classification system for SWeML Systems. We conclude with a discussion of our main findings and an outlook on future work.

2 SEMANTIC WEB MACHINE LEARNING SYSTEMS

2.1 Definition

SWeML Systems are the result of combining SW technologies and an inductive model. More precisely, they describe a *system* that makes use of an *SW knowledge structure* as well as an *ML sub-system* to solve a specific task.

[1] https://swemls.github.io/swemls/.

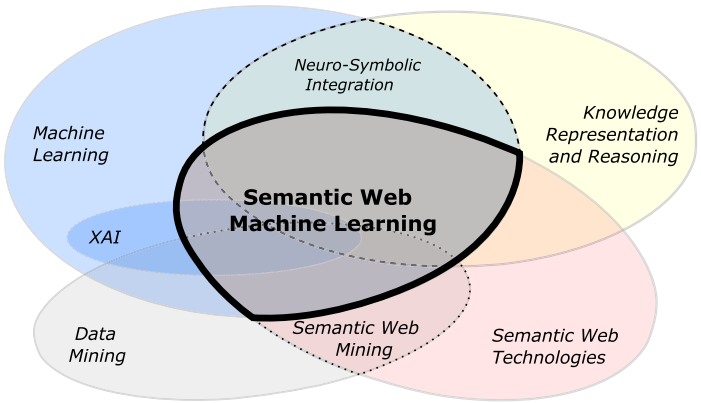

Fig. 1. Relation of different research areas around SWeML.

For the purposes of this survey, we define such *SW knowledge structures* as a symbolic representation of a conceptual domain model and data complying with such domain models. These resources can have varying levels of formalization ranging from formal logical foundations (in particular, description logic) to lightweight semantic structures. Examples include vocabularies, taxonomies, ontologies, linked datasets, and knowledge graphs. The *ML sub-system* consists of an inductive model that can generalize over a given set of examples. These models include rule learning systems, traditional ML models such as support vector machines, random forests, or multi-layer perceptrons, as well as more recent deep learning models. SWeML *Systems* are tangible systems with a software implementation. These implementations might be of different maturity, ranging from prototypes to enterprise-ready systems. However, all of these systems aim at solving specific tasks as opposed to being generic use components such as libraries and conceptual frameworks.

Although SWeML Systems are required to include the usage of both a *semantic module* and an *ML module*, they are not restricted in the number of these modules, nor in the incorporation of components that do not fall under the definition of either of these modules. This means that SW knowledge structures as well as ML models can be used in various parts of the system—either in a pure form or complemented with other models or knowledge structures—but they are not required to be applied in all of the parts. Furthermore, no assumptions about the patterns of interactions between the semantic and the ML module are made, yielding a wide variety of possible design patterns for SWeML Systems.

2.2 Background and Positioning

To further deepen the understanding and definition of the SWeML field, it is helpful to draw connections to related fields. An overview of the connection of these research fields is shown in Figure 1.

Neuro-Symbolic Integration. Recent developments in AI mostly rely on an underlying ML system; however, a more traditional approach is based on symbolic KR. **Neuro-Symbolic Systems (NeSy)** aim to integrate both these approaches to combine and exploit the advantages of an inductive and deductive system [12]. As KR and SW technologies are tightly connected, SWeML is significantly overlapping with the area of neuro-symbolic integration, despite having a different focus: SWeML Systems may incorporate deductive reasoners, and NeSy frequently work with SW data representations.

Explainable Artificial Intelligence. **Explainable Artificial Intelligence (XAI)** systems aim to increase the interpretability and comprehensibility of AI systems [4], where *interpretable* refers to

the ability to understand how inputs are processed on a systems level, whereas comprehensible systems provide the user with symbols that enable them to draw conclusions on how properties of the input influence the output [11]. A SWeML System can improve the interpretability and comprehensibility of the ML sub-system through its incorporation of a semantic symbolic subsystem. Such explainable SWeML Systems have been used in different domains to solve a wide variety of tasks [29]. However, not every SWeML System results in an XAI system.

SW Mining. SW mining addresses the combination of Web mining and SW technologies. Web mining describes the application of data mining techniques to Web resources to extract useful patterns and information [32]. The patterns of interaction between Web mining and SW are diverse—that is, data mining can be used to construct SW resources, or SW data can be exploited for Web mining [30]. Even though there are parallels between a SWeML System and a SW mining system, they have a significantly different focus: SW mining systems aim at Web-based data, whereas SWeML Systems do not restrict the type of their data sources. On the other side, data mining techniques used in Web mining do not necessarily have to be based on inductive systems.

3 RELATED WORK

3.1 Related Surveys

The vision of combining symbolic knowledge and learning has a long history [12] with intensified research activities over the past 5 years [28]. As a result, a number of survey papers have addressed the areas of SWeML and neuro-symbolic integration as synthesized in Table 1 in terms of the *system type* they cover (e.g., *NeSy*, *SWeML System*) and their *paper selection* procedure.

Starting with the works investigating *NeSy*, Besold et al. [5] survey learning and reasoning approaches from a holistic perspective. The paper investigates the intersection of computer science, cognitive science, and neuroscience on a general level. Coming from the ML perspective, Von Rueden et al. [35] coin the term *informed ML*, which is one approach to NeSy. They propose a taxonomy of methods to integrate knowledge into learning systems [35]. Although knowledge graphs and other approaches are mentioned, SW data and symbolic representation methods are not the key topics.

Two surveys on design patterns for NeSy propose a taxonomically organized vocabulary to describe both processes and data structures [33, 34]. However, their paper selection is ad hoc, without an attempt to be exhaustive. Hitzler et al. [16] provide an initial overview of neuro-symbolic AI for SW and discuss their mutual benefits. Examples include deductive reasoning and knowledge graph embeddings [16]. This paper offers first insights into techniques and examples but does not discuss common architectures or frequencies of used models. Recently, Sarker et al. [28] surveyed 43 papers from well-established AI conferences to characterize neuro-symbolic AI using two different taxonomies. Although they offer a first perspective into trends, details concerning dataflows, used models, and architectures are not discussed.

Finally, Seeliger et al. [29] and D'Amato [8] focus on SWeML Systems, although they do not define SWeML Systems as a concept. Seeliger et al. [29] investigate in a systematic literature review how to make opaque ML algorithms explainable through SW technologies, exploring which combinations of SW and ML techniques are used to obtain explanations, in which domains they are mainly used, and how these explanations are evaluated. From an SW perspective, D'Amato [8] describes research directions for incorporating ML techniques into symbolic approaches. Examples include instance retrieval, concept learning, knowledge completion, or learning disjointedness, but more of an overview of these techniques is offered than a deep analysis.

We conclude that existing surveys mostly focus on the broader category of NeSy systems and only a few target SWeML Systems. Additionally, with exception of the work of Seeliger et al. [29],

Table 1. Related Survey Papers Focusing on the Intersection of SW and ML

Ref.	Authors	Venue	Type	Year	Paper Selection	Contribution
[5]	Besold et al.	Neuro-Symbolic Artificial Intelligence: The State of the Art	NeSy	2021	Custom	*Overview of NeSy from different perspectives*: Computer science, cognitive science, cognitive neuroscience
[35]	Von Rueden et al.	IEEE Transactions on Knowledge & Data Engineering	NeSy	2021	Custom	*Overview of informed ML*: Taxonomy, overview of knowledge types
[33, 34]	Van Bekkum et al., Van Harmelen and ten Teije	Applied Intelligence, Journal of Web Engineering	NeSy	2019, 2021	Custom	*Design patterns for hybrid AI*: Taxonomy to describe processes and data structures, case study
[16]	Hitzler et al.	Semantic Web	NeSy	2020	Custom (vision paper)	*Overview of NeSy for SW*: Knowledge graph embeddings, explainable deep learning, deductive reasoning
[28]	Sarker et al.	arXiv	NeSy	2021	Survey papers collected from NeurIPS, ICML, AAAI, ICLR, IJCAI	*Overview of NeSy*: Grouping into a taxonomy [19] and into dimensions [3]
[29]	Seeliger et al.	SemEx ISWC	XAI SWeML System	2019	Systematic literature review: *(Q1)* "deep learning" OR "data mining"; *(Q2)* "explanation*" OR "interpret*" OR "transparent*"; *(Q3)* "Semantic Web" OR "ontolog*" OR "background knowledge" OR "knowledge graph*"	*Overview of*: SW for XAI, application domains and tasks important to this research field, forms of explanations, evaluation
[8]	D'Amato	Semantic Web	SW SWeML System	2020	Custom	*Overview of ML methods for SW*: Probabilistic latent variable models, embedding models, vector space embeddings

all surveys adopt a custom approach to collecting the surveyed papers. Therefore, there is a need for a systematic survey that adopts a solid review methodology and focuses on SWeML Systems, which we aim to address with this article.

3.2 Existing Classification Systems

Several works aim to characterize NeSy: Bader and Hitzler [3] made an early attempt to propose eight dimensions for classification purposes. More recently, Van Harmelen and ten Teije [34] proposed a set of 13 design patterns, similar to design patterns in software engineering. This taxonomy is extended with processes and models in the work of Van Bekkum et al. [33]. Kautz [19] introduced a neuro-symbolic taxonomy of six different types of systems. Although the goal is similar, his taxonomy does not reflect the internal architectures of the investigated systems. The taxonomies proposed by Sarker et al. [28] and Von Reuden et al. [35] are more fine-grained but with less focus on how to combine different system architectures.

For SWeML Systems, no works that target their classification could be found. To fill this gap, we are proposing a classification system that uses ideas from the taxonomically organized vocabulary to describe both processes and data structures for NeSy in the work of Van Harmelen and ten Tieje [34]. However, our proposed classification system (1) takes a more coarse-grained system-level view as opposed to the fine-grained view in characterizing data structures/processes;

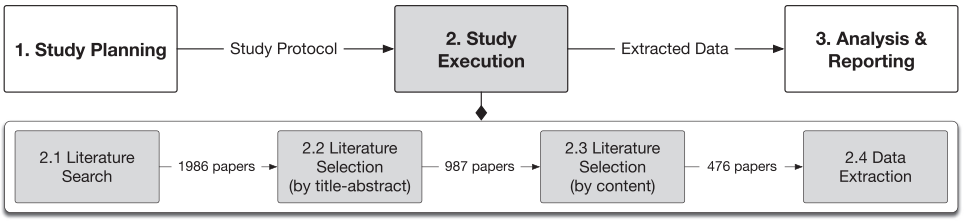

Fig. 2. Overview of the SMS process.

(2) focuses on a particular type of neuro-symbolic AI systems, namely SWeML Systems; and (3) has been derived as part of the SMS from a large number of papers.

4 METHODOLOGY

To gain an overview of existing research that falls under the term of SWeML Systems, we conducted an SMS [20], which is well suited to structure broad research areas. A more detailed explanation of the methodology (e.g., details on keyword selection and selection criteria) can be found in the study protocol.[2] The SMS consists of three consecutive phases (cf. Figure 2) as follows:

(1) *Study Planning*, where we design and develop the *Study Protocol*, as detailed in (Section 4.1);
(2) *Study Execution*, consisting of *(2.1) Literature Search*, *(2.2–2.3) Literature Selection*, and *(2.4) Data Extraction*, which rely on details specified in the *Study Protocol* (Section 4.2); and
(3) *Analysis & Reporting*, focusing on the analysis of the extracted data and the reporting thereof (Section 5).

4.1 Study Planning

As the first phase of the study, planning focuses on *scoping the study* and, accordingly, *proposing the methodology* for each step of the study as documented in the study protocol. Study scoping includes positioning the planned work in the context of related research areas and related work in terms of similar surveys. This forms a basis for deriving pertinent research questions (Section 4.1.1), which are then translated into appropriate search queries (Section 4.1.2), a number of paper selection criteria (Section 4.1.3) used to identify relevant papers, and a data extraction form that facilitates the objective and unbiased extraction of data. The methodological details captured in the study protocol aim to make the study process transparent and reproducible.

4.1.1 Detailed Research Questions. We refine our two overall research questions (announced in Section 1) into a number of more detailed research questions, which (1) help identify emerging trends in the area and (2) provide insights into the characteristics of SWeML Systems for deriving a classification scheme thereof:

RQ1 *Bibliographic characteristics:* How are the publications temporally and geographically distributed? How are the systems positioned, and which keywords are used to describe them?
RQ2 *System architecture*: What processing patterns are used in terms of inputs/outputs and the order of processing units?
RQ3 *Application areas*: What kind of tasks are solved (e.g., text analysis)? In which domains are SWeML Systems applied (e.g., life sciences)?

[2] https://swemls.github.io/swemls/.

Table 2. Sub-Queries for the Search Query $Q = Q1 \cap Q2 \cap Q3$

Sub-Query	Search Strings
Q1 (SW module)	knowledge graph, linked data, semantic web, ontolog*, RDF, OWL, SPARQL, SHACL
Q2 (ML module)	deep learning, neural network, embedding, representation learning, feature learning, language model, language representation model, rule mining, rule learning, rule induction, genetic programming, genetic algorithm, kernel method
Q3 (system)	Natural Language Processing, Computer Vision, Information Retrieval, Data Mining, Information integration, Knowledge management, Pattern recognition, Speech recognition

Each sub-query consists of a disjunction (OR) of search strings.

RQ4 *Characteristics of the ML module*: What ML models are incorporated (e.g., SVM)? Which ML components can be identified (e.g., attention)? What training type(s) is used during the system training phase?

RQ5 *Characteristics of the SW module*: What type of SW structure is used (e.g., taxonomy)? What is the degree of semantic exploitation? What are the size and the formalism of the resources? Does the system integrate semantic processing modules (i.e., KR)?

RQ6 *Maturity, transparency, and auditability*: What is the level of maturity of the systems? How transparent are the systems in terms of sharing source code, details of infrastructure, and evaluation setup? Does the system have a provenance-capturing mechanism?

4.1.2 *Digital Libraries and Search Queries.*

Digital Libraries. We performed a query-based search in the following digital libraries to retrieve important conference and journals papers, as they are referred to as good sources for software engineering publications [7, 20]: (i) Web of Science, (ii) ACM Digital Library, (iii) IEEE Xplore, and (iv) Scopus.[3]

Search Query. The search query was derived from the study research questions and iteratively refined to obtain a high number of relevant papers while keeping the number of retrieved papers manageable. The query consists of three sub-queries targeting the *SW module* (Q1), the *ML module* (Q2), and the *system* aspect (Q3) of SWeML Systems, respectively. The search strings contained in these queries are presented in Table 2. Each sub-query consists of a union of the listed search terms; the final query used for the study search is an intersection of the three sub-queries. The collection of the search terms for the sub-queries followed a systematic methodology, which is described in more detail in Appendix A.1.

4.1.3 *Study Selection Criteria.*

We have selected eight study selection criteria. For each criterion (C), an *inclusion criterion* (*IC*) and a complimentary *exclusion criterion* (*EC*) are given. This improves the specificity of the criteria. Inclusion criteria 1 through 5 concern metadata of the publications, such as publication date (2010–2020), language (English), publication type (peer reviewed), accessibility (accessible to authors), and duplicates (latest version). C6 and C7 refer to the SWeML Systems definition: whether described systems have an interconnection between the SW and ML component (C6), and whether the system solves a task (C7). C8 filters out papers with low English and/or scientific quality that cannot be fully understood.

[3](i) http://www.webofknowledge.com/, (ii) https://dl.acm.org/, (iii) https://ieeexplore.ieee.org/, (iv) https://www.scopus.com/.

4.2 Study Execution

Based on the study protocol, the study is executed through the steps described in the next sections.

Literature Search. The execution of search queries in the four digital libraries returned 2,865 papers.[4] After merging the four result sets, 1,986 papers remained (cf. box 2.1 of Figure 2). We used a combination of automatic merging of bibtex entries,[5] as well as manual checking to ensure the correctness of the merged results.

Literature Selection. Literature selection includes two separate selection steps. The first step focuses on metadata, titles, and abstracts (cf. box 2.2 of Figure 2). We divided the retrieved papers into 10 batches of 200 papers each and assigned two researchers to each batch. The first researcher decided on inclusion or exclusion considering the criteria C6 and C7. The second assignee checked unclear cases plus 10% of the decisions of the first researcher. This step reduced the number of papers to 987 papers. In the second step (cf. Box 2.3 of Figure 2), all papers were investigated more thoroughly based on their content and in terms of the study selection criteria C1 through C8, leading to 476 papers selected for data extraction.

Data Extraction. This step (cf. box 2.4 of Figure 2) was conducted with the help of a shared data extraction form that defines how and which data is collected from papers to answer the study research questions (Section 4.1.1). The form was prepared *prior* to study execution to reduce researcher bias and allow multiple researchers to extract data objectively [20]. The detailed data extraction form is available in the study protocol.

5 DATA ANALYSIS

Because our study criteria lead to 476 publications that are included in the data analysis, it is not possible to perform a publication-wise analysis. We will therefore limit ourselves to meta-analysis. However, to provide a better understanding of the concrete outcomes, we will exemplify the extracted values with seven example papers included in the analysis: PUB1 [13], PUB2 [39], PUB3 [2], PUB4 [24], PUB5 [38], PUB6 [9], and PUB7 [31].

5.1 RQ1 Bibliographic Characteristics

5.1.1 Temporal Distribution. Figure 3 shows the non-exclusive distribution per year of the 476 publications.[6] We observed two trends in the publication count over the years. Starting from 2016, there is a surge in the number of papers in all digital libraries. Furthermore, a large portion of the selected papers were retrieved from Scopus. Between 2010 and 2016, the published papers account yearly for less than 5% of the total number of selected publications. From 2016 onward, 15% to 20% were retrieved yearly, increasing in 2019 and 2020 to more than 35% of all publications selected for data extraction. An important aspect we take into account in the remainder of the data analysis is that the decrease from 2019 might be because the set of papers from 2020 is incomplete.[7]

5.1.2 Thematic Distribution. For identifying the positioning and focus of a paper, we concentrate on author-defined keywords. If no keywords were provided in the paper (e.g., PUB4

[4]The literature search was executed on November 2, 2020.
[5]Using bibliographic data management software from Mendeley (https://www.mendeley.com/) and Zotero (https://www.zotero.org/).
[6]Note for Figure 3 that several papers were counted multiple times in the graph due to being available in more than one digital library.
[7]The search for papers was performed on November 2, 2020, and many digital libraries have delays of several months for indexing publications, whereas some relevant conferences only take place in December.

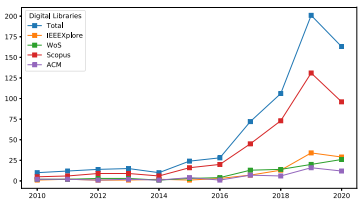

Fig. 3. Number of selected publications in individual digital libraries per year (non-exclusive).

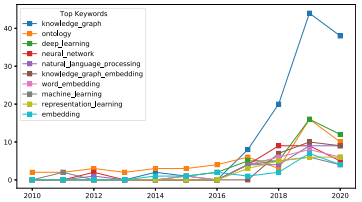

Fig. 4. Popularity of the top 10 keywords.

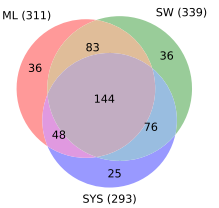

Fig. 5. Thematic distribution of the selected papers based on the author keywords.

and PUB5), they were automatically generated by extracting the terms with the highest TF-IDF score [22] from the title and abstract of the paper. To increase quality, only those generated keywords were considered that also appeared in the global list of all author-provided keywords. With this procedure, we, for example, generated for PUB5 (titled *Using Distributional Semantics for Automatic Taxonomy Induction*) the keywords "taxonomy," "textual entailment," and "natural language processing."

Keywords. Figure 4 shows the evolution of the top 10 keywords over the years. Two of the top 3 keywords describe types of semantic resources: (1) *knowledge graph* (110 papers) and (2) *ontology* (54 papers), followed by (3) *deep learning* (39 papers). From a semantics perspective, until 2016, *ontology* was used as a frequent term, whereas from 2017 on, a substantial switch toward *knowledge graphs* can be seen (for a detailed analysis of semantic resources and their types, cf. Section 5.5). *Deep learning* gained traction from 2016 on, as well as *embeddings* from 2017 on (both word and knowledge graph embeddings, cf. Section 5.4 for details on the ML components). However, for both assertions, it is important to note that the number of included papers in our study significantly increased from 2016 on with 57 papers from 2010 to 2015 compared to 419 papers from 2016 to 2020. Until 2015, the small number of selected papers and thus few keywords do not support conclusive insights for this period. As the field matures, future studies might show a drift toward more common system tasks and application areas. However, until now, mainly ML and SW components are used for the definition of systems.

Positioning. Figure 5 depicts the positioning of papers according to their specified area. To that end, we categorized keywords that appeared in at least two papers[8] into *Machine Learning* (*ML*), *Semantic Web* (*SW*), and *Systems* (*SYS*).

Each keyword can be associated with one or multiple categories—for example, *knowledge graph* is in the *SW* category, *deep learning* is in the *ML* category, and *knowledge graph embedding* is associated both with *SW* and *ML*. *Question answering* and *information retrieval* are examples of *SYS* keywords. The majority of papers fall into all three areas of *ML*, *SW*, and *SYS* (144 papers); second is the intersection between *ML* and *SW* (83 papers); and third, *SW* and *SYS* (76 papers). The *SYS* area has been assigned less often, which could be both related to our choice of system keywords, or an indication that SWeML Systems remains an evolving field with a high variety in use cases and domains, without clearly dominant tasks or system-related keywords.

5.1.3 Geographical Distribution. Figure 6 illustrates the regional distribution of the publishing institutes of the included publications. We found three major regional clusters of publishing institutions in the domain of SWeML Systems. More specifically, 43% of the surveyed papers have

[8]Twenty-eight of the 476 papers used only unique keywords and hence were not considered in this analysis.

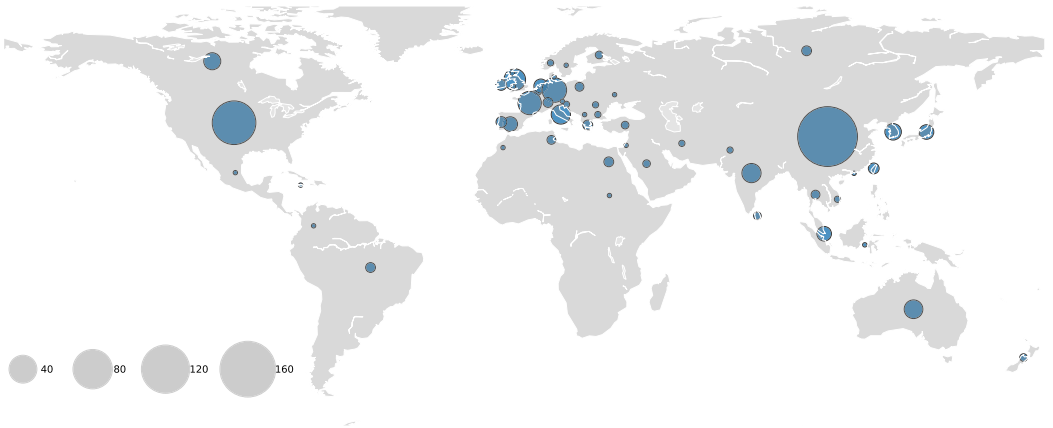

Fig. 6. Regional distribution of affiliation of paper authors. The area of the circles corresponds to the number of publications. The distribution is non-exclusive (e.g., PUB5 appears in both Germany and Pakistan). The raw data can be found in Appendix A.2.

an author affiliated with an institution in Asia, approximately 29% have one affiliated in Europe, and nearly 19% have an author based in North America. Among the Asian countries, in 71% of the cases, the author is based in China, whereas in North America, the authors are located in the United States in 86% of the cases. The geographical distribution in Europe is less skewed, with Germany, France, the United Kingdom, and Italy being the most frequent countries of affiliation of the authors, each in more than 10% of the cases. In only 1% of the cases, publications have an author from Africa or from South and Central America. Similarly, the Middle East and Oceania are also underrepresented, as in only 2.5% to 3.5% of the cases, respectively, publications have an author affiliated with an institution from one of the two regions.

5.1.4 Conclusions: RQ1. We conclude that there is a recent and accelerated growth in interest (and corresponding published papers) in the area of SWeML that is present worldwide, with a strong cluster in China and a general weak representation of the global south. Furthermore, based on their keywords, papers reporting on SWeML Systems mostly relate to the *SW* area (in particular, *ontology, knowledge graph*) or *ML* area (most frequently *deep learning*) or a combination thereof, and less to a particular domain (although *natural language processing* is a prominently used keyword).

5.2 RQ2 System Architecture

To gain deeper insights into how SW and ML modules are combined in a SWeML System, we analyze the overall processing flows and the roles these modules play. To depict the processing flows in SWeML Systems, we use a boxology notation framework to define interaction *patterns*.

5.2.1 SWeML Systems Boxology. To efficiently analyze the internals of SWeML Systems, it is necessary to abstract their processing flows under a common framework. Such a framework would not only enable comparability of system architectures but would also facilitate the common understanding of these systems, as it provides a unified and intuitive way of describing, documenting, and visualizing.

In this survey, we introduce a visual framework that focuses on depicting the flow of information through SWeML Systems. Herefore, a set of basic elements is defined, which can be combined into reusable design patterns. Our work builds on top of the boxology for NeSy introduced by Van

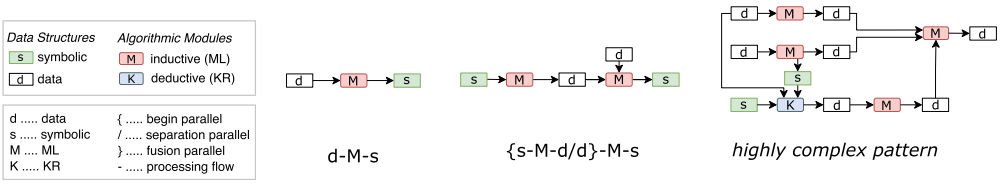

Fig. 7. Visual (top) and flat (bottom) notation of system patterns according to the SWeML Systems boxology. The patterns (from left to right) correspond to the processing workflows from PUB5, PUB1, and PUB2.

Harmelen and ten Teije [34], which proposes two base elements: algorithmic modules (i.e., objects that perform some computation) that can be of type *inductive* (*ML*) or *deductive* (*KR*), and data structures, which are the input and output to such modules that can be of *symbolic* (*sym*) (e.g., semantic entities or relations) or *non-symbolic* (*data*) nature (e.g., text, images, or embeddings) (Figure 7).

Based on our definition, each pattern that describes a SWeML System needs to incorporate at least one *ML* and one *sym* (=*Semantic Web resource*) module. Although deductive *KR* modules are not necessary for a SWeML System, their presence and therefore the documentation of their participation in a processing flow is of great interest. Each of the algorithmic modules ingests some input and produces some output, meaning that each must have at least one incoming and one outgoing data structure; the chaining of algorithmic modules without intermediate data modules is not permitted. However, an algorithmic module can consist of a combination of model parts (e.g., a Transformer model and attached classification layer can be represented using one *ML* module).

The processing pattern for a specific system can most intuitively be represented visually, as shown in Figure 8; however, to be able to also efficiently refer to them in writing, we are further introducing a flat notation. Herefore, we use the symbols `M`, `K`, `s`, and `d` for *ML* and *KR*, and *sym* and *data*, respectively. Furthermore, the flow arrows are simplified into dashes, whereas parallel sub-flows are separated by slashes and enclosed by curly brackets. After the closing curly bracket, the parallel processes fuse. We would like to point out that the used notation is prone to limitations, as highly complex patterns (e.g., those including loops) cannot be represented; however, most of the patterns, we found this notation to be useful for easy comprehensible textual reference.

Figure 7 provides an overview and some examples of processing patterns. The first depicted pattern shows a low complexity and corresponds to the processing flow in PUB5, where the authors use a textual corpus (*data*) on which they applied a distributional word embedding model (*ML*) to induce a taxonomy (*sym*). The second pattern is more complex, depicting the creation of graph embeddings (`s-M-d`) that are together with image data (`d`) fed into a CNN model (`M`) to create image classifications (`s`) from PUB1. The third illustrated highly complex pattern is derived from PUB2, where the authors build a visual question answering system: a language model creates embeddings from the question and a CNN model from the image (both `d-M-d`). Both kinds of embedding are used as input to a complex neural network to produce the answer (upper part of the diagram). Furthermore, the CNN is deployed to provide attribute and object labels `s` that are—together with the input question—used to query ConceptNet and retrieve concept descriptions (`{s/d/s}-K-d`) that are further fed to a language model to produce embeddings that serve as additional input to the complex neural network.

Although in the original boxology the introduced design patterns are based on the task of their underlying system, in this work we aim to separate these concerns. The purpose of a design pattern is to show the structural characteristics of a system only; therefore, it focuses on the input consumed and output produced by the different modules, as well as the connection between these modules to aggregate common paths rather than perfectly depicting a single system. Although

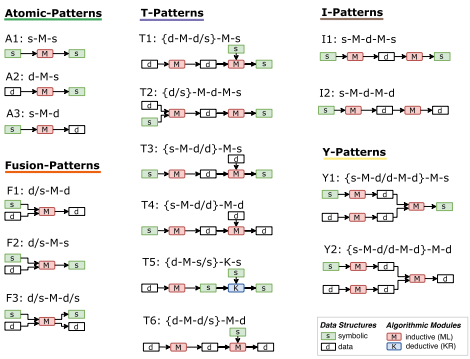

Fig. 8. For each pattern type, the most popular patterns are shown in boxology and flat notation.

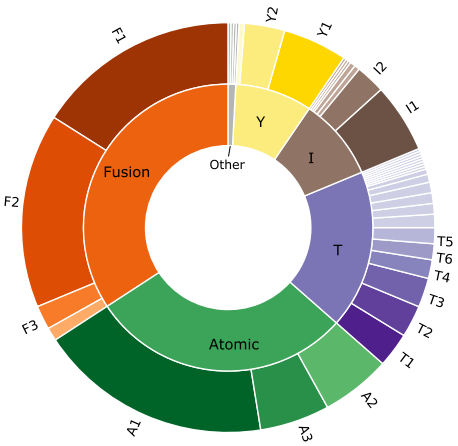

Fig. 9. Overall distribution of patterns and pattern types.

this abstraction naturally leads to the loss of some details, it facilitates the understandability of the patterns, as it reduces complexity, while providing an overview summarizing the most important processing information, and enables their aggregation into pattern types.

5.2.2 Pattern Types. We started off with the existing 11 patterns presented in the original paper.[9] During the annotation process, the annotators were tasked to reuse already discovered patterns; however, if none of the existing patterns captured the processing flow of a given paper, the annotator introduced a new pattern. This resulted in 33 new patterns, summing up to a total of 41 different processing flow patterns that we include in this analysis.[10] To allow a conclusive analysis, we further classified these patterns into pattern types. As the boxology itself focuses on the architectural properties of the processing flows, we based our classification schema on the structural characteristics of the pattern shape as a feature of its complexity, resulting in the following six types (see examples in Figure 8):

Atomic Pattern: A single algorithmic module consumes a single input.
Fusion Pattern: A single algorithmic module consumes more than one input.
I-Pattern: A chain of Atomic Patterns.
T-Pattern: A chain of Atomic and Fusion Patterns (usually, an I-Pattern with one Fusion Pattern).
Y-Pattern: Combination of two (or more) Atomic Patterns via a Fusion Pattern.
Other Pattern: Patterns that do not fall in any of the previous types. These patterns are typically quite complex; however, a further reduction would lead to loss of essential insights into the processing workflow. An example of this is the third pattern in Figure 7.

As can be seen in Figure 9, the majority of systems (more than 63%) exploit rather simple design patterns (i.e., Atomic and Fusion Patterns). The most often used pattern is `s-M-s` (A1), which is, for example, the classical pattern for link prediction in a KG based on cosine similarity of graph embeddings; however, rule-learning systems such as presented in PUB2 also use this pattern. Within

[9]The original paper introduced 15 patterns; however, patterns (1) and (2) were excluded because they do not represent a SWeML System, (10) forms a duplication of (6), and (14) does not follow our definition of processing flow.
[10]The original patterns (8), (13), and (15) were not assigned to any analyzed system and therefore not considered in the analysis.

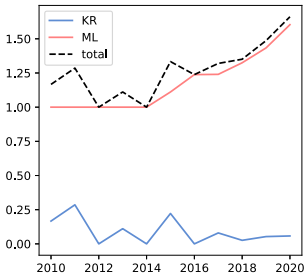

Fig. 10. Mean number of KR and ML modules in analyzed systems.

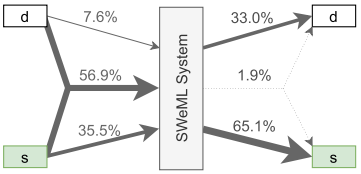

Fig. 11. Overall input types consumed and output types produced by analyzed systems.

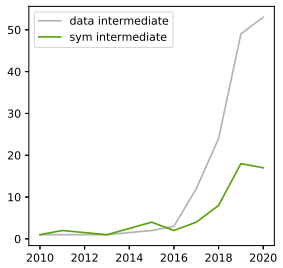

Fig. 12. Intermediate representation type of systems with at least two algorithmic modules.

the fusion patterns, `d/s-M-d` (F1) and `d/s-M-s` (F2) are equally important, showing the flexibility of a system that consumes both *sym* and *data* input.

Among the more complex patterns, T-Patterns are the most prominent, without a clear trend for a specific pattern. A different picture is drawn for I- and Y-Patterns: in I-Patterns, `s-M-d-M-s` (I1) is most prominent (using *sym* input to produce *sym* output with intermediate data representation), whereas for Y-Patterns, `{s-M-d/d-M-d}-M-s` (Y1) and `{s-M-d/d-M-d}-M-d` (Y2) represent the vast majority. These Y-Patterns include (but are not limited to) creating embeddings for both *sym* and *data* input, and combining these embeddings in a further inference step. A concrete example of such a processing flow can be seen in PUB6, where a system for the extraction of adverse drug events and related information (e.g., drugs, their attributes, and reason for administration) from unstructured medical documents is introduced: a linguistic model generates semantic word embeddings from the textual input while, at the same time, a graph embedding model calculates embeddings from an accompanying medical knowledge structure for entities identified in the text. Both these embedding types are then fed into a neural network to create the predictions.

Over time, systems tend to grow larger in terms of number of modules per pattern: in Figure 10, we observe that the number of ML modules continuously grows starting from 2014, whereas the usage of KR modules remains on a low frequency over the years.

5.2.3 Pattern Abstraction and Meta-Flow Analysis. By abstracting the patterns to the highest degree possible, an input-output-centric view can be achieved (Figure 11). From this representation, we can see that in 92.4% of the cases, *sym* input is consumed by the SWeML System (where it is used as sole input type in 35.5%), whereas *data* is taken as input in 64.5% of the systems (7.6% use data as sole input). In fact, most of the systems (56.9%) use both *sym* and *data* as input.

However, about one-third of the papers produce only *data* output compared to 65.1% producing only *sym* output (1.9% produce both). Therefore, although consuming SW resources seems to be quite essential to a SWeML System, the creation of new or extension of existing SW resources is targeted in two-thirds of the cases.

To perform a deeper analysis of how inputs and outputs are connected, and which intermediate data structures are used (Figure 12), the paths from all patterns used in the analyzed papers were aggregated into a *meta processing flow* (Figure 13), which shows the popularity of different paths. It can be seen that *sym* and *data* are most often fused in the very first step via an ML module. This processing path is taken by the very frequently used patterns `F1` and `F2`, but also by some T-patterns such as `T2`. By comparing the incoming and outcoming arrows of the second ML and KR module, we can further investigate that also for combinations of *sym* and *data* in later steps, ML modules are used more frequently than KR modules (e.g., patterns `T1, T3, T4, T6`, and `Y1-Y2` use the path via the second ML module, `T5` via the second KR module).

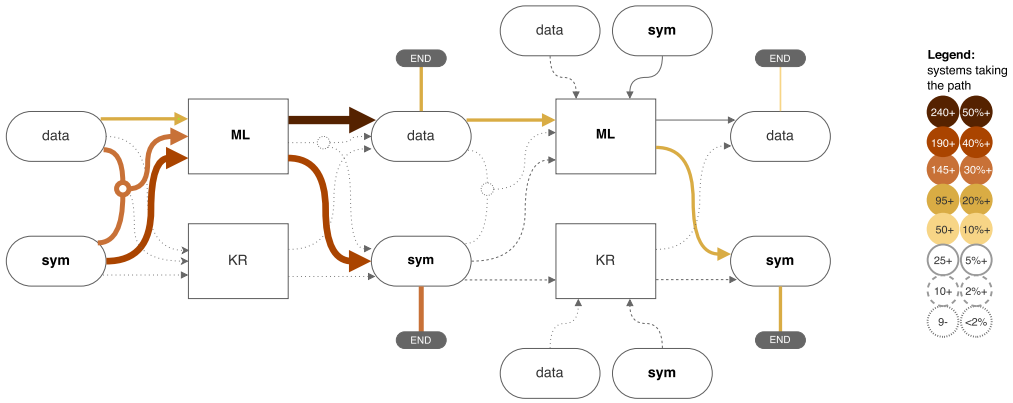

Fig. 13. Meta processing flow of interaction patterns. The most common patterns are fused into a meta pattern to illustrate the most popular paths. The thickness and color of the connections show the number of systems taking this path. A cycle represents a fusion.

The output of type *sym* is in most cases either produced in one processing step (e.g., `s-M-s`) or via an intermediate *data* representation (e.g., `s-M-d-M-s`). This conclusion is also applicable to *data* output, although with an overall lower frequency. Generally, *data* enjoys significantly higher popularity as intermediate representation type compared to *sym* (patterns `T1-T4`, `T6`, `I1-I2`, and `Y1-Y2` exclusively use the path via the intermediate *data* representation, but only T5 via intermediate *sym*). The trend can be also observed in the temporal analysis in Figure 12, showing a steep incline of publications with intermediate representations starting from 2016. In those papers, *data* is used more frequently as intermediate representation compared to *sym*. This development could be explained by the increased incorporation of embedding and representation learning methods as pre-processors.

5.2.4 Conclusions: RQ2. The usage of the boxology framework allowed us to gain deeper insights into the processing patterns of SWeML Systems. Overall, we discovered 41 different patterns, where simple patterns that only incorporate one ML module are more often used than more complex ones; however, we observed that the number of modules used in SWeML Systems is growing over time.

In terms of input-output analysis, we observe that *sym* data structures are almost always used as input, and often as output of SWeML Systems, whereas *data* is often used as intermediate representation. Combining both *data* and *sym* as input is quite popular, especially through an *ML* module.

5.3 RQ3 Application Areas

SWeML Systems can be characterized in terms of the kind of tasks they aim to solve and the domains in which they are applied.

5.3.1 Targeted Tasks. From the papers in this survey, we identified several *tasks* targeted by SWeML Systems, which we grouped into four main *task categories*: tasks based on *Natural Language Processing* (*NLP*), on *Graphs*, on *Image*, and *Other* tasks. Tasks and task categories can be seen in Figure 14. PUB6 and PUB7 target Named Entity Recognition and relation extraction and therefore fall under the *NLP*-based tasks *Annotation* and *Information Extraction*. Another *NLP* system is the geological text classification approach introduced in PUB4, which solves a *Text Analysis* task. The *Creation* task of automatic taxonomy induction from PUB5 and the *Extension*

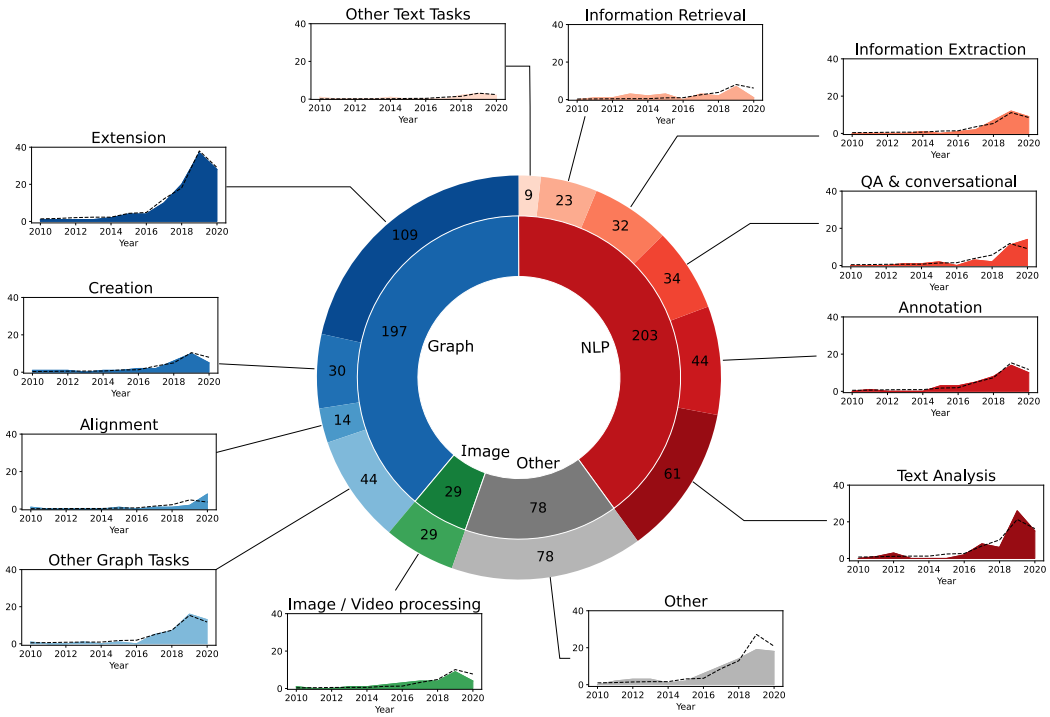

Fig. 14. Distribution of papers per targeted task. The dashed line represents the expected distribution $E(\mathbf{x}_T) = \frac{N_T}{N} * \mathbf{x}_T$ for task T, where N is the total number of papers published, N_T is the total number of papers published that target task T, and x_T represents the publications per year that target task T.

task of identifying missing links from PUB3 are examples for the *Graph*-based task category. Finally, PUB1 and PUB2 target *Image* tasks with their artwork analysis model and visual question answering system, respectively.

Distribution of Tasks. Figure 14 shows the distribution of systems in terms of their tasks. The task categories *NLP* and *Graph* are the most frequent, covering 40% and 38.9% of systems, respectively. In contrast, only 5.7% of the SWeML Systems are concerned with the least common category of *Image*-related tasks (i.e., image or video annotation and classification, image segmentation, action recognition, object detection), whereas 15.4% of the surveyed papers target *Other* tasks, such as recommender systems, data augmentation, or association rule learning. When taking a closer look at the *Graph*-based tasks, we see that almost 55% focus on *Graph Extension*. All remaining sub-tasks each represent 7% to 22% of the total number of *Graph*-based tasks. In comparison to *Graph*-based tasks, the distribution of *NLP* sub-tasks is not as skewed. In this category, *Text Analysis* is targeted most often, in roughly 30% of the papers, followed by *Annotation* in 21.7% of the publications, and by *QA & conversational* and *Information Extraction* sub-tasks in nearly 16% of the papers.

Tasks over Time. The evolution of targeted tasks over the years is also shown in Figure 14. To facilitate the interpretation when correcting for the overall number of published papers, an expected temporal distribution was added that indicates increased interest (actual publications higher than expected value) and decreased interest (actual publications lower than expected value) in a specific task for a given year.

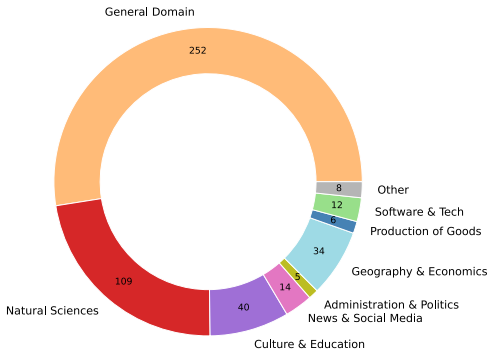

Fig. 15. Overall distribution of systems across domains.

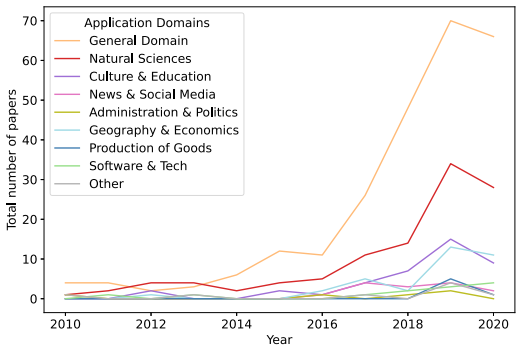

Fig. 16. Temporal evolution of interest for domains. The total number of systems published in a specific domain is shown.

Although before 2016 we observe an increased interest in *Information Retrieval* tasks, this seems to slowly decline in more recent years. The current trends of the last 2 years include *QA & conversational* and *Text Analysis* and *Graph Alignment* tasks. In contrast, solving *Image* tasks seems to have become less relevant to the SWeML community recently.

5.3.2 Application Domain. SWeML Systems can either be domain independent (e.g., PUB2 and PUB5) or used in a specific application domain, such as *Natural Sciences* (e.g., biology for PUB3 and health for PUB6), *Culture & Education* (e.g., art for PUB1), or *Geography & Economics* (e.g., geology for PUB4).

Distribution of Application Domains. More than half of the surveyed SWeML Systems are *General-Domain* ones (Figure 15). However, some papers target specific application domains. Among the domain-dependent systems, the largest share (26.7%) is from *Natural Sciences* domains, such as biology, medicine, or chemistry, followed by *Culture & Education* (including education, academia, digital humanities, art) and *Geography & Economics*. The least popular application domains for SWeML Systems appear to be *Production of Goods* (e.g., manufacturing, transportation, and logistics) and *Administration & Politics*, represented in only 1.5%, respectively 1.2%, of the surveyed papers. Later, Figure 28 shows an overview of concrete SW resources specific to these domains.

Application Domains over Time. Figure 16 shows the evolution of application domains over time. On the one hand, the distribution of papers applied to domains such as *Administration & Politics*, *Production of Goods*, or *News & Social Media*, has remained relatively constant over the years. On the other hand, the number of *General Domain* papers, as well as of those targeting *Natural Sciences*, *Culture & Education*, and *Geography & Economics*, showed a steep increase from 2016. The number of *General Domain* papers has not only increased significantly in the past 5 years (by 466% between 2015 and 2019) but has kept growing, although at a slower pace, throughout the last 2 years. In contrast, the number of domain-dependent papers from *Natural Sciences* and *Culture & Education* started to decrease in 2019, by approximately 12% and 50%, respectively. A similar trend can be observed in the case of the *Geography & Economics* domain, which suffered a slow decrease since 2019. However, this observed decrease might be influenced by the incomplete number of papers from 2020 included in the survey.

5.3.3 Correlation of Tasks and Application Domain. The majority of *NLP* and *Image* tasks are applied in *General Domain* scenarios, as illustrated in Figure 17. Among domain-dependent

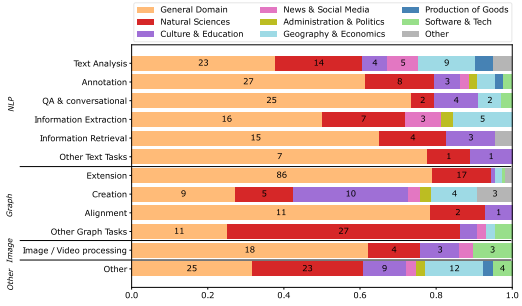

Fig. 17. Normalized distribution of domains per targeted tasks. In the bars, the absolute numbers are provided.

Table 3. Distribution of Input Types Consumed by Systems for Domains with at Least 10 Publications

Domain	Data	Data & Sym	Sym
General Domain	3%	50%	46%
Natural Science	5%	68%	27%
Culture & Education	22.5%	55%	22.5%
News & Soc. Media	14%	79%	7%
Geo. & Economics	21%	56%	23%
Software & Tech	8%	67%	25%
All systems	*8%*	*57%*	*35%*

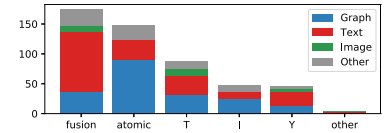

Fig. 18. Task categories solved by systems per pattern type.

applications, *NLP* and *Image* tasks are most often encountered in the field of *Natural Sciences*. One exception is *QA & conversational* tasks, which are applied in the *Culture & Education* domain more often than the *Natural Sciences*. Another interesting observation is that both *Text Analysis* and *Information Extraction* are quite popular in the *Geography & Economics* field. In comparison, the distribution of *Graph* tasks shows much more variation: whereas *Graph Extension* and *Graph Alignment* are mostly applied in *General Domain* scenarios, *Graph Creation* and *Other Graph* tasks are heavily applied in specific domains. More specifically, *Graph Creation* was most widely utilized in the field of *Culture & Education*, followed by the *General Domain*, *Natural Sciences*, and *Geography & Economics*. Similarly, *Other Graph* tasks, such as node clustering or graph pattern mining, are also mainly domain dependent and appear most often in the *Natural Sciences* domain. Furthermore, *Other* tasks are most often applied either in *General Domain* scenarios or in the *Natural Sciences* domain. The field of *Software & Technology* generally represents a rarely targeted application domain for all tasks and is even not encountered in combination with *Text Analysis*, *Graph Alignment*, or *Graph Creation*, whereas 10% of *Image- or Video*-based tasks are applied to this domain.

5.3.4 Correlations of Application Areas and Patterns. In the analysis of input types per target task, we identify that for *Graph* tasks, *sym* is used as sole input in 58% of the cases. Conversely, for *NLP* (77%) and *Image* (90%) tasks, the majority of systems take both *sym* and *data* as inputs. This observation is also reflected in the analysis by pattern types (Figure 18): Fusion and Y-Patterns are often used for *NLP* tasks, whereas Atomic and I-Patterns are more common for *Graph*-related tasks. *Image* tasks are mostly solved with T-, Y- and Fusion Patterns combining image/video and *sym* inputs in the first step.

When analyzing the input types per application domain, we can see different distributions for general domain and domain-specific areas: Table 3 shows that domain-independent systems overproportionally consume only *sym*, whereas domain-bound systems tend to incorporate *data* more often than on average, either as sole input (*Culture & Education* and *Geography & Economics*) or in combination with *sym* (*Natural Sciences*, *News & Social Media*, *and Software & Tech*). A particular outlier is *News & Social Media*, where the usage of only *sym* is 30 percentage points less likely than on average.

5.3.5 Conclusions: RQ3. Overall, we observed that SWeML Systems are used in a wide range of domains, as well as for solving a variety of tasks, which contributes to their increasing importance for authors of numerous scientific disciplines. Furthermore, SWeML Systems are versatile, offering

solutions both in the general domain and in specific application domains, particularly in those that are data intensive, such as *Natural Sciences* or *Culture & Education*. Last, in terms of addressed tasks, *NLP*- and *Graph*-based tasks are the most frequent ones, the latter being facilitated by the growing interest in knowledge graphs in recent years. We further observed that different *Targeted Tasks* and *Application Domains* have preferences for certain pattern types, especially with respect to their input types. This insight is useful from a systems engineering perspective since it helps the engineer select an appropriate processing pattern given a task, a domain, or an SW resource.

5.4 RQ4 Characteristics of the ML Module

Each ML module of a SWeML System can consist of multiple parts: not only can it be a combination of different *ML models* or categories, but this combination can also introduce further ML *components*. In the ML module of PUB4, for example, a word2vec and a Bi-LSTM *model* are applied, which are further extended by an attention mechanism *component*. Investigating these module-based characteristics helps to landscape architectural preferences in the field. Additionally, the systems can be analyzed based on their overall *training type*, providing possible insight into the efforts needed to develop such systems.

5.4.1 Model Categories. Due to the great variety of ML models used in the analyzed systems, an abstraction into *ML categories* was necessary to enable meaningful analysis. ML categories summarize related families of ML models—for example, the modular co-attention network introduced in PUB2 goes together with BERT-based models (among others) in the category *Transformer*, whereas word2vec, fasttext, and RDF2vec are all part of the category *Plain Encoder*. As it was not possible to construct a global taxonomy for these ML categories,[11] we only introduce the shallow separation of (1) *Classical ML* (i.e., not neural network based) and (2) *Deep Learning* (*DL*) (i.e., neural network based). As the emerging field of neural networks that can directly operate on and exploit structural information of graph data is of special interest when analyzing SWeML Systems, we introduce a third super-category for (3) *Graph Deep Learning* (*Graph DL*). Examples for *Classical ML* are the association rule mining approach introduced in PUB3 or the SVM applied in PUB7, whereas models such as word2vec (PUB5-7), CNNs (PUB1), LSTMs (PUB2, 4 and 6), and approaches such as LINE (PUB7) and node2vec (PUB1) are classified as *DL*. The raw data of ML categories can be found in Appendix A.3.

Distribution of ML Categories. Figure 19 summarizes the temporal evolution of ML categories, sorted by their usage frequency. The overall frequency is also shown later in Figure 21. The three most frequently used ML categories are *Encoders*, *Plain Feed Forward Neural Networks* (*FFNNs*), and *Translational Distance Models*, where the large occurrence of *Encoders* is mainly due to the heavy usage of the word2vec algorithm in the surveyed publications. The top five ML categories are all *DL*, whereby there is only one dedicated *Graph DL*.

Since the number of publications varies significantly over time, we add the expected distribution of ML categories if the categories would have the same share of publications each year. This helps identify usage trends while correcting for overall growing publication numbers. With the exception of *Matrix Factorization* and *Recurrent GNNs*, the number of systems incorporating *DL* models has been rapidly growing in recent years, showing an increasing interest by the research

[11] In fact, we found that there is no overarching taxonomy of ML algorithms, as existing classifications are either too coarse-grained regarding modern DL approaches (e.g., [21]) or only focus on (sub-areas of) DL (e.g., [27], [40]). Since constructing one ourselves would be biased toward the papers selected in this survey, we decided to create appropriate categories for this survey for which we do not claim universal completeness.

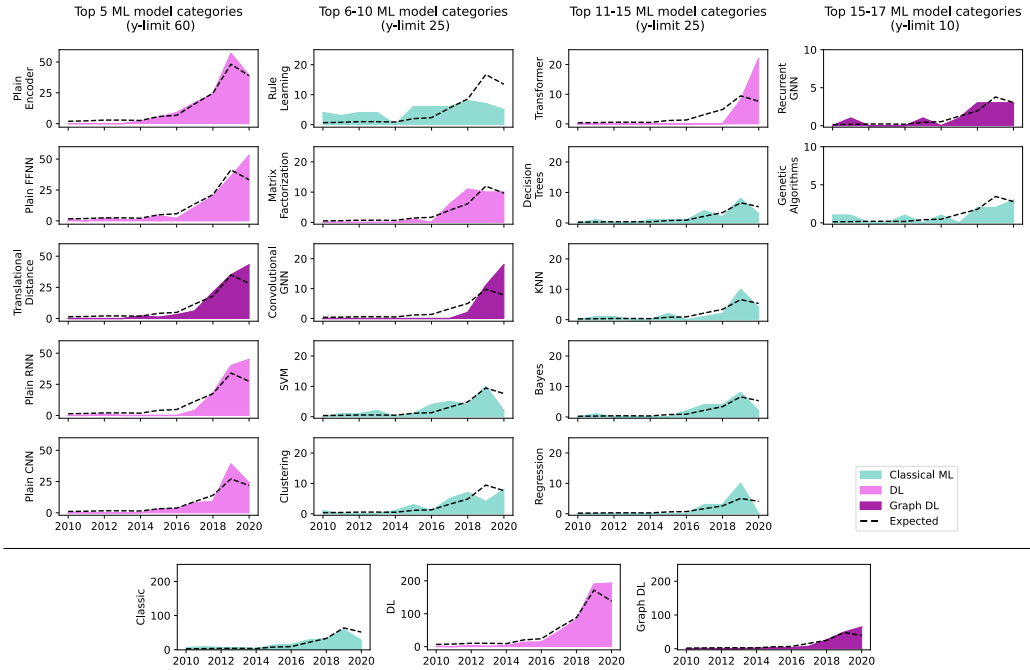

Fig. 19. Temporal evolution of ML categories that appear in at least 10 papers. Plots are sorted by their aggregated usage frequency. The dashed line represents the expected distribution $E(\mathbf{x}_C) = \frac{N_C}{N} * \mathbf{x}$ for ML category C, where N is the total number of papers published, N_C is the total number of papers published that contain the ML category C, and x_C represents the publications per year for ML category C.

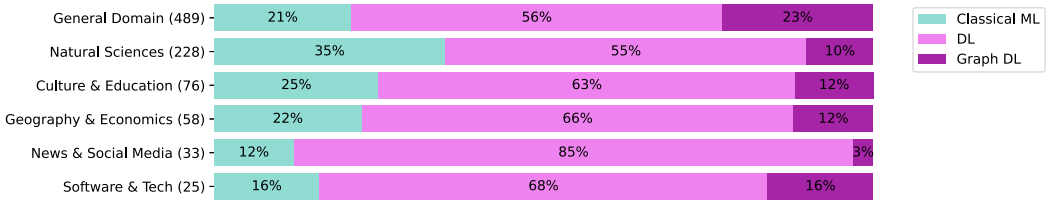

Fig. 20. ML categories per domain, for those domains that characterize at least 10 papers. In parentheses, the number of models (not systems) deployed for the specific domain is shown.

community. In contrast, publications of *Rule Learning* approaches, which were the most popular type of ML category before 2014, stay at a constant rate.

ML Category Usage in Application Domains. The distribution of ML super-categories for each of the six largest application domains is presented in Figure 20. In all domains, *DL* is most prominent; it is used on average in around two-thirds of the papers. Our analysis shows that *Classical ML* is most prevalent in *Natural Sciences*, whereas the relative share of *DL* models is the greatest in *News & Social Media*. In the *General Domain*, models from *Graph DL* are notably more often applied compared to their usage in specific domains. This could be seen as an indicator for *Graph DL* still being in the phase of developing and establishing models, which are evaluated on established domain-independent benchmarks (e.g., based on DBpedia, WordNet, or Freebase)—only slowly transitioning toward domain-specific applications.

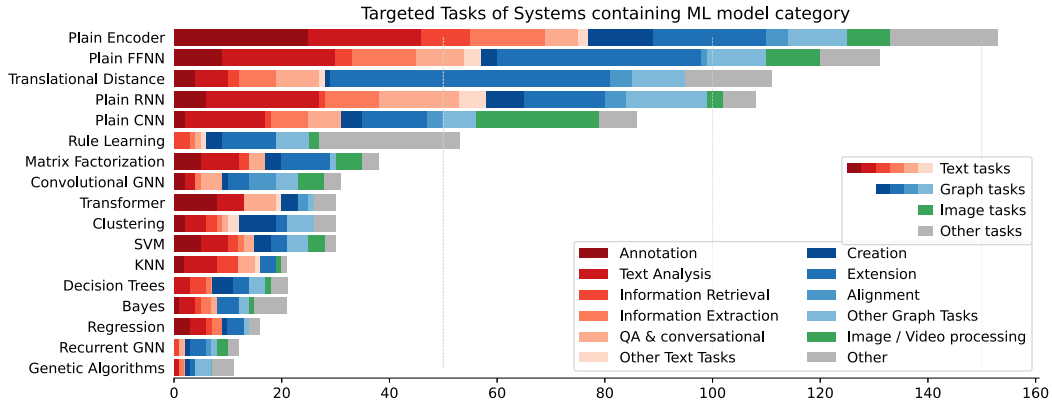

Fig. 21. Relation between ML categories (those that are used in at least 10 papers) and tasks performed by the analyzed systems. Both task categories and individual tasks are shown via color coding, and ML categories are sorted by total frequency.

ML Category Usage per Task. The individual ML categories were analyzed in combination with the *Targeted Tasks*. Figure 21 shows the total number of occurrences for each category together with the most frequent tasks and task category. *Graph* and *NLP* tasks appear in every ML category, typically fairly balanced. ML categories that focus on solving *NLP* tasks include *Plain Encoders*, *Transformer* models, and *kNN*, whereas *Graph DL* model categories such as *Translational Distance* models and *Convolutional GNNs*, but also *Rule Learning* models, mainly focus on *Graph* tasks. *Image* tasks are addressed particularly with *CNN* algorithms. Overall, we do not see domination of a task category for a specific ML category; instead, ML categories are generally used across tasks—even presumably task-bound algorithms such as *Translational Distance* models or *Transformer* models are broadly used in multiple tasks.

5.4.2 Deep Learning Components. Since *DL* models may be composed of multiple architectural patterns, we further analyzed which basic building blocks were used in these models. We identified the following *components*, which we assigned on the ML *module* level: (1) *Feed Forward* (*FF*), (2) *Recurrent* (*Rec*), (3) *Convolution* (*Conv*), and (4) *Attention* (*Att*). *FF* components can be found for example, in the word2vec algorithm of PUB4-6; *Rec* components are present in the LSTM models (including ELMO) in PUB2, PUB4, and PUB6; and PUB1-2 also incorporate *Conv* components. *Attention* can be seen as a standard component of a deployed model, such as the modular co-attention network introduced in PUB2, or as an extension to other models, such as in PUB4, where a word2vec and a BiLSTM model are combined with an attention mechanism.

Figure 22 summarizes the main statistical properties of the DL components of SWeML Systems—that is, their total frequency as well as their combined usage. Given the overall usage, *FFs* are clearly the main DL component in SWeML Systems, whereas the remaining components (*Rec*, *Conv*, and *Att*) are used equally often. We found that all possible combinations of components occur. The combination of *Conv* and *Rec* is quite unpopular; however, both of these components are frequently (and almost equally likely) combined with *FF* and *Att*, which reflects the universality of these components—for example, to add a classification layer or to improve prediction quality, respectively. Furthermore, almost 70% of systems that incorporate *Att* also use *FF* components, where the 28 identified *Transformer*-based models play a major role.

An analysis of DL components per task reveals that *FFs* dominate each task (Figure 23). *FF* is a broad group of many simple DL architectures, and therefore its broader usage is not surprising.

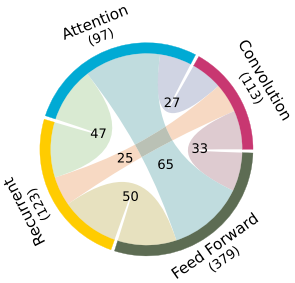

Fig. 22. Co-occurrence of DL components within a system. In parentheses, the total frequency is provided.

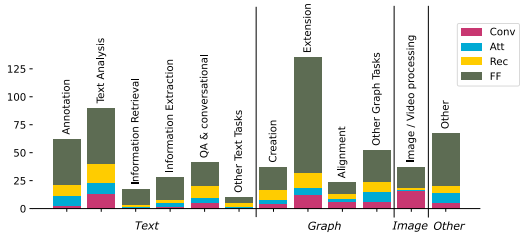

Fig. 23. Tasks solved by systems containing a certain DL component.

The remaining components are relatively evenly spread. An exception is the dominance of *Convs* for *Image* tasks.

5.4.3 Training Type. We identified five distinct training types, which are analyzed independently of the model to provide a higher-level understanding. In the following, we introduce each training type whereby the total number of papers falling into each category is provided in parentheses (five papers did not provide sufficient information about the training type used):

Supervised ML (194): A supervised model is trained based on labeled data—for example, a fraud detection system that recognizes fraudulent financial transactions based on 10,000 manually labeled transactions.
Self-supervised ML (188): A self-supervised model uses unlabeled data in combination with a training objective—for example, a word2vec algorithm that uses an unlabeled text corpus as input. Given a sentence from the corpus, the algorithm is trained to predict the surrounding words for each word in the sentence.
Semi-supervised ML (24): A semi-supervised model is trained based on a small set of labeled and a large set of unlabeled data. A classifier is typically trained on the labeled data and then used to create further pseudo-labeled data. The entirety of labeled and pseudo-labeled data is used to train the final classifier. This approach is often used, for example, for text classification.
Reinforcement learning (7): Reinforcement learning does not require labeled data but instead requires a feedback loop where the machine making decisions obtains feedback and optimizes its strategy over time. An example would be an algorithm that is tasked to play a game such as Go or chess.
Unsupervised ML (58): In unsupervised learning situations, a model is obtained without any labeled data. An example would be an algorithm that clusters products of a company.

Note that these training types are applied on the system level rather than on the individual modules or models. For example, if a system—such as PUB6—uses models that are trained in a self-supervised fashion (e.g., word2vec) to train a supervised approach (the BiLSTM-CRF classifier), we consider the entire system supervised. With regard to the overall distribution of the different training types, we found that supervised and self-supervised models account for approximately 40% of the systems each, no supervision is used in approximately 12% of the systems, and models based on semi-supervision or reinforcement are rare. Examples for semi-supervision are, for example, the iterative hierarchy induction approach from PUB5 and the surrogate learning approach from PUB7. This finding is intuitive since on the one hand supervised systems are the most widely

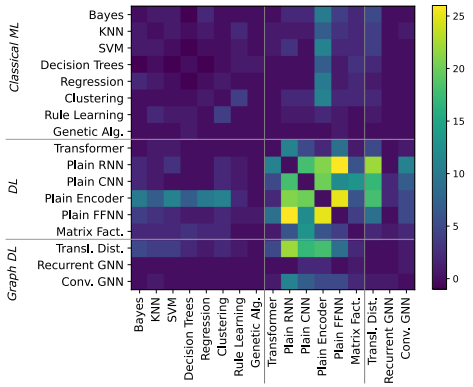

Fig. 24. Combination frequency of ML categories (i.e., their co-occurrence within a system). ML categories are grouped by super-categories.

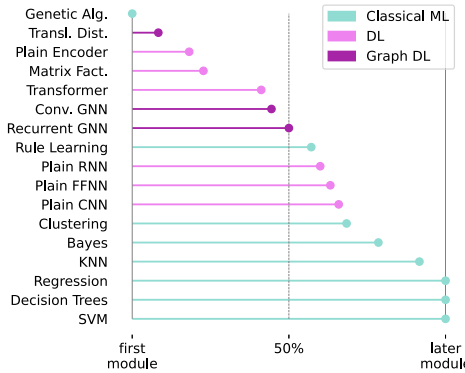

Fig. 25. Position distribution of ML categories in systems with more than one algorithmic module. Shown is the relative frequency of usage within the first or a later module.

adopted and established type of system (e.g., PUB1, PUB2, PUB4, and PUB6), and on the other hand a wide range of models that are most commonly trained in a self-supervised fashion such as different kinds of embedding models (e.g., for systems targeting *Graph Extension* tasks) or rule learning approaches (cf. PUB3) are heavily applied in the analyzed systems.

5.4.4 Combination of ML Categories and Patterns. The analysis of ML categories combined in a single SWeML System is shown in Figure 24. It can be seen that, in general, a combination of *DL* models is more common than a combination of *Classical ML* models. *Plain Encoders* (e.g., word2vec) are combined with a wide variety of models, including *Classical ML*, *DL*, and *Graph DL* categories. As depicted in Figure 25, *Plain Encoders* mostly take the role of a pre-processor, meaning that they are applied in the first processing step of the SWeML System. The same observation can be made for *Translation Distance* models; however, these graph embedding models are less likely to be combined with *Classic ML* models.

Plain neural network architectures (e.g., *Plain FFNN*, *Plain RNN*, *Plain CNN*) are more often combined than advanced architectures (e.g., *Transformer*), where plain neural networks have a slight trend to be used in a later step, whereas *Transformer* models are slightly more often used in the first processing step. The trend of combining simpler models is also reflected in the *Graph DL* area: complex models such as *Recurrent GNNs* are far less often combined than simpler ones such as *Translational Distance* models (e.g., TransE).

Many *DL* and *Graph DL* categories show a rather balanced position distribution, except for those categories mainly consisting of simple graph and text embedding methods (see Figure 25). *Classical ML* categories seem to have a more established position: *Genetic Algorithms*, for example, are always used in the first step, whereas *Decision Trees*, *SVMs*, and *Regression* models are exclusively being used in later steps when combined with other ML models.

5.4.5 Conclusions: RQ4. Our analysis showed that SWeML Systems use primarily *supervised* and *self-supervised* models. ML categories are used across applications without a significant exposure of one category to exclusively one task. Throughout all application domains, *DL* categories are most prevalent (with a share of 60%) followed by *Classical ML* categories. *Graph DL* categories occur with low frequency (<20%) in all domains except the *General Domain*, where they occur significantly more often. The popularity in the *General Domain* potentially indicates that the models are still being explored and did not transition into more specific applications.

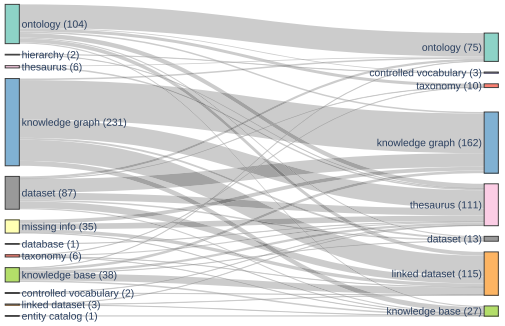

Fig. 26. Comparison of semantic resource types as assigned by authors (left) and the study team (right).

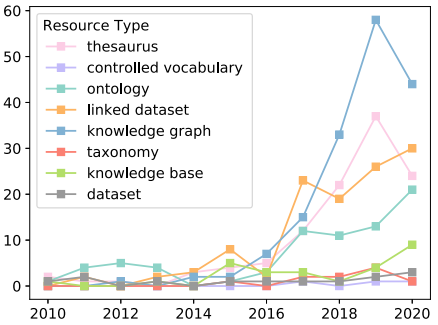

Fig. 27. Temporal evolution of semantic resource types. Types correspond to those assigned by the study team.

Interestingly, we found that even presumably task-bound ML categories (e.g., *Transformers*) are used in a variety of tasks (i.e., are not purely dependent on a particular task). Furthermore, *DL* categories—especially those that include typical and rather simple embedding models—are most likely combined with other ML models and used as system pre-processors. Component-wise, we found that the most broadly used DL component in SWeML Systems is *FF*, which is most often combined with *Att*, whereas the combination of *Conv* and *Rec* is least popular.

5.5 RQ5 Characteristics of the SW Module

SW knowledge structures (aka SW resources) play an important role in SWeML Systems. From the reviewed 476 papers, a large number of papers (307) make use of already existing (i.e., *predefined*) semantic resources such as general domain resources (e.g., *DBpedia*, *YAGO*) that usually play the role of *inputs*. Furthermore, some of the reported systems create *custom* SW resources as their *output* or as internal, *intermediary representations*. Since in the papers reviewed the information about semantic resources, such as type, size, and representation formalism, is generally weakly specified, we focus our analysis on the predefined semantic resources, where we can collect this information from the sources that describe the resources as such (i.e., resource websites, relevant papers). As some papers rely on several SW resources, we identified 516 non-unique (corresponding to 139 unique) predefined resources in the 307 papers that make use of such resources.

5.5.1 *Types of Semantic Resources.*

Author Assigned Type. The left part of Figure 26 depicts the types of the 516 predefined SW resources as mentioned by the authors of the analyzed papers. Several papers fail to name the type of the resource; however, others most often mention employing resources of type *knowledge graph*, followed by *ontology*, *dataset*, and *knowledge base*. Less frequent mentions of semantic resource types are *taxonomy*, *thesaurus*, *hierarchy*, *controlled vocabulary*, *linked dataset*, and *entity catalog*. This indicates that on the one hand, there is a preference for using rather generic terms (*ontology*, *knowledge graph*, *dataset*) as opposed to more specific terms that describe more specialized types of resources (e.g., *controlled vocabulary*, *taxonomy*), and on the other hand, terms novel to the SW community are introduced (e.g., *database*, *entity catalog*). Both could be explained by the fact that several of the studies originate from communities other than the SW community, where such terminological knowledge is present to a more limited extent.

Type Assigned by the Study Team. As our study includes papers from several communities, we noticed an inconsistent use of terminology among these communities as well as over time (with

the same SW resource being categorized as *ontology*, *linked dataset*, or *knowledge graph* depending on the trending terminology at the time the corresponding paper was written). For example, the term *knowledge graph* was popularized only in 2012 by Google, although some of the resources that would be called *knowledge graphs* today were already in use before. Therefore, besides collecting the types of these resources as mentioned by the paper authors, we also evaluated the semantic resources and assigned them a type[12] based on the following predefined glossary[13]:

> *Thesaurus*: A controlled vocabulary connected with relations that express linguistic relations such as equivalency (synonyms), and broader/narrower relations without strict logical semantics such as subsumption. Examples are Agrovoc, WordNet, ConceptNet.
> *Taxonomy*: A domain model containing terminological (T-Box) information limited to concepts and their subsumption hierarchy.
> *Ontology*: A terminological model richer than a taxonomy containing also additional named relations and axioms (T-Box).
> *Dataset*: Contains semantic instance data (or metadata), corresponding to an A-Box in logics. A collection of triples describing instances can be considered a dataset.
> *Knowledge base*: Contains both terminological and instance knowledge (TBox+ABox).
> *Linked dataset*: A differentiating feature of linked datasets from the semantic structures described earlier is that they contain links (in terms of URI references) to other semantic resources. Additionally, such datasets contain large numbers of instance data, although they may include (lightweight) terminological knowledge as well. The canonical example is DBpedia.
> *Knowledge graph*: *Knowledge graph* was most recently defined as "a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities" [17]. This is a very broad definition that actually subsumes all the semantic resource type definitions presented earlier. However, when one resource could be clearly classified in the preceding categories, we did so. This still left a number of resources that represented graph data using a non-semantic formalization (e.g., JSON, XML, proprietary formats), thus making it impossible to classify these along the ABox-TBox distinction which underlies the description of the preceding semantic resource types. Such resources were classified as knowledge graphs.

The right side of Figure 26 shows the types (and corresponding number) of the 516 predefined resources in terms of the types assigned by the study team. Accordingly, we can confirm that *knowledge graphs* are most frequently used. However, contrary to the author-based classification, *thesauri* and *linked datasets* precede *ontologies* in terms of the frequency of their use, indicating a focus on instance data as opposed to terminological information.

We can also conclude on some aspects of terminological (mis)use. The terms that are mostly incorrectly used are *dataset* (to describe resources that are *ontologies*, *knowledge graphs, linked dataset*, or *knowledge bases*), *knowledge graph* (almost half of the uses of this type can be mapped to other resource types), and *knowledge base*. Interestingly, the term *linked dataset* is seldom used by paper authors.

[12]The assigned type reflects the type of the resource per se, independently of its use by the SWeML System—for example, a semantically rich ontology would be assigned the type *ontology* even if the system would only make use of its taxonomic structure.

[13]Although the definition of the semantic resource types reflects the joint understanding of the study's author team, we are aware that, as with all terminologies, variations in definitions exist. Nevertheless, our goal here was to establish a common baseline to define resources that would allow having a consistent view of the analyzed resources and account for terminology variations across communities.

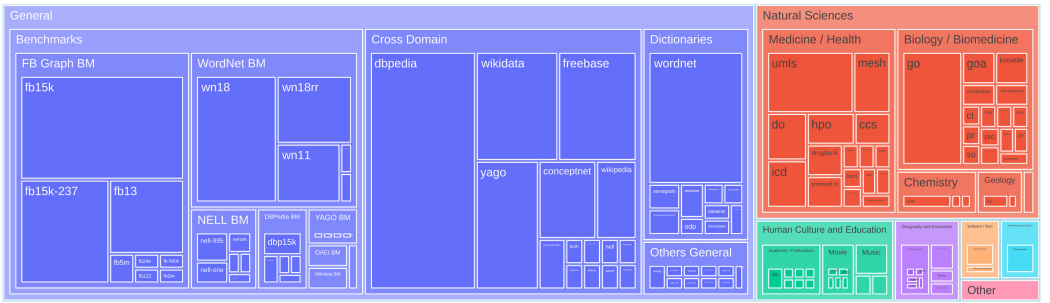

Fig. 28. Concrete SW resources, the frequency of their use, and their domain classification. The raw data can be found in Appendix A.4.

From a *temporal* perspective (Figure 27), whereas ontologies appear as the most frequently used resource types until 2014, from 2016, besides *ontologies*, there is a (sharply) increased interest in other data types, particularly *thesauri*, *linked datasets*, and *knowledge graphs*. *Knowledge graphs* dominate the scene from 2018 onward.

5.5.2 Concrete Semantic Resources. Figure 28 depicts concrete semantic resources used by the reviewed papers, the frequency of their use, and their domain classification (it covers the 516 non-unique predefined resources). Reconfirming findings related to the application domains of the papers (see Figure 15 in Section 5.3.2), the majority of studies make use of domain-agnostic resources, whereas the most domain-specific resources stem from the topmost popular domains—that is, *Natural Sciences*, *Culture & Education*, and *Geography & Economics*.

Analyzing the concrete resources themselves, we conclude that a large number of different resources are used, and there is a mix between frequently used resources and more specialized/niche resources. Among the domain-agnostic resources, *DBpedia*, *YAGO*, *Freebase*, and *WordNet* are the most frequently used resources, either as the resources themselves or benchmarks based on them. From the domain-specific resources, most frequent are those from *Natural Sciences*, particularly *Medicine/Health* (with often used reference resources such as *UMLS*, *Mesh*, and *ICD*) and *Biology* (with the *Gene Ontology* being the most frequently used resource). In the *Human Culture and Education* domain resources related to *Movies*, *Music* and *Academic Publishing* are the most frequent (e.g., the AIFB research ontology being the most frequently used).

5.5.3 Resource Size and Representation Formalism. Figure 29 depicts the size of used resources (in terms of number of triples) over time considering the resource size as collected by the reviewing team in 2021 (which, in some cases, might differ from the resource size when it was used by the reporting study). We conclude that although in the first part of the past decade (up to 2014) there is more or less an equal mix of resource sizes, from 2014 larger resources (with > 1M triples) are more prevalent and there is a sharp rise in the use thereof. This could correlate with the increased popularity of ML methods that require larger training datasets, the availability of large-scale linked datasets and knowledge graphs, and the increased scalability of processing tools as well as the available compute power. Indeed, several of the frequently used resources, such as *DBpedia*, *Freebase*, or *YAGO*, exceed 1M triples. Additionally, several very large domain-specific resources are employed—for example, *UMLS* (Health), *MusicBrainz* (Music), and *Microsoft Academic Graph* (Academia/publishing). In terms of the (knowledge) representation formalisms used (Figure 30), whereas until 2014 there is no clear trend, from 2014 most frequently RDF/RDF-S is employed, followed by a widespread practice to serialize SW resources in non-semantic formats such as XML, JSON, and a number of non-mainstream/proprietary formats. This is in line with

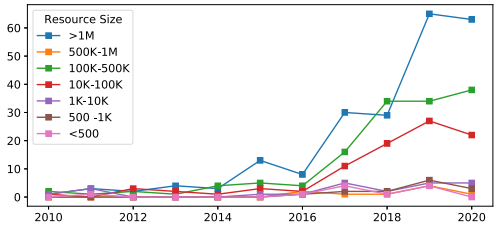

Fig. 29. Temporal evolution of semantic resources in terms of their size in number of triples.

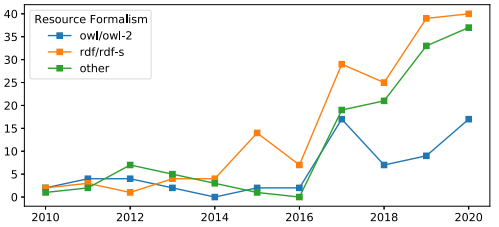

Fig. 30. Analysis of semantic resources in terms of their representation formalism.

the trend of using large, instance-heavy resources for which such serialization formats with light-weight semantics basis suffice. The least frequent was the use of OWL/OWL-2 representations, which allow semantically more fine-grained knowledge representation.

5.5.4 Usage of SW Resources. To better understand how the SW resources are employed, we investigate the extent to which these resources are used:

- *Only labels (43)* of concepts or relationships were used, but no structural information. For example, in PUB4, the labels from an ontology provided keyword importance for processing textual input.
- *Simple relations (44)* (i.e., triples of one relation type, only considering one hop) were used. An example is PUB2, where descriptive statements are queried from ConceptNet, given object labels and attributes.
- *Hierarchical structure (53)* includes typical *is_a* kind of relations, which could span over multiple levels. PUB3, for example, makes use of the hierarchical structure of *Gene Ontology*.
- *complex structures (289)* include collections of concepts and relations, and property chains over multiple relationships. These are, for example, used by graph embedding models, as it is the case in PUB1.

There is a tendency to use complex structures from the resource as opposed to more simple aspects thereof. Indeed, the majority of the papers (60.71%) made use of *complex structures* from the resources, whereas 11.13% explored the *hierarchical structure* of the resources. *Simple relations* were exploited by 9.24% of the papers, whereas 9.03% of the papers made use of *only labels*. For 6.72% of the papers, this aspect was not applicable (e.g., in PUB5, where a semantic resource is created), and for 3.15% of the papers, this information was not available.

5.5.5 Semantic Processing and NeSy. Interestingly, only a small number of papers (29, approximately 6%) report on making use of *semantic processing* (*KR*) units: in 20 papers, *reasoning* capabilities are used to infer new knowledge from the semantic resources, whereas in 9 papers, *SPARQL queries* are employed to select a relevant subset of the resource. Despite their small number, these systems that combine techniques from ML and KR (i.e., NeSy) are of particular interest. We identified 14 patterns that contain such a combination and subsequently analyzed the types of processing that were performed by these systems. Based on the results in Table 4, we summarize our findings next.

Distribution. The distribution of these NeSys follows the general trends: there is no growth over the years beyond the general increase of papers over the years, and the selection of application domains is reflecting the general distribution of papers (a high number of papers in the *General Domain*, *Natural Sciences*, and *Human Culture and Education*). The NeSys differ in their targeted

Table 4. Analysis of the Connection of Interaction Families/Processing Types and Patterns in the Analyzed NeSy

Interaction F.	Processing Type	I3	O1	T7	T5	T8	T9	T10	T11	T12	T13	T14	T15	I4	O2
ML → KR	Contribute to same task	2	1	1	2										
	Learn & improve ruleset				4	1									
	Learn & apply ruleset					3	3	1	1						
KR → ML	Data enrichment									5	1				
	Model enrichment											1	1	1	
ML || KR	Contribute to same task														1

I3: d-M-s-K-s
T7: {{{d/s}-M-d/s}-M-s/s}-K-d
O1: *highly complex*
T5: {d-M-s/s}-K-s
T8: {s-M-s/s}-K-s
T9: {s-M-s/d}-K-d
T10: {d-M-d/d/s}-K-s
T11: {{d/s}-M-s/d}-K-d
T12: {d/s}-K-s-M-s
T13: {d/s}K-d-M-d
T14: {s-K-d/d}-M-s
T15: {s-K-s/s}-M-s
I4: s-K-d-M-s
O2: d-{M||s-K}-s

Pattern O1 is of very high complexity and cannot be depicted in flat notation. The corresponding boxology notation can be found in Figure 7 on the right.

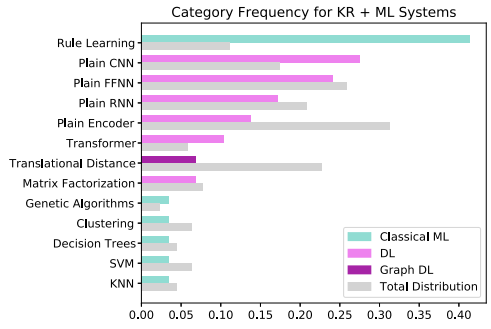

Fig. 31. Distribution of ML categories used in NeSys compared to their overall distribution in all analyzed SWeML Systems.

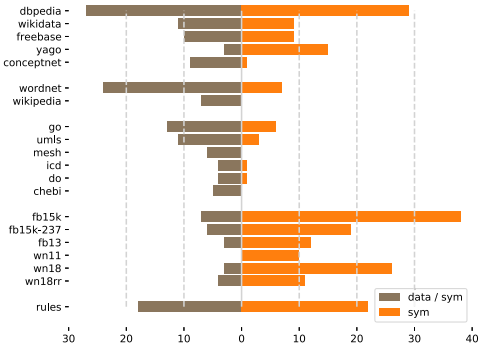

Fig. 32. Input type of SWeML Systems containing a specific SW resource.

tasks: *Image* analysis and *Other* tasks are overrepresented among them compared to the low number of papers targeting these tasks in the general corpus. Because of addressing *Image* tasks, the ML modules often use *CNNs* (Figure 31). Furthermore, *Rule Learning* models are used to a much greater extent than in the overall distribution. This is to be expected since *Rule Learning* combines naturally with the symbolic representations used in the KR component. Finally, embedding models (*Plain Encoder* and *Translational Distance*) are underrepresented.

Interaction Families and Processing Types. Using ML before KR is much more popular than KR before ML: two-thirds of the papers belong to ML→KR, almost one-third to KR→ML, and only 1 to ML || KR (i.e., using ML and KR in parallel). These three interaction families were further subdivided into five processing types: (i) the ML systems learn a ruleset that is subsequently improved by a KR module (e.g., through checking for redundancy or inconsistency); (ii) the ML system learns rules which are subsequently applied by a KR module; (iii-iv) a KR module is first used to enrich either the ML model or the data which is then used by an ML system; and (v) an ML and KR approach are both applied to the same task. By far the most frequent use of combined systems is to extract rules from data (13 out of 29). These extracted rulesets are then either improved with a KR module or directly applied in a KR module.

Coupling Between Patterns and Processing Types. Four of the 14 patterns (*T5*, *T8*, *T9*, *T12*) account for 18 of the 29 papers, and all of these 4 are T-patterns. Twelve of the 14 observed patterns

contribute to only a single processing type (in other words, the matrix from Table 4 forms a near diagonal). This suggests a very strong coupling between our patterns and the processing type that the system is performing.

5.5.6 Combination of SW Resources and Patterns. In Figure 32, we depict the types of input taken by SWeML Systems per used *SW resource*. We see general-purpose resources such as *DBpedia*, *Wikidata*, *Freebase*, *YAGO*, and *WordNet* equally used with *data* and *sym*, and only *sym* inputs. Domain-specific resources such as *GO*, *UMLS*, *MeSH*, *ICD*, *DO*, and *CheBI* are more often used in combination with data to enhance the analysis. Dumps of *Freebase* and *WordNet* are popular benchmarks for various algorithms and hence are often used as the only *sym* input. Remarkably, *Wikipedia*, *MeSH*, *CheBI*, and *ConceptNet* are used almost exclusively with additional data inputs.

Upon analyzing the co-usage of SW resources, no clear pattern has been observed. There are some cases where the most common general domain SW resources are combined (*DBpedia & Wikidata* and *DBpedia & YAGO* in 10 cases each, *YAGO & Wikidata* in 4 cases); however, otherwise, the combination of SW resources—which could be considered as one of the most powerful aspects of the SW—seems surprisingly unpopular. Within the *Biology* domain, where the incorporation of different knowledge sources has a long tradition, the most commonly used resource GO is combined with other *SW resources* in less than 25% of the cases, and the second most common *UMLS* is combined with other *SW resources* in only 14% of the cases.

5.5.7 Conclusions: RQ5. SW resources play diverse roles in SWeML Systems (input, intermediary structure, output); however, their characteristics (in terms of type, size, serialization) are generally weakly reported in the studies. We identified a large number of SW resources that are used in SWeML Systems, both cross domain (*DBpedia*, *YAGO*, *Freebase*, *WordNet*) and domain specific (e.g., *UMLS*, *Mesh*, *ICD*). In terms of resource type, there is a terminological shift across communities and time periods; however, after the alignment to a proposed type terminology, we conclude that although until 2014 the focus is on employing smaller resources of terminological type (*ontologies*), from 2014 there is an increased tendency of focusing on resources rich in instance data (*linked datasets, knowledge graphs*) with very large sizes (>1M triples) and often encoded in non-semantic serializations. Most papers make use of the various complex relations encoded in the semantic resources; however, the use of reasoning mechanisms that fully explore semantic features is limited: only 29 papers combine ML and KR modules. These papers demonstrate a tendency to use an ML module to abstract data into knowledge to be subsequently used by a KR module. We found that although domain-independent resources are combined with *data* input in about half of the cases, this combination is more popular when incorporating domain-specific resources.

5.6 RQ6 Maturity, Transparency, and Auditability

Through the combination of SW and ML methods, researchers and practitioners solve complex tasks with SWeML Systems. However, the applicability of these solutions is often hindered by a low degree of system maturity. This maturity can be defined from a software engineering point of view, meaning that the system should be stable, tested, and verified. Although the first two aspects are not significantly different from other software engineering projects, verification—despite its importance—is often overlooked in AI development, with many claims about benefits and utility not backed by easily accessible evidence. Both companies and the scientific community recognized these problems. Google defined principles guiding its AI development [14]. The third principle stresses that AI-based systems must be tested in constrained environments and that their operation must be monitored after deployment. In the scientific community, the ACM has issued a "Statement on Algorithmic Transparency and Accountability"that emphasizes the need for auditability of models, algorithms, and decisions [1].

To enable the transparency and auditability of systems used for data analysis, it is essential to capture and understand the broad context in which the system operates: data sources, algorithms, processes, and services that the deployed systems depend on. To this end, this research question aims to investigate the maturity of SWeML Systems, with a special focus on auditability and transparency aspects. First, we will categorize the system according to their overall *maturity* level taking into account features like stability and usability. Second, we investigate the *transparency* of the SWeML Systems by checking the following aspects: system components (i.e., description of infrastructure and software stacks) and evaluation-centric parameters (i.e., process steps, metrics, data, and data-split). Our goal is to check whether these aspects were documented, but not to check whether the results could be reproduced. Third, we check whether the SWeML Systems support *auditability* (e.g., by capturing the provenance of the data processing). In this sub-research question, we assumed the field to be still evolving and the notion of auditability not being thoroughly developed. Therefore, we limited the scope of this sub-research question to focus on (i) identifying whether the system captures data processing provenance and what kind of context information is captured, and (ii) the reasoning behind the provenance capturing.

5.6.1 Maturity. Reviewers manually assigned one of the following maturity levels to each paper (total numbers of assigned papers are shown in parentheses):

low (209+236) scripts and prototypes if no source code is available, nor any statement about the system maturity by the authors, the reviewers assigned *probably low*.
medium (23) simple user interface or error handling.
high (8) stable system, available UI, (non-prototype) system is used in a professional context.

In Figure 33, the maturity of systems over the years is depicted. In total, only a small amount of systems qualified for *medium* (4.8%) and *high* (1.7%) maturity, underlining the notion that SWeML is an emerging field with ongoing research efforts. The majority of papers were either *low* (43.9%) or *probably low* (49.6%), describing proof of concepts and initial prototypes for exploring use cases and techniques.

Focusing on changes over time, between 2012 and 2015, the majority of papers did not provide sufficient information on the described systems (*probably low*). Starting with 2016, more papers provided scripts and prototypes (*low*); however, despite initiatives for reproducibility in computer science,[14,15] in the last 3 years, the balance between *probably low* and *low* maturity systems remained rather stable, with a slight surplus of *probably low* maturity systems. Highly mature systems only appear from 2016 onward; however, it is important to note that in this period, the number of publications also increased. An analysis of the correlation between system maturity levels and application domains was inconclusive. In general, the number of mature systems is very low, but future studies might identify an increase in stable systems applied to specific problem domains.

5.6.2 Transparency. Our investigation found that evaluation parameters are typically well documented, with more than 50% of the papers containing all of these parameters (cf. Figure 34(a)). Documentation about evaluation datasets is detailed in almost all papers (96.6%), as well as the evaluation metrics (95.8%) used for quantitative evaluation. Process steps, in which authors report on their evaluation methodology, follow in third place (88.9%). The ML model parameters (84.6%) and data-split (68.9%) were reported less often, which might be because not all systems require them. In contrast with the evaluation parameters, information about system implementation and

[14]ACM Artifact Review and Badging guidelines: https://www.acm.org/publications/policies/artifact-review-badging.
[15]The ACM Task Force on Data, Software, and Reproducibility in Publication: https://www.acm.org/publications/task-force-on-data-software-and-reproducibility.

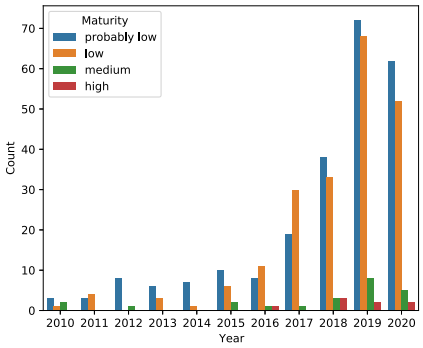

Fig. 33. Temporal evolution of system maturity.

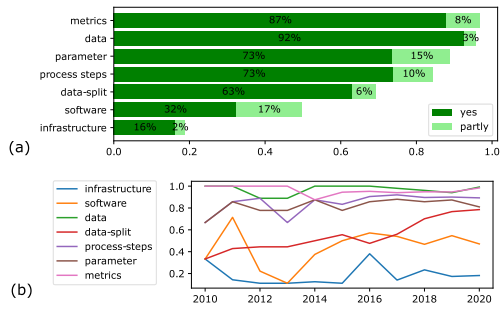

Fig. 34. Coverage of transparency parameters on systems in terms of the percentage of described parameters (a) and normalized described parameters (b) over the years.

evaluation infrastructure is less commonly available. Only 49.6% of SWeML Systems report the software stack (e.g., software framework, libraries) used to implement their system, whereas information about the infrastructure (e.g., hardware, OS) used for evaluation is only available for 18.7% of all systems. Interestingly, we did not see any strong trend toward improvement in this aspect over the years (cf. Figure 34(b)), with only a slight increase in data-split and software specification documentation.

5.6.3 Auditability. We found only three papers that report the capability of capturing the provenance of their systems as the basis for auditability. This situation is understandable given that the focus of most papers is to address the task at hand. Yin et al. [37] reported a system that captures the input data's contribution rate to the final results. To this end, they developed an algorithm for their RNN model to calculate each input contribution. This information is used to explain the result, showing the dynamics between input events and the final results. Song et al. [31] devised a system that accommodates an additional provenance node to RDF each triple. This additional node is then used to capture data source information (e.g., Wikidata or DBPedia). However, they stated that the approach is not limited to this specific usage, as users can also use the additional node for any other information. The goal of the approach is to improve the quality and auditability of the produced data. Finally, Voogd et al. [36] proposed a system where experts provide input values between concepts before and after the training phase of a model. This approach enables experts to interact with the result of system training and explain specific results of the systems. The goal of capturing the provenance of experts' inputs is to enable better explainability by linking results and the original expert inputs.

Additionally, we found an interesting mechanism of provenance capturing provided as part of an evaluation framework for graph embedding techniques in the work of Pellegrino et al. [23]. Although the approach itself is not a SWeML System and therefore is excluded in the data selection phase, GEval provides a mechanism to capture traces of graph embedding system (including SWeML) results and execution of each run to ensure the reproducibility and allow comparison to other systems.

5.6.4 Conclusions: RQ6. We found that the maturity of these systems is relatively low, and over the observed period, there is only limited improvement. Additionally, the transparency aspect of the SWeML Systems is relatively well addressed, with more than half of the papers providing information about their evaluation setup. However, the situation still can be improved by providing information about the implementation aspects, where less than half of the papers explain the

software setup and even less the infrastructure used to evaluate the approach. Auditability and provenance capturing are not the focus of most papers, and separate efforts would be needed to examine this aspect of SWeML Systems.

6 SWEML SYSTEMS CLASSIFICATION SYSTEM

In addition to the formal definition of SWeML Systems (Section 2.1), we further introduce a framework with which such systems can be described, documented, and visualized to improve common understanding. To this end, we provide a *classification system* that not only helps to identify the main characteristics of a SWeML System but also provides guidance on what information should be given by authors when introducing such systems. We formalized our classification of SWeML Systems as an ontology structured in accordance to our research questions (Section 4.1.1) and provide initial instances that were identified during the study conduction. The complete ontology and its documentation are available online.[16]

In the SWeML ontology, we reuse a number of concepts introduced by Van Bekkum et al. [33], including `Instance` and its sub-concepts `Data` and `Symbol`, as well as `Model` and its sub-concepts `StatisticalModel` and `SemanticModel`. To complement these classes, we added a number of classes and properties related to a SWeML System and the connection between the system and its components. The main classes include `System`, to represent a specific SWeML System; `Pattern` to describe the SWeML patterns (Section 5.2); `SystemComponent`, which describes the system components that take one or more input `Instances`, and uses one or more `Models` to produce output `Instances`; `Task` to describe specific tasks of the SWeML System; and `Documentation`, which describes information regarding transparency and reproducibility. We have a dedicated class `SemanticWebResource` to describe the involved SW resources. Instances of `SemanticWebResource` can be used as a `SemanticModel` or `Symbol`, since they contain both their structure and data. The complete description of the SWeML ontology classes are given in Figure 35 and Appendix A. We illustrate the SWeML System ontology using the seven example papers from Section 5. An excerpt of the RDF graph visualization of these instances is shown in Figure 36.

7 CONCLUSION AND FUTURE WORK

This article reported on an SMS of the emerging field of systems that combine SW and ML components (SWeML), a subset of AI systems relying on both symbolic and sub-symbolic techniques. By following a systematic approach to collect the reviewed papers, we conducted a first-of-its-kind large-scale survey, analyzing almost 500 papers. Therefore, by capturing insights drawn from papers across many communities (e.g., NLP, computer vision), this study provides insights into the status and trends of this field and proposes a classification system for SWeML Systems that facilitates their standardized documentation. Our main conclusions are presented next.

Multi-Disciplinary, High-Impact, Rapidly Developing Field. The papers collected in this study demonstrate that research in the area of SWeML is highly *multi-disciplinary*, not only in terms of the techniques employed, which are drawn from the SW and ML communities primarily, but also in terms of (i) the variety of tasks addressed, covering the three large clusters of *Natural Language Processing* (*NLP*), *Graph Processing*, and *Image/Video Analysis*, with graph-based tasks the most popular in the past 5 years, and (ii) the high share and variety of domain-specific use cases addressed, with nearly half of the papers addressing domain problems, particularly those domains where SW technologies already enjoy wide adoption, so that domain-specific semantic resources are already available: the most frequent domain for the SWeML System is *Natural Sciences* (*Medicine, Biology*)

[16] https://w3id.org/semsys/ns/swemls.

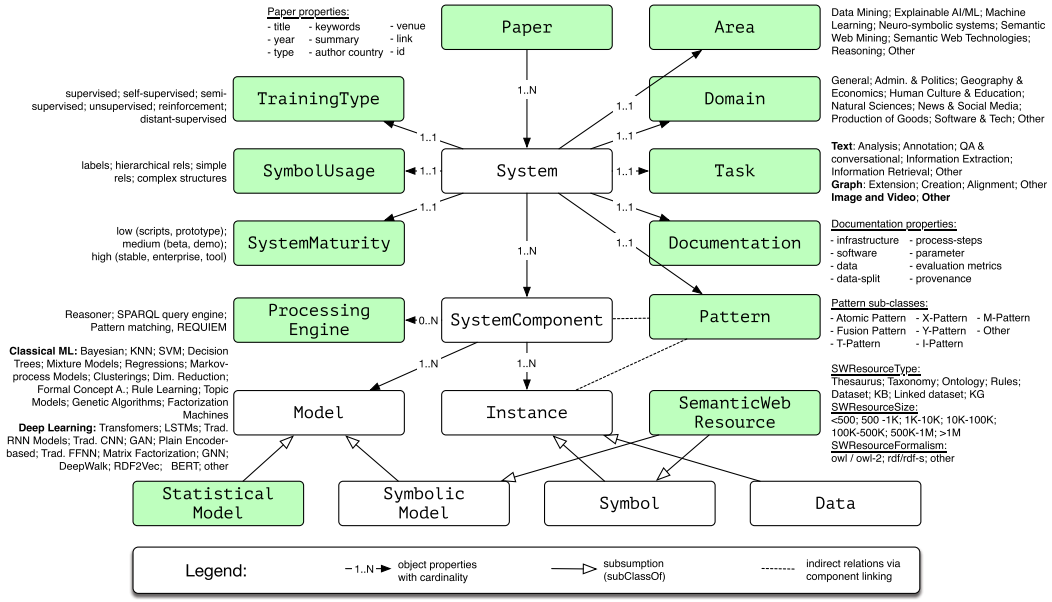

Fig. 35. SWeML Systems classification concepts, relations/properties, and identified instances. Green-colored boxes represent concepts that are subject to the data extraction in this survey, whereas white boxes represent supplementary classes.

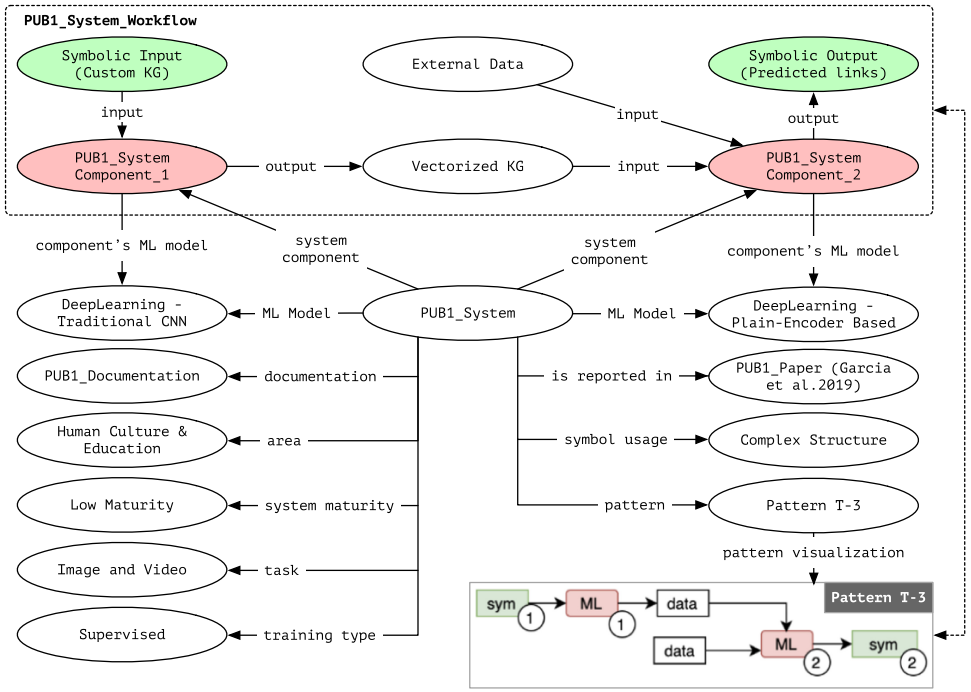

Fig. 36. SWeML System ontology instance of PUB1.

due to the availability of domain-specific semantic resources (e.g., ontologies like *UMLS*, *MESH*, and *GO*). By addressing all these disciplines, the SWeML field is inherently a *high-impact* field. Given such impact potential, a *rapid development* in this field is not surprising: indeed, the publication volume significantly increased from 2014, showing a worldwide interest in the area, with a strong cluster in China and a low representation of the global south. Although growing rapidly, the SWeML field displays the characteristics of a young discipline: the proposed systems have a low maturity level (mostly scripts and prototypes), and although most papers report on elements that allow reproducing their results, provenance tracking and auditability are still very weakly covered.

Deep Learning Is a Key Catalyst for SWeML Systems. Looking at the ML components of SWeML Systems, a wide range of ML algorithms are used pertaining to three broad categories: *Classical ML*, *Deep Learning* (*DL*), *and Graph Deep Learning* (*Graph DL*). *DL* models are the most prevalent (60% of all models), and the interest in them in the past years is above average at the expense of *Classical ML*. Their most frequently used components are *Feed Forward Neural Networks* (*FFNNs*). We also observe that *Graph DL* models are mostly used to solve domain-specific tasks and have not yet been widely adopted in generic tasks. In terms of training, mostly supervised and self-supervised models are used.

Large Knowledge Graphs Are on the Rise. In terms of the SW aspects of the SWeML Systems, SW structures play primarily the role of input resources or are generated as outputs. Less frequently, SWeML Systems rely on internal structures that can be considered a semantic resource. We furthermore found a terminological mismatch across the reviewed papers, as terms denoting the types of SW resources are being used inconsistently across papers and communities. Future work on terminology clarification would therefore be beneficial. Based on a glossary introduced in this article, we found that although until 2014 the focus is on employing smaller terminological resources (e.g., ontologies, taxonomies), from 2014 there is an increased focus on resources of very large size (>1M triples) and often encoded in non-semantic serializations. Most SWeML Systems make use of the complex relations encoded in these semantic resources. However, the use of reasoning mechanisms that fully explore semantic features is limited. Furthermore, information related to the type, size, and serialization of semantic resources is weakly reported. Our recommendation to authors of future papers on SWeML Systems is to more carefully report the technical characteristics of the SW components.

High Diversity of System Patterns. Although the individual ML and SW components have been studied before, in this article we also focused on understanding *patterns* of how such components are used and connected through the information flow in the system. Initially we assumed that the 15 boxology patterns of Van Harmelen and ten Teije [34] will suffice in our analysis to classify SWeML Systems; however, we found a much larger variety of patterns used in practice and introduced 33 additional patterns, as well as a classification scheme of these patterns based on their structural complexity. This newly derived pattern catalog greatly advances the understanding of this aspect of SWeML Systems while also providing new insights for future work toward a *science of patterns*.

High Diversity in Reporting SWeML Systems Requires a More Uniform Classification Scheme. As a rapidly emerging, multi-disciplinary field, the area of SWeML Systems is characterized by a high diversity in reporting these systems by aligning to the canons of the originating disciplines. Such misaligned approaches to reporting these systems hamper communication across communities and advances in the SWeML area. Therefore, we leverage the understanding and analysis of this field gained through the performed study to also propose a *classification system for SWeML Systems* that could lead to a more uniform reporting of such systems to the benefit of the development of the field as a whole. We provide the derived scheme as an ontology both for a better understanding and for allowing authors to create machine-actionable metadata about their systems.

Limitations. Due to the tremendous amount of papers being published in the ML domain, analyzing all of them would not be realistically feasible in a single review paper but would require a series of studies. Therefore, during *study scoping*, we had to carefully choose the keywords to include in our search query. We decided to implement an objective strategy for the collection process as described in Appendix A.1. As a consequence, several keywords, which would be in line with the definition of a SWeML System, were not included in the final ML query. We refrained from the manual addition of further keywords in an effort to avoid the introduction of bias (e.g., originating from our own background or assumptions), and to keep the scope of the study feasible (e.g., adding the keyword *machine learning* to the search query would have added an additional 800 papers to the initial results). Furthermore, the chosen systematic methodology ensures comparability and extension of the presented outcomes, leaving the current work as a starting point for future analysis.

Second, during *analysis*, abstraction of several extracted values was necessary to improve the comprehensibility and interpretability of the results. Due to the lack of established, reusable hierarchies for the ML area or SW resources, these hierarchies were created based on the values extracted in the course of this study. Consequently, for the created hierarchies, we do not claim they represent the entire corresponding field. This exemplifies the need for comprehensive classification systems and taxonomies in these areas and beyond.

Third, we discovered several limitations of the *boxology* (Section 5.2.1) that served as the core of the introduced SWeML classification system. One limitation is that another algorithmic component that is neither *ML* nor *KR* but a generic *Calculation* (e.g., the calculation of a similarity metric) would be necessary to accurately depict the workflow of various systems. Its absence leads in some cases to oversimplified patterns, as in the case of Jayawardana et al. [18], which is associated with the pattern `T1:{d-M-d/s-M-s}`. A second limitation is that although in most cases the workflow of SWeML Systems during the training and prediction phase is the same, where only the exact input data differs, sometimes these phases are inherently different (e.g., in PUB1). In this survey, the entire workflow including both the training and the prediction phase is represented in a single pattern. Although this facilitates the aggregation of the patterns, a distinction of these two phases might provide valuable insights and a more fine-grained classification. A third limitation is that it is only possible to specify that a resource is an input to a system—more fine-grained roles cannot be captured.

Future Work. The data collected during this work allows further in-depth analysis to address research questions beyond the scope of this survey. For example, which concrete resources (e.g., DBpedia) are used for evaluating certain tasks (e.g., graph extension), and is there a danger of biasing research toward using the same (or a handful of) benchmark resources? What are typical patterns used for certain tasks? For example, our analysis shows that *Image*-related tasks are mostly solved with T-, Y-, and Fusion patterns that combine image/video and symbolic inputs in the first step. A similar analysis could also be performed for other tasks (e.g., graph extension), which would lead to a better understanding of emerging patterns addressing those tasks.

The SWeML classification system (and its ontology-based representation) reflects the current understanding of this field while being open to modifications and extensions as understanding of SWeML Systems will advance—for example, providing the ontological support for describing the internal flow of the patterns in more detail.

Finally, the boxology is subject to a number of possible extensions such as (i) adding new algorithmic module types beyond ML and KR to capture system architectures with higher fidelity; (ii) considering a complementary representation format that distinguishes between different lifecycle phases of the system (cf., [33]); and (iii) adding metadata to the flow arrows of the patterns to distinguish the role of the resource within the system (e.g., to depict whether the SW resource is used

as input data for the ML model), or to improve the training process indirectly (e.g., by adapting the loss function [13] or to shape the models' outcomes [24]).

A APPENDIX

A.1 Search Query Selection

Below, we describe the methodology applied to retrieve the three search sub-queries.

Q1 - SW Module. Q1 keywords include (1) the name of the overall field of interest *(semantic web)*; (2) the most frequent terms to refer to semantic structures *(ontolog*, linked data,* and *knowledge graph)* as well as (3) a number of W3C standard names that are likely to be used when implementing semantic web structures of interest for this survey. During the keyword selection a number of keywords were tested but not included in the final query as they all lead to very large result sets containing a lot of false positive hits. These include: semantic, semantic model, vocabulary.

Q2 - ML Module. The field of Machine Learning is extremely wide which is why the task of finding the keywords that most accurately describe the ML module of SWeMLS is highly challenging. For example, including generic search terms such as *machine learning* in the query leads to an unmanageable number of results of way over 2800 works. To overcome this problem and to avoid bias in selecting, an intersection of keywords present in multiple sources was chosen as basis for this query. These sources were ACM Computing Classification System[17] where we considered all concept narrower to *CCS→Computing methodologies→Artificial intelligence* and *CCS→Computing methodologies→Machine learning*, Topics extracted by *Microsoft Academic*[18] that were child Topics to *Machine Learning*[19] or *Artificial Intelligence*[20], and subcategories and pages of the *Wikipedia* categories *Artificial Intelligence*[21] and *Machine Learning*[22]. A keyword was considered, if its concept appeared in all three resources.

To reduce the amount of keywords considered and to increase their quality, terms that described a specific approach or model such as *BERT* or *PCA*, as well as terms that were too unspecific and do not show a clear connection to Machine Learning or Artificial Intelligence without knowing their broader terms (e.g., *visual inspection*) were filtered out during a manual inspection step.

Q3 - System. To assure that retrieved documents present systems that are aiming at solving specific tasks a separate query is introduced. The inclusion of specific tasks as search terms would bias the results as assumptions about possible tasks tackled by SWeMLS must be made prior to knowing the task landscape. Therefore, this query focuses on application fields. An intersection of all relevant children of (1) ACM Computing Classification System1 concepts *CCS→Computing methodologies→Artificial intelligence*, *CCS→Computing methodologies→Machine learning*, and *CCS→Information systems*, (2) Microsoft Academic2 Topics *Machine Learning*3 and *Artificial Intelligence*4, and (3) Wikipedia categories *Artificial Intelligence*5 and *Machine Learning*6 was taken into account, where children were considered relevant. if they represent an application area.

A.2 Regional Distribution of Publishing Institutes

- **Africa**: morocco (1), sudan (1), tunisia (4)
- **Asia**: china (184), hong kong (1), india (19), indonesia (1), japan (11), pakistan (2), republic of korea (14), singapore (11), sri lanka (3), taiwan (6), thailand (4), vietnam (2)

[17] https://dl.acm.org/ccs, visited Oct 2nd 2020
[18] https://academic.microsoft.com/topics
[19] https://academic.microsoft.com/topics/41008148,119857082, visited Oct 2nd 2020
[20] https://academic.microsoft.com/topics/41008148,154945302, visited Oct 2nd 2020
[21] https://en.wikipedia.org/wiki/Category:Artificial_intelligence, visited Oct 2nd 2020
[22] https://en.wikipedia.org/wiki/Category:Machine_learning, visited Oct 2nd 2020

- **Europe**: austria (1), belgium (3), bulgaria (2), denmark (1), finland (3), france (28), germany (31), greece (5), ireland (6), italy (19), kosovo (1), netherlands (10), norway (2), poland (4), portugal (6), romania (2), russia (5), scotland (1), slovenia (2), spain (11), sweden (1), switzerland (5), united kingdom (24), wales (1)
- **Middle East**: egypt (5), iran (2), palestine (1), saudi arabia (3), turkey (3), uae (1)
- **North America**: canada (15), usa (98)
- **Oceania**: australia (18), new zealand (3)
- South and Central America: brazil (5), colombia (1), jamaica (1), mexico (1)

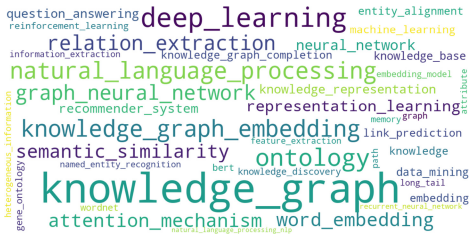

Fig. 37. Wordcloud with the top 40 keywords over all publications.

A.3 Concrete Machine Learning Approaches

In the following, we list the concrete ML approaches grouped by their category and super-category and used in the analyzed papers, accompanied with the corresponding counts. It shall be noted that the low-level approaches are of different granularity, therefore only the categroies are used in the analysis of this review.

- **Classical ML**
  - **BAYES**: bayes (21)
  - **KNN**: knn (21)
  - **SVM**: svm (30)
  - **DECISION TREES**: m5 rules (2), decision trees (19)
  - **MIXTURE MODELS**: gmm (2)
  - **REGRESSION**: regression (17)
  - **MARKOV MODEL**: rbm (1), crf (3), markov model (3)
  - **CLASSIC CLUSTERING**: affinity propagation (1), info map (1), louvain (1), hiearchical clustering (2), dbscan (3), matrix clusterization (5), k means (17)
  - **DIMENSIONALITY REDUCTION**: svd (1), t sne (1), linear discriminant analysis (2), pca (3)
  - **FORMAL CONCEPT ANALYSIS**: fca (3)
  - **RULE LEARNING**: ilp (3), arm (50)
  - **TOPIC MODELS**: lsa (1), lda (8)
  - **SOM**: ghsom (2)
  - **GENETIC ALGORITHMS**: genetic algorithms (11)
  - **FACTORIZATION MACHINES**: factorization machines (3)
  - **VSA**: vsa (1)
- **DL**
  - **TRANSFORMER**: albert (1), electra (1), mcan (1), xlnet (1), transformer (8), bert (16)
  - **LSTM**: elmo (2), bi lstm crf (6)

  - **PLAIN RNN**: attentive bi gru (1), han (1), hopfield network (1), knowledge attention (1), ksr (1), neural attention model (1), rkge (1), rnn gru (1), rsn (1), policy network (2), bi gru (3), rnn (18), lstm (67)
  - **PLAIN CNN**: faster rcnn (1), gated cnn (1), pcnn (3), resnet (3), cnn (76)
  - **GAN**: gan (2)
  - **PLAIN ENCODER**: metapath2vec (1), sentence2vec (1), rdf2vec (4), encoder (5), fasttext (5), deep walk (6), node2vec (8), doc2vec (9), w2v (110)
  - **PLAIN FFNN**: dkrl (1), hole (1), joie (1), literale (1), order embeddings (1), proje (1), sdne (1), simple (1), tare (1), dqn (2), line (4), ntn (4), complex (9), distmult (12), ffnn (83)
- **Graph DL**
  - **MATRIX FACTORIZATION**: analogy (1), concept net (1), dre (1), rdf2vec glove (1), swivel (3), rescal (6), glove (24)
  - **TRANSLATIONAL DISTANCE**: asymmetrical model (1), hyte (1), on2vec (1), poincare (1), ukge (1), tce (2), transx (101)
  - **RECURRENT GNN**: g rnn (1), sse (1), ggnn (3), gnn (7)
  - **CONVOLUTIONAL GNN**: conve (4), gat (6), gcn (20)
  - **GRAPH FFNN**: gram kge (1), graph2seq (1), senn (1)

A.4 Concrete Semantic Web resource

In the following, we list the concrete Semantic Web resource used in the analyzed papers, accompanied with the corresponding counts. It shall be noted that these only cover the predefined resources used, i.e., custom resources are omitted.

- **General**
  - **Benchmarks**
    - *FB Graph BM:* fb-500k (1), fb122 (1), fb24k (1), fb2m (1), fb5m (2), fb13 (15), fb15k-237 (25), fb15k (45)
    - *WordNet BM:* wn-100k (1), wn36 (1), wn11 (10), wn18rr (15), wn18 (29)
    - *Wikidata BM:* wiki-one (1), wikidata12k (1)
    - *DBPedia BM:* db111k-174 (1), dbpedia500k (1), wk31-15k (1), dbpedia50k (2), dbp15k (4)
    - *NELL BM:* nell-1m (1), nell-50k (1), nell79k (1), nell186 (2), nell-one (3), nell-995 (4)
    - *SemEval BM:* semeval-2010 task 8 (1)
    - *OAEI BM:* oaei-benchmark (2)
    - *YAGO BM:* yago11k (1), yago15k (1), yago26k-906 (1), yago3-10 (1)
  - **Cross Domain:** framester (1), google knowledge graph (1), imagenet (1), kb4rec (1), lcsh (1), nell (1), shenma (1), sumo (1), microsoft satori (3), wikipedia (7), conceptnet (10), yago (18), freebase (19), wikidata (20), dbpedia (56)
  - **Dictionaries**: babelnet (1), framebase (1), indowordnet (1), odenet (1), odp (1), ruwordnet (1), ppdb paraphrase dataset (2), sensigrafo (2), probase (4), wordnet (32)
  - **Others General**: ace 2005 (1), baidu ske (1), cifar-100 (1), duie2.0 dataset (1), icews 05-15 (1), icews .14 (1), infuse ontology (1), nist tac-kbp (1), webnlg (1), mutag (2)
- **Natural Sciences**
  - **Archeology**: package-slip knowledge graph (1)
  - **Biology / Biomedicine**: arabidopsis hormone database 2.0 (1), cl (1), enzymekg (1), gnbr (1), kegg (1), nhc ontology (1), pato (1), po (1), pr (1), pto (1), so (1), ncbi taxonomy (2), ontobiotope (2), goa (3), knowlife (3), go (19)
  - **Chemistry**: chemical complex ontology (1), rex (1), chebi (6)
  - **Geology**: geoonto (1), bgs (3)

Table 5. Brief descriptions of the main concepts of the SWeMLS Classification

`System`	Systems that rely on Semantic Web resources and Machine Learning components (SWeMLS).
`Pattern`	A concept to represent the overall processing flows and the roles of the SW and ML modules in a SWeML system. We use the boxology notation framework [34] to define the interaction patterns between these modules.
`SystemComponent`	A concept adapted from the term "*Algorithmic Module*" from the boxology notation framework. A Processor uses one or more input `Instance`, applies one or more `Model` and produces one or more output `Instance`. Furthermore, a `Processor` could involve additional processes, e.g., data pre-processing.
`Task`	The kind of tasks a SWeML system aims to solve.
`Domain`	The application domain of a SWeML system.
`Model`	A system of postulates, data, and inferences presented as a mathematical description of an entity or state of affairs.
`StatisticalModel`	A statistical model is usually specified as a mathematical relationship between one or more random variables and other non-random variables.
`SymbolicModel`	A conceptual model represents .concepts. (entities) and relationships between them. Semantic technologies formally represent the meaning involved in information. For example, an ontology can describe concepts, relationships between things, as well as their categories.
`Instance`	An example or single occurrence of something.
`Data`	Factual information (such as measurements or statistics) used as a basis for reasoning, discussion, or calculation.
`Symbol`	Something that stands for or suggests something else by reason of relationship, association, convention, or accidental resemblance. An arbitrary or conventional sign used in writing or printing relating to a particular field to represent operations, quantities, elements, relations, or qualities.
`Maturity`	The classification of the maturity level of a SWeML system (cf. Section 5.6.1)
`Documentation`	Document contains information about the transparency and auditability components of a SWeML system (cf. Section 5.6.2, Section 5.6.3).

  - **Medicine / Health**: biological interaction (1), cvdo (1), entrez gene (1), gimi mammography (1), iscn (1), meddra (1), twosides (1), usc (1), ccs (3), drugbank (3), snomed ct (3), hpo (4), do (5), icd (5), mesh (6), umls (14)
- **Human Culture and Education**
  - **Academia / Publications**: aminer (1), dblp (1), microsoft academic graph (1), open academic graph (1), scholarly ontology (1), swrc (1), aifb (3)
  - **Cultural Heritage**: am (1)
  - **Education**: aaup (1)
  - **Movie**: imdb (1), linkedmdb (1), mo (1), movies (1), wikimovies (1)
  - **Music**: albums (1), magnatune (1), musicbrainz (1)
- **Geography and Economics**
  - **ECommerce**: alibaba taobao.s dataset (1)
  - **Finance / business**: forbes (1), lca ontology (1)
  - **Geographical Domain**: cities (1), countries (1), geonames (1), geoquery-880 (1), nations (1)
  - **Tourism**: lonley planet (1)
- **Software / Tech**
  - **Pervasive Computing**: lminws (1)

  - **Software Engineering**: eclipse bugs (1), owls-tc v2.1 (1), s-case ontologies (1)
- **Administration & Politics**
  - **EGovernment**: govwild (1), kinship (3)
- **Other**
  - **Food**: winecloud (2)
  - **Security**: uco (1)

REFERENCES

[1] ACM. 2017. *Statement on Algorithmic Transparency and Accountability*. ACM US Public Policy Council (USACM).

[2] Giuseppe Agapito, Mario Cannataro, Pietro Hiram Guzzi, and Marianna Milano. 2015. Using GO-WAR for mining cross-ontology weighted association rules. *Computer Methods and Programs in Biomedicine* 120, 2 (2015), 113–122.

[3] S. Bader and P. Hitzler. 2005. Dimensions of neural-symbolic integration—A structured survey. *arXiv preprint cs/0511042* (2005).

[4] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, et al. 2020. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115.

[5] Tarek R. Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, et al. 2021. Neural-symbolic learning and reasoning: A survey and interpretation 1. In *Neuro-Symbolic Artificial Intelligence: The State of the Art*, Pascal Hitzler and Md. Kamruzzaman Sarker (Eds.). IOS Press, 1–51.

[6] G. Booch, F. Fabiano, L. Horesh, K. Kate, J. Lenchner, N. Linck, A. Loreggia, K. Murugesan, N. Mattei, F. Rossi, and B. Srivastava. 2021. Thinking fast and slow in AI. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*. 15042–15046.

[7] Pearl Brereton, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software* 80, 4 (2007), 571–583.

[8] C. D'Amato. 2020. Machine learning for the Semantic Web: Lessons learned and next research directions. *Semantic Web* 11, 1 (2020), 195–203.

[9] B. Dandala, V. Joopudi, C.-H. Tsou, J. Liang, and P. Suryanarayanan. 2020. Extraction of information related to drug safety surveillance from electronic health record notes: Joint modeling of entities and relations using knowledge-aware neural attentive models. *JMIR Medical Informatics* 8, 7 (2020), e18417.

[10] Daniel Kahneman. 2017. *Thinking, Fast and Slow*. Macmillan.

[11] D. Doran, S. Schulz, and T. Besold. 2017. What does explainable AI really mean? A new conceptualization of perspectives. In *Proceedings of the 1st International Workshop on Comprehensibility and Explanation in AI and ML.*

[12] A. Garcez, K. Broda, and D. Gabbay 2002. *Neural-Symbolic Learning Systems: Foundations and Applications*. Springer.

[13] Noa Garcia, Benjamin Renoust, and Yuta Nakashima. 2019. Context-aware embeddings for automatic art analysis. In *Proceedings of the 2019 International Conference on Multimedia Retrieval*. 25–33.

[14] Google. 2018. *Artificial Intelligence at Google—Our Principles*. Technical Report. Google. https://ai.google/principles.

[15] Pascal Hitzler. 2021. A review of the Semantic Web field. *Communications of the ACM* 64, 2 (2021), 76–83.

[16] P. Hitzler, F. Bianchi, M. Ebrahimi, and M. Sarker. 2020. Neural-symbolic integration and the Semantic Web. *Semantic Web* 11, 1 (2020), 3–11.

[17] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, et al. 2021. Knowledge graphs. *ACM Computing Surveys* 54, 4 (2021), 1–37.

[18] V. Jayawardana, D. Lakmal, N. de Silva, A. S. Perera, K. Sugathadasa, B. Ayesha, and M. Perera. 2017. Semi-supervised instance population of an ontology using word vector embedding. In *Proceedings of the 17th International Conference on Advances in ICT for Emerging Regions (ICTer'17)*. 1–7.

[19] Henry Kautz. 2022. The third AI summer: AAAI Robert S. Engelmore Memorial Lecture. *NSF Convergence Accelerator* 43, 1 (2022), 105–125.

[20] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for performing systematic literature reviews in software engineering version 2.3. *Engineering* 45, 4ve (2007), 1051.

[21] Hongyu Liu and Bo Lang. 2019. Machine learning and deep learning methods for intrusion detection systems: A survey. *Applied Sciences* 9 (2019), 4396. https://doi.org/10.3390/app9204396

[22] Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

[23] Maria Angela Pellegrino, Abdulrahman Altabba, Martina Garofalo, Petar Ristoski, and Michael Cochez. 2020. GEval: A modular and extensible evaluation framework for graph embedding techniques. In *The Semantic Web: 17th International Conference, ESWC 2020*. Springer, 565–582.

[24] Q. Qiu, Z. Xie, L. Wu, and L. Tao. 2020. Dictionary-based automated information extraction from geological documents using a deep learning algorithm. *Earth and Space Science* 7, 3 (2020), 1–18.
[25] P. Ristoski and H. Paulheim. 2016. Semantic Web in data mining and knowledge discovery: A comprehensive survey. *Journal of Web Semantics* 36 (2016), 1–22.
[26] R. Sapna, H. G. Monikarani, and S. Mishra. 2019. Linked data through the lens of machine learning: An enterprise view. In *Proceedings of the 3rd IEEE International Conference on Electrical, Computer, and Communication Technologies (ICECCT'19)*. IEEE, Los Alamitos, CA.
[27] Iqbal H. Sarker. 2021. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science* 2, 6 (2021), 420. https://doi.org/10.1007/s42979-021-00815-1
[28] M. Sarker, L. Zhou, A. Eberhart, and P. Hitzler. 2021. Neuro-symbolic artificial intelligence current trends. *arXiv:2105.05330* (2021).
[29] A. Seeliger, M. Pfaff, and H. Krcmar. 2019. Semantic Web technologies for explainable machine learning models: A literature review. In *Proceedings of the 1st Workshop on Semantic Explainability (SemEx'19) Co-lLocated with the 18th International Semantic Web Conference (ISWC'19)*. 30–45.
[30] S. Singh and M. Singh. 2018. Semantic Web mining: Survey and analysis. *Journal of Web Engineering & Technology* 5 (2018), 20–31.
[31] D. Song, F. Schilder, S. Hertz, G. Saltini, Ch. Smiley, Ph. Nivarthi, O. Hazai, et al. 2017. Building and querying an enterprise knowledge graph. *IEEE Transactions on Services Computing* 12, 3 (2017), 356–369.
[32] G. Stumme, A. Hotho, and B. Berendt. 2006. Semantic Web mining: State of the art and future directions. *Journal of Web Semantics* 4, 2 (2006), 124–143.
[33] M. van Bekkum, M. de Boer, F. van Harmelen, A. Meyer-Vitali, and A. ten Teije. 2021. Modular design patterns for hybrid learning and reasoning systems. *Applied Intelligence* 51, 9 (2021), 6528–6546.
[34] F. van Harmelen and A. ten Teije. 2019. A boxology of design patterns for hybrid learning and reasoning systems. *Journal of Web Engineering* 18, 1-3 (2019), 97–124. arXiv:1905.12389
[35] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, et al. 2023. Informed machine learning—A taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge & Data Engineering* 35, 1 (2023), 614–633.
[36] J. Voogd, P. de Heer, K. Veltman, P. Hanckmann, and J. van Lith. 2019. Using relational concept networks for explainable decision support. In *Proceedings of the International Cross-Domain Conference for Machine Learning and Knowledge Extraction*. 78–93.
[37] Changchang Yin, Rongjian Zhao, Buyue Qian, Xin Lv, and Ping Zhang. 2019. Domain knowledge guided deep learning with electronic health records. In *Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM'19)*. IEEE, Los Alamitos, CA, 738–747.
[38] B. Zafar, M. Cochez, and U. Qamar. 2016. Using distributional semantics for automatic taxonomy induction. In *Proceedings of the 2016 International Conference on Frontiers of Information Technology (FIT'16)*. 348–353.
[39] L. Zhang, S. Liu, D. Liu, P. Zeng, X. Li, J. Song, and L. Gao. 2021. Rich visual knowledge-based augmentation network for visual question answering. *IEEE Transactions on Neural Networks and Learning Systems* 32, 10 (2021), 4362–4373.
[40] Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2022. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge & Data Engineering* 34, 1 (Jan. 2022), 249–270. https://doi.org/10.1109/TKDE.2020.2981333

Received 29 December 2021; revised 6 December 2022; accepted 14 February 2023