SeLoC-ML: Semantic Low-Code Engineering for Machine Learning Applications in Industrial IoT

Haoyu Ren[1,3], Kirill Dorofeev[1(✉)], Darko Anicic[1], Youssef Hammad[1,3], Roland Eckl[2], and Thomas A. Runkler[1,3]

[1] Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany
{haoyu.ren,kirill.dorofeev,darko.anicic,youssef.hammad,
thomas.runkler}@siemens.com
[2] Siemens AG, Siemenspromenade 1, 91058 Erlangen, Germany
eckl.roland@siemens.com
[3] Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany

Abstract. Internet of Things (IoT) is transforming the industry by bridging the gap between Information Technology (IT) and Operational Technology (OT). Machines are being integrated with connected sensors and managed by intelligent analytics applications, accelerating digital transformation and business operations. Bringing Machine Learning (ML) to industrial devices is an advancement aiming to promote the convergence of IT and OT. However, developing an ML application in Industrial IoT (IIoT) presents various challenges, including hardware heterogeneity, non-standardized representations of ML models, device and ML model compatibility issues, and slow application development. Successful deployment in this area requires a deep understanding of hardware, algorithms, software tools, and applications. Therefore, this paper presents a framework called **S**emantic **L**ow-**C**ode Engineering for **ML** Applications (SeLoC-ML), built on a low-code platform to support the rapid development of ML applications in IIoT by leveraging Semantic Web technologies. SeLoC-ML enables non-experts to easily model, discover, reuse, and matchmake ML models and devices at scale. The project code can be automatically generated for deployment on hardware based on the matching results. Developers can benefit from semantic application templates, called *recipes*, to fast prototype end-user applications. The evaluations confirm an engineering effort reduction by a factor of at least three compared to traditional approaches on an industrial ML classification case study, showing the efficiency and usefulness of SeLoC-ML. We share the code and welcome any contributions (https://github.com/Haoyu-R/SeLoC-ML).

Keywords: Machine learning · Neural network · Industrial internet of things · Semantic web · Knowledge graph · Low-code engineering

1   Introduction

One of the biggest challenges in industrial digitization is to bridge the gap between Operational Technology (OT) and Information Technology (IT). OT is centered on a physical world composed of machines, manufacturing equipment, and other hardware, where a massive amount of data is generated. However, IT is focused on the contemporary digital world, using data centers, servers, and smart applications to consume the data. These two domains have traditionally functioned in isolation [6]. The rise of Industry 4.0, along with increasing connectivity between humans, machines, and sensors, is driving the convergence of IT and OT, shifting data-supported decision-making from the individual to the system level and enhancing factory efficiency. However, IT/OT convergence is difficult to achieve. One example is the deployment of ML on industrial devices, where ML presents the IT world and industrial devices present the OT world.

ML is one of the fast-growing technical advancements. Applying ML, specifically Neural Network (NN), in the industry by leveraging sensor and system data can provide reliable insights into the factory and accelerate smart manufacturing. Standard ML applications transfer massive field data to the cloud and centrally process the data against NN models. Concerns have been raised because this data transfer causes numerous issues, such as high energy consumption and latency, privacy leaks, bandwidth congestion.

With the Internet of Things (IoT) growth, factories will be equipped with increasingly powerful, connected, and intelligent devices. This plays a key role in the continuing industrial evolution. Offloading ML intelligence from the cloud to the Industrial IoT (IIoT) devices enables performing ML tasks near data sources and reducing reliance on data transfer, which addresses the latency and security concerns. However, applying on-device ML in the industry where mass deployment happens is still challenging.

IIoT devices are specialized to fulfill different tasks. They come in all shapes and sizes, differ in terms of onboard sensors, available memory and storage capacities, and have various runtime platforms. In the context of on-device ML, they rely on NN models to interpret sensor data, make predictions about their environments, and take intelligent actions locally. NN models are developed with various structures, e.g., different combinations of layers and individualized pre- and postprocessing blocks. Additionally, most trained NN models are distributed as binary files without a clear and standardized description of their usages. The diversification and proliferation of hardware (IIoT devices) and software (NN models) widen the gap between each other.

Many compatibility issues must be carefully investigated to run ML properly on the devices, such as sensor input format, memory constraints, and sensor availability. Specifically, we want to answer two sets of questions:

1. How do we achieve the co-management of IIoT devices and NN models?
   (a) How do we determine which devices may execute a specific NN?
   (b) Given a device, how do we determine which trained NN model is compatible with it? Does the model meet requirements for accuracy, memory, and latency?

2. How do we accelerate the engineering and deployment of ML applications in IIoT? How might cross-domain collaborations be facilitated and the solution be made accessible to all?

We present a framework called Semantic Low-Code Engineering for ML Applications (SeLoC-ML), a system for managing and deploying ML on devices in IIoT at scale. Here, we propose to use formalized semantic models to describe heterogeneous IIoT devices and NN models, respectively. With ontology schemas, we can model the knowledge about devices and NN models semantically in a unified language and centrally store it in a Knowledge Graph (KG), making the knowledge searchable like Web resources. As a result, many features are enabled, such as vendor-agnostic knowledge discovery and matchmaking model requirements with device capabilities.

Another aspect of our work is deploying NN models on the devices and integrating ML applications into end-user IIoT applications. More crucially, we aim to make the approach accessible and understandable to non-experts. We propose integrating SeLoC-ML into a low-code platform, allowing developers without necessary expertise to use semantic services and deploy ML models declaratively, advancing IT/OT convergence. In the background, user inputs are parsed, and corresponding SPARQL queries[1] are formulated to retrieve the information from the central graph database automatically. Additionally, different deployment options will become available depending on matching results, allowing developers to generate projects and deploy ML on the devices with minimal effort. Last but not least, we leverage semantic application templates, so-called *recipes* [24], to assist developers in integrating ML applications into greater pipelines and creating end-user applications rapidly. We support application development by matching the data types used in *recipes* with the data points provided by the devices, which are defined by common semantic models.

As an example of this approach, we present the solution on a Siemens Programmable Logic Controller (PLC) SIMATIC S7-1500[2] using the Siemens low-code platform Mendix[3] [12]. We demonstrate in Mendix how to search and matchmake an NN model with a SIMATIC S7-1500 Technology Module Neural Processing Unit (TM NPU)[4] connected with an Intel RealSense camera[5]. The goal is to find an trained NN model compatible with the TM NPU for classifying different types of objects on a conveyor belt. Following a successful match, an engineering project for Totally Integrated Automation (TIA) Portal[6] - a Siemens

---

[1] https://www.w3.org/TR/rdf-sparql-query.

[2] https://new.siemens.com/global/en/products/automation/systems/industrial/plc/simatic-s7-1500.html.

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

[4] https://new.siemens.com/global/en/products/automation/systems/industrial/plc/simatic-s7-1500/simatic-s7-1500-tm-npu.html.

[5] https://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html.

[6] https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal.html.

Integrated Development Environment (IDE) for industrial automation - can be automatically created and ready for deployment. Later, we present how to use a semantic *recipe* to orchestrate the ML application and easily build an end-user application in Mendix to monitor the classification results. The evaluation results show that SeLoC-ML can reduce engineering effort by a factor of at least three compared to conventional approaches.

The remainder of this paper first presents related work on ML in IoT, Semantic Web technologies, ML management, and low-code programming in Sect. 2. Section 3 describes SeLoC-ML from the semantic system to low-code platform integration. Section 4 demonstrates SeLoC-ML on an industrial classification problem using Siemens products as an example. In Sect. 5, we evaluate the approach, compare it with the traditional workflow and provide the benefits of SeLoC-ML. Finally, Sect. 6 concludes the paper and discusses future work.

2   Related Work

Advancements of On-Device ML. On-device ML is more than just an algorithm. It is about the proliferation of hardware, progress on algorithms, emerging ecosystem, and transformative applications. Ultra-low-power devices have been designed for always-on applications [7,11]. Various algorithms have been proposed to fully exploit ML models on the devices without compromising performance [17,22]. Collaborative ecosystems can further squeeze the potential from the synergism of hardware and software [5,16]. Last but not least, many applications have been brought from a proof-of-concept to products [1,8].

Semantic Web Technologies. Semantic Web technology provides means for building, storing, and handling diverse data sources of different structures, making it an ideal candidate for information modeling and integration in IoT. Evidence has shown the benefits of semantics in industrial domains [13,19]. In the context of IoT, ontologies like Sensor, Observation, Sample, and Actuator Ontology (SOSA) [10], Semantic Sensor Network Ontology (SSN) [3], and Semantic Smart Sensor Network Ontology (S3N) [20] are few prominent semantic models for describing intelligent IoT devices, their properties, and interactions. The Thing Description (TD) [2] ontology developed by the World Wide Web Consortium (W3C) Web of Things (WoT)[7] working group specifies the metadata and interfaces of IoT devices. *iotschema.org*[8] is yet another semantic model in the IoT domain, which is used to enrich the data among connected things. This study is interested in combining these semantic schemas with our proposed NN model ontology [18] for modeling the heterogeneous knowledge about devices and NN models in IIoT.

---

[7] https://www.w3.org/WoT.
[8] http://iotschema.org/.

Management of On-Device ML. There are hundreds of billions of IoT devices today, and new ML models are developed and distributed daily. To manage these resources at scale, it is necessary to increase the interoperability and transparency of the ecosystem. Open Neural Network Exchange (ONNX)[9] aims to provide a shared exchange format that allows developers to use ML models across different deep learning frameworks. However, it fails to provide descriptions of models in a formal way. To overcome this limitation, TensorFlow Lite Metadata[10] and Model Card [14] are introduced to formally document ML models. Few databases [15,26] are introduced for tracking ML models. Nevertheless, they do not scale well since many manual works are required, and their information models do not express the relationships between ML models and hardware. ML models need to be studied together with the specifications of hardware to achieve joint management.

Low-Code Engineering. Despite remarkable IT/OT integration achievements, the current state of developing complex IIoT applications is still far from satisfactory [25]. The concept of low-code engineering and corresponding platforms, such as Mendix [12], are introduced to support fast application development without a prerequisite of having enhanced coding skills [9]. Low-code concepts find their applications in the manufacturing domain [21,27], allowing to quickly build industrial applications based on the services provided by the machines on the shop floor. To match the business requirements with the existing functionalities of the machines and compose them meaningfully, we use the notion of *recipes* [24] as an easy way to model such compositions. *Recipes* can be seen as application templates [23] developed to solve a class of problems and can be later easily configured for a specific use case.

3   Approach

We present SeLoC-ML considering the interoperability and deployment obstacles in ML applications in IIoT. This section starts by introducing the framework setup. The proposed architecture relies on a semantic system designed to cover but not be limited to the use case addressed in this study. We illustrate the semantic system, from the ontology to the semantic services. Next, a simplified KG and two SPARQL queries are presented to exemplify the system's advantages. We then propose integrating SeLoC-ML into the Siemens Mendix low-code platform allowing developers to easily identify and matchmake components, deploy NN models to the devices upon matching, and quickly prototype IIoT user applications.

3.1   Framework Architecture

Figure 1 presents the SeLoC-ML framework. The figure on the left illustrates that developers are faced with a gap between software (NN models) and hardware

---

[9] https://onnx.ai.
[10] https://www.tensorflow.org/lite/convert/metadata.
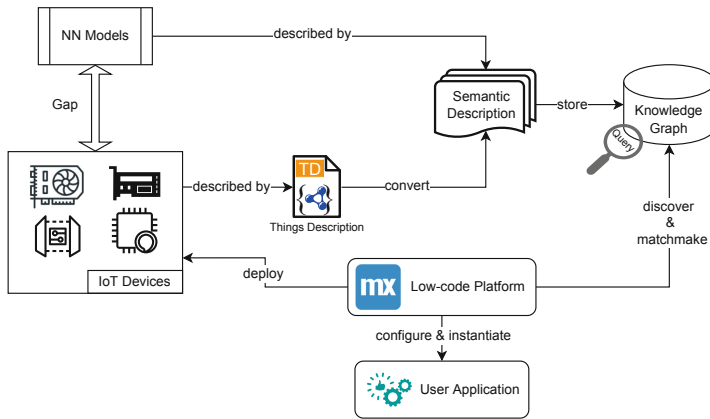
Fig. 1. Framework architecture of SeLoC-ML.

(IIoT devices). We first propose to utilize two semantic models to describe IIoT devices and NN models, respectively. Here, any formalized semantic models can be applied, but for demonstrating the technology in the industrial environment, we choose the standardized W3C TD [2] to describe devices. Aligning with the TD, we design a semantic ontology [18] with a conversion tool for describing NN models in terms of their metadata, structures, and hardware requirements. Thus, knowledge about heterogeneous IIoT devices and NN models can be translated into unified semantic descriptions against their ontologies and be hosted together in a KG, as shown on the right side of the figure. The bottom of the figure shows that even non-experts can easily scrape the KG with Mendix. Mendix will automatically formulate queries based on user inputs and retrieve desired answers from the graph. Upon matchmaking, different deployment options are made available. A ready-to-be-deployed engineering project can be generated based on user configurations and the retrieved semantic information. This is known as ML-as-a-service [4]. Finally, developers can leverage semantic application templates *recipes* to accelerate user application development.

3.2 Semantic System

Ontology. We presented an ontology[11] [17] to describe NN models in the context of IoT, as shown in Fig. 2. By reusing existing schemas, such as S3N and SOSA, we aligned the ontology with other Web standards and avoided reinventing the wheel. For research and demonstration purposes, the ontology has been designed to guarantee its interoperability and compatibility with TD, which we

---

applied to describe IIoT devices. The ontology can render three different forms of information about a NN: 1) metadata, such as the date of creation, category, and literal description; 2) structure, such as the input and output layers; 3) hardware requirements, such as memory and sensors.

As references, we provide interested readers with examples of semantic descriptions of IoT devices and NN models in our repository. Additionally, scripts are available, which can generate a semantic representation of a given NN model along with some user inputs since not all information can be obtained by parsing the NN model, such as dataset and author information.
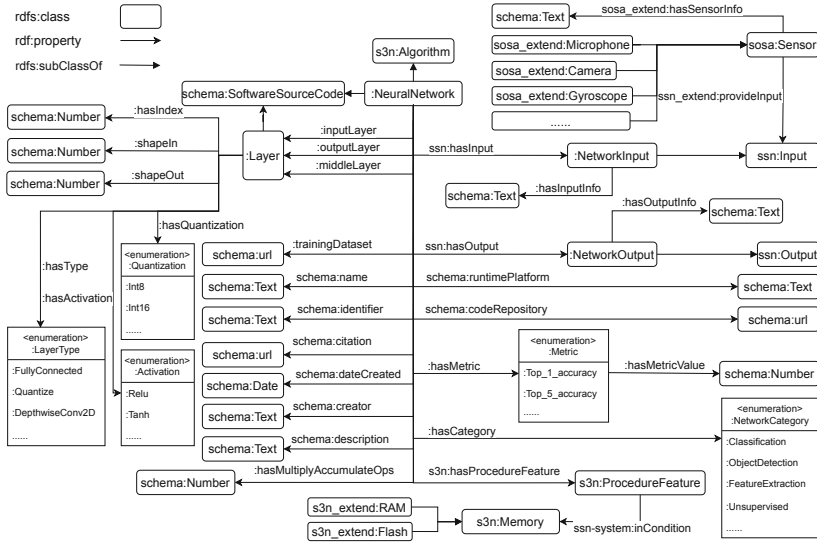

Fig. 2. Ontology of NN model in IoT.

Knowledge Graph. We can design a central KG that stores information from NN models and IoT devices using the semantic schemas introduced above. For an example, we used GraphDB[12] to demonstrate a simplified KG composed of nine IoT devices and 22 NN models. As depicted in Fig. 3, the NN model and device nodes are drawn on the left and right sides of the figure, respectively. An NN model called *Move* is expanded in the center, displaying its properties. We collect NN models trained using TensorFlow, one of the most prominent deep learning frameworks. However, our approach can be easily scaled to cover different devices, NN models, and frameworks. As previously mentioned, we provide the code and examples for creating KG and interacting with it.

---

[12] https://www.ontotext.com/products/graphdb.

Discovery and Matchmaking. Many specific uses and services can be enabled once the KG has been created. We use two simple queries to answer two questions against the KG example introduced above for a demonstration. More queries can be found in our repository. The used namespaces and corresponding prefixes are given as follows:

```
# Our NN ontology
nnet: <http://tinyml-schema.org/
      networkschema/>
# Schema.org Vocabulary
schema: <https://schema.org> .
# Units of Measure Vocabulary
om: <http://www.ontology-of-units-of
    -measure.org/resource/om-2/> .
# SSN Ontology
ssn: <http://www.w3.org/ns/ssn/> .
```

```
# S3N Ontology
s3n: <http://w3id.org/s3n/> .
# Extension of the SOSA ontology
sosa_extend:  <http://tinyml-schema.org/
                sosa_extend#> .
# Extension of the SSN ontology
ssn_extend:  <http://tinyml-schema.org/
               ssn_extend#> .
# Extension of the S3N ontology
s3n_extend:  <http://tinyml-schema.org/
               s3n_extend#> .
```
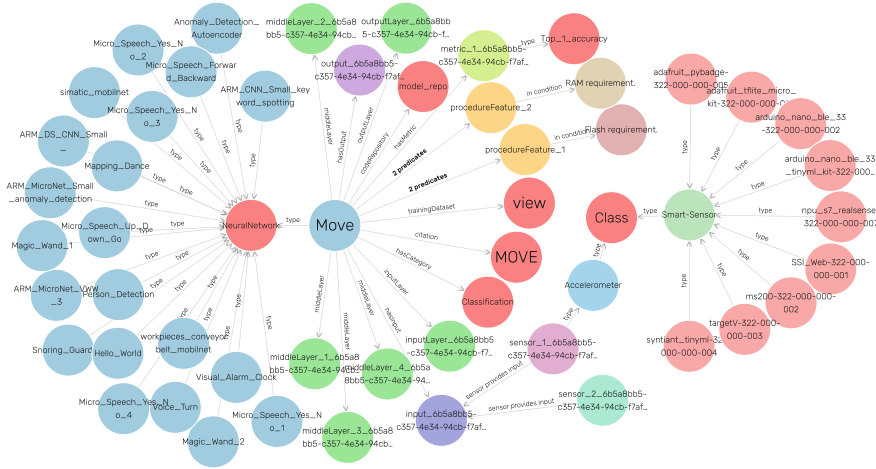

Fig. 3. A simplified KG containing 22 ML models and nine devices.

1. We have an IoT device on which we want to deploy an NN model. The device is equipped with a camera, and it has 144 and 621 Kb of available RAM and Flash, respectively. We want to determine all possible NN models that can be executed on this device.
2. We trained an NN model for motion classification using gyroscope and accelerometer data. Given that the minimum RAM and Flash requirements for running this model are 121 and 610 Kb, respectively, we want to know which available devices can run this model.

```
Query 1:

SELECT ?uuid ?MACs ?RAM ?Flash ?Description
WHERE {
    ?nn a nnet:NeuralNetwork ;
        schema:identifier ?uuid ;
        schema:description ?Description ;
        ssn:hasInput ?input;
        nnet:hasMultiplyAccumulateOps ?MACs ;
        s3n:hasProcedureFeature ?x_1 ;
        s3n:hasProcedureFeature ?x_2 .
    ?x_1 ssn-system:inCondition ?cond_1 .
    ?x_2 ssn-system:inCondition ?cond_2 .
    ?cond_1 a s3n_extend:RAM ;
        schema:minValue ?RAM ;
        schema:unitCode om:kilobyte .
    ?cond_2 a s3n_extend:Flash ;
        schema:minValue ?Flash ;
        schema:unitCode om:kilobyte .
    ?sensor ssn_extend:provideInput ?input;
        a sosa_extend:Camera .
    FILTER (?RAM <= 144)
    FILTER (?Flash <= 621)
}

Result:
uuid: 2c... ; MACs: 7158144; RAM: 94 Kb; ...
uuid: 49... ; MACs: 7387976; RAM: 116 Kb; ...
```

```
Query 2:

SELECT ?Device ?RAM ?Flash
WHERE {
    ?Device a s3n:SmartSensor ;
        ssn:hasSubSystem ?system_1 ;
        ssn:hasSubSystem ?system_2 ;
        ssn:hasSubSystem ?system_3 .
    ?system_1 a sosa_extend:Accelerometer .
    ?system_2 a sosa_extend:Gyroscope .
    ?system_3 a s3n:MicroController ;
        s3n:hasSystemCapability ?x .
    ?x ssn-system:hasSystemProperty ?cond_1 .
    ?x ssn-system:hasSystemProperty ?cond_2 .
    ?cond_1 a s3n_extend:RAM ;
        schema:value ?RAM ;
        schema:unitCode om:kilobyte .
    ?cond_2 a s3n_extend:Flash ;
        schema:value ?Flash ;
        schema:unitCode om:kilobyte .
    FILTER (?RAM >= 121)
    FILTER (?Flash >= 610)
}
ORDER BY ?RAM

Result:
Device: 002; RAM: 172 Kb; Flash: 628 Kb.
Device: 003; RAM: 187 Kb; Flash: 785 Kb.
```
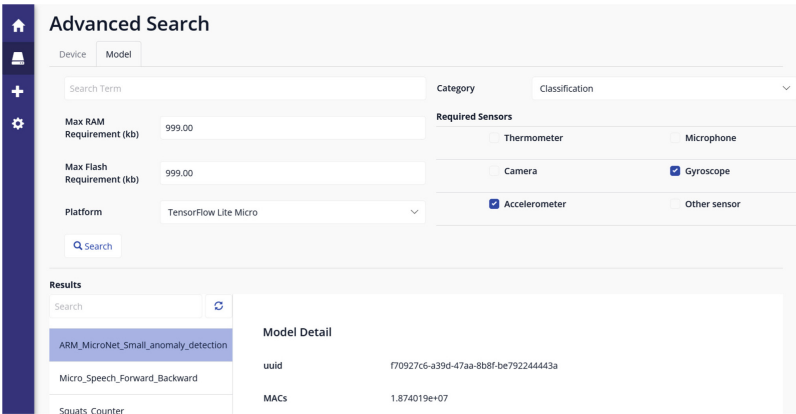

Fig. 4. Semantic similarity search.

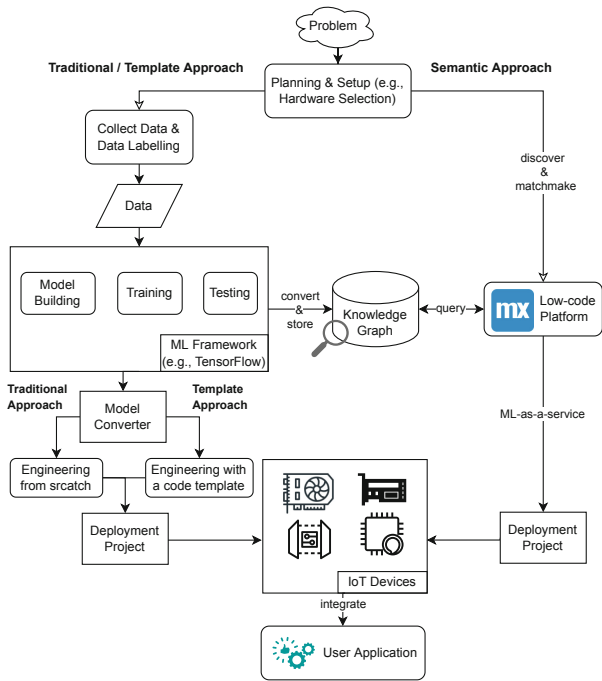### 3.3   Low-Code Platform Integration

Semantic Web techniques are not easy to learn and use. Likewise, on-device ML is another entirely different field that is challenging to understand. To motivate cross-domain collaborations and simplify IT/OT convergence, we encourage integrating SeLoC-ML into a low-code platform - Mendix. Mendix allows developers to design, build, deploy, and operate IoT applications rapidly.

Semantic Management of On-Device ML. We created a Mendix application with a user-friendly Graphical User Interface (GUI) connected with a KG

in the background. The application package is published in our repository. Three main semantic services are provided in the application:

1. **Discovery**: Developers can browse through all available NN models and IIoT devices in the graph database and inspect their details.
2. **Matchmaking**: Once the developer selects an IIoT device/NN model in the application, SPARQL queries are automatically formulated to retrieve all compatible NN models/devices.
3. **Semantic Similarity Search**: Imagine if hundreds of thousands of IIoT devices and NN models were hosted in the KG, it would be tedious to examine them one after another manually. Semantic similarity search[13] enables users to explore relevant objects in the KG by typing a search text, similar to Google Search. In our example, users can search the stored components by filling in the provided form with their requirements, as shown in Fig. 4.

ML-As-a-Service. Moreover, depending on the matched devices, different deployment options become available, and corresponding project code can be generated by parsing the retrieved information. Of course, specific user configurations will be asked to complete the project creation. We aim to provide a high-level abstraction for deployment, which is as hardware-agnostic as possible.

Fig. 5. Comparison between the traditional, template, and semantic approach.

---

[13] https://graphdb.ontotext.com/documentation/standard/semantic-similarity-searches.html.

Rapid User Applications Development. Further, we use Mendix to rapidly prototype an IIoT application using the concept of *recipes. Recipes* provide platform-agnostic application templates that can be easily deployed and configured for common automation tasks. They specify the application logic and the information about input and output required by the tasks [23]. Device data is described using standardized semantic models, such as OPC UA companion specifications[14]. Such a semantic model is stored in the knowledge graph and, at the same time, it is used for developing the *recipes* logic in Mendix. End-user applications can be instantiated by matching the data points provided by system components with the data types required by the *recipe*. As a result, developers quickly orchestrate an end-user application, ready to be delivered.

4   Workflow Comparison and Case Study

This section first outlines SeLoC-ML by comparing it with the State-of-the-Art approaches. Figure 5 presents three different workflows for constructing ML applications in IIoT. Traditionally, after the project planning, ML developers are engaged to engineer an ML model systematically, from data collection and labeling to model building and training. Afterward, embedded engineers take over the work, where the trained model is optimized and converted for the target runtime platform through Model Optimizer. An embedded project is then engineered with the ML model and uploaded to the device. Later, software engineers design a user application to integrate the ML application and report the results to end-users. As can be seen, it is difficult to feature an IIoT ML application that requires cross-domain expertise and a significant amount of engineering work.
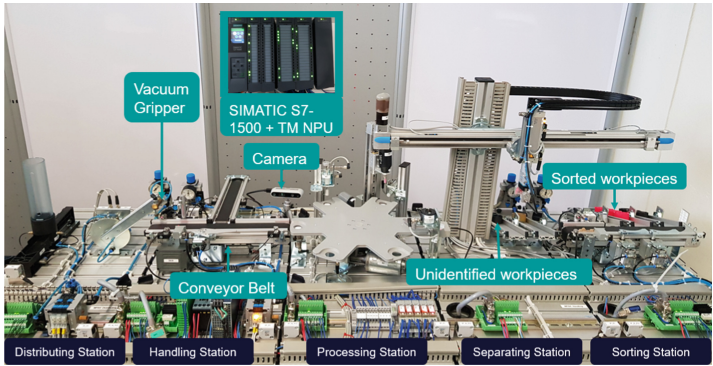
SeLoC-ML offers an all-in-one solution based on Mendix low-code platform to alleviate the situation. SeLoC-ML is generic enough, but for easily quantifying the evaluation and demonstrating its benefits, we illustrate it on an industrial ML classification use case where a NN model is to be discovered and applied on a Siemens TM NPU. Of course, the SeLoC-ML framework can be quickly scaled and applied to other domains and/or use cases.

In-Use: Building an ML Application on Siemens SIMATIC

The case study is conducted on a Festo Didactic workstation[15] controlled by a Siemens SIMATIC S7-1500 PLC, as shown in Fig. 6. In the running example, the vacuum gripper on the left side puts workpieces on the conveyor belt, transporting them to the following process. Different workpieces need to be classified for different downstream handlings, and unidentified objects should be sorted out before the next step. A TM NPU connected with an Intel RealSense camera

---

[14] https://opcfoundation.org/about/opc-technologies/opc-ua/ua-companion-specifications/.

[15] https://www.festo-didactic.com/int-en/learning-systems/process-automation-control-theory.

Fig. 6. Festo Didactic working station controlled by a SIMATIC S7-1500 PLC with a connected TM NPU.

is installed on the workstation, controlled by the SIMATIC S7-1500 PLC. TM NPU enables the execution of ML models directly on Siemens PLCs. Our goal is to leverage Mendix to discover, configure, and deploy an NN model on the TM NPU for classifying workpieces on the conveyor belt using images captured by the camera.

Discovery and Matchmaking. We explore all the reusable ML models in the KG that can run on the SIMATIC TM NPU without spending much time going through the traditional approach and generating a new model from scratch. This can be done in Mendix with a simple click, and all compatible ML models will show up in a pop-up window, as depicted in Fig. 7. After reviewing the results, we select the model *workpieces_conveyorbelt_mobilnet* for our use case.
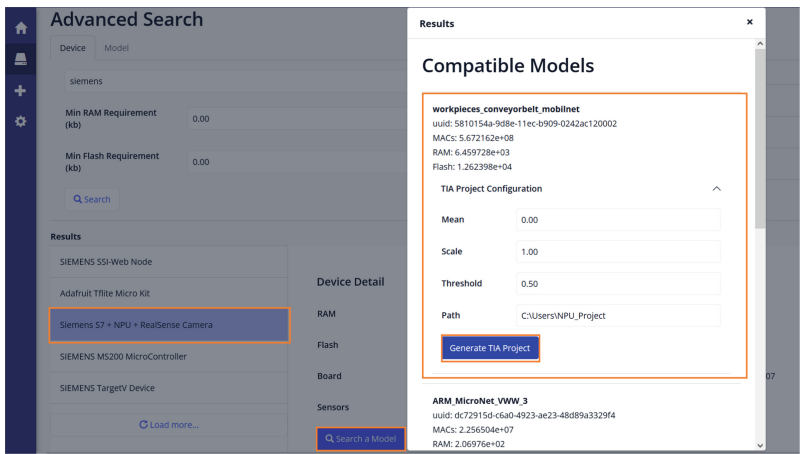
Fig. 7. Discovery and matchmaking results.

Deploying ML on Hardware. After matchmaking, different deployment options become available depending on the selected devices and their runtime platforms. In our case, Mendix creates relevant files for the TM NPU and PLC project. As shown in Fig. 7, specific configurations still need to be given by users, but most of the information in the project is filled automatically by parsing the retrieved semantic descriptions. With all project files loaded to the hardware, the ML application is now ready for execution.

Creating a User Application Using Recipe. The classification results from the TM NPU are made available via an OPC UA server. We created a *recipe* that provides a template for visualizing object classification results based on their color for our running example. To instantiate the *recipe*, Mendix matches the data supplied by the PLC with the data types defined in the *recipe*, based on the definitions given in the OPC UA companion specification. The application developer must acknowledge the match before the application can run. Then, Mendix runtime gets the results of the NN processing, available in the address space of the respective OPC UA server, and represents them in the dashboard, enabling real-time monitoring, as illustrated in Fig. 8.
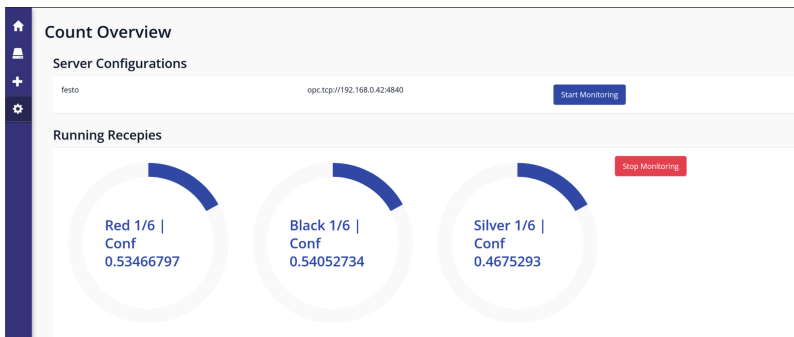

Fig. 8. End-user application that monitors the classification results.

5   Evaluation

This section first compares SeLoC-ML with the traditional approach qualitatively. Further, a quantitative analysis is conducted. For that, the example from the last section is chosen based on our available products to quickly generate results.

5.1   Qualitative Analysis

Reliability and Flexibility. Reliability is one of the essential factors in the industry since a single failure can cause significant losses. Unfortunately, it is

not easy to achieve successful ML applications in IIoT using the conventional workflow since many processes are involved, each of which requires extensive domain knowledge and labor. This can potentially raise the failure probability. SeLoC-ML provides consistent services within one tool that can automate the engineering work by complying with semantic standards, reducing error, and guaranteeing reliability. Moreover, it is important to provide flexible solutions to keep pace with the fast-evolving IIoT world. SeLoC-ML is generic enough to be applied in other scenarios, avoiding examination on a per-instance basis.

Scalability and Interoperability. It is more beneficial to reuse existing ML models than to invent a new one every time since reusability means less cost and better scalability. However, one of the major concerns in IIoT is heterogeneity. Numerous trained NNs could be used for various industrial applications, but it is unclear how to apply them in a concrete use case or on specific hardware. As the IIoT network expands, millions of devices from different vendors emerge, making manual management of a massive amount of hardware and software almost impossible. The proposed approach presents information in a unified language. This ensures that both humans and machines can consistently interpret the stored data and enable automatic development at scale. Besides, our semantic framework is vendor-independent and platform-neutral, enhancing the transparency and interoperability of the ecosystem.

5.2   Quantitative Analysis

We quantitatively evaluate the approach using the conveyor belt example, described in Sect. 4, since this industrial application is representative and similar results have been achieved on other use cases and platforms. We present the experimental results in the following steps: 1) we describe the file structure in the deployment project; 2) we compare the semantic approach (SeLoC-ML) against the other two methods regarding the engineering effort for generating the project, scalability, error rate, and tools required.

A minimum of five engineering artifacts should be engineered and created in the project: a configuration of the ML model used, and a user logic for processing, as well as the corresponding logic in the PLC for exchanging the data with TM NPU and consuming its results, as shown in Table 1.

Table 1. Project files

	Function
npu_app.conf	Configure the ML model on TM NPU
main.py	Configure on-device ML model execution
DataTypes.udt	Define data type(s) for PLC/TM NPU interaction
fbLogic.scl	Define function block to interact with TM NPU
ControlData.db	Define data block to store the data internally in PLC memory

We compare the implementation effort for programming our running example project for three different implementations: 1) traditional approach: programming the whole application from scratch; 2) template approach: providing a user with the pre-developed ready code template that need to be additionally configured for a specific application; 3) semantic approach: SeLoC-ML. We count the number of LOC needed to be manually programmed to implement the running example for every solution. Here, we define LOC to include the number of lines developers need to program and other configuration input that they must provide, for example, the user input in Mendix, as shown in Fig. 7. Table 2 presents the results of measuring the engineering effort.

Table 2. Engineering effort in LOC based on the running example

	npu.conf	main.py	datatypes.udt	fbLogic.scl	dataBlock.db	Total
Traditional approach	20	284	40	408	14	766
Template approach	10	19	3	3	3	38
Semantic approach	4	6	1	1	1	13

Moreover, we studied the flexibility of the approaches in terms of their scalability, i.e., the ability to add new data to the interaction between the PLC and TM NPU. This is especially important when switching between different use cases and/or NNs. Compared to the traditional approach, both template and semantic approaches showed a significant reduction of the LOC. It is worth noting that the semantic approach scales better than the template approach, as we managed to reduce the engineering effort by a factor of three with SeLoC-ML.

Another aspect to consider is the error rate. Getting the most of the code generated will decrease the number of errors made during programming. Once the code generation process is validated, the produced code will be errorless.

Additionally, we consider the number of tools needed to create the entire project for PLC and TM NPU. Using our approach, we generate the entire solution in one place using our Mendix application. Both traditional and template approaches require an engineer to have competencies in at least three different tools: a model converter tool is needed for creating the NN configuration (npu.conf), some IDE for python programming to edit the user logic for NN processing (main.py), and TIA Portal for PLC programming. Table 3 provides an overview of the tools required for each solution.

Table 3. Engineering tools used for programming the running example

	npu.conf	main.py	*.udt, *.scl, *.db	Total
Traditional approach	Model converter	Python IDE	TIA Portal	3
Template approach	Model converter	Python IDE	TIA Portal	3
Semantic approach	Mendix			1

6 Conclusion and Future Work

We have experienced various challenges in implementing ML in IIoT due to the heterogeneity of the ecosystem. This study presents SeLoC-ML for managing and deploying ML on the IIoT devices at scale by leveraging Semantic Web technology. Many out-of-the-box features were enabled using KG, such as knowledge discovery, similarity search, and matchmaking software (NN models) and hardware (devices). By integrating SeLoC-ML in the low-code platform, Mendix, we open new possibilities even for non-experts to easily access these semantic functionalities, use ML-as-a-service for deploying ML models to hardware across the platforms, and prototype end-user applications. The ontology and code examples are available online and can be freely used and further extended.

The next steps, which are already underway, include further improvement and integration of our approach to the production processes. As illustrated in our repository, we have developed SeLoC-ML to support other platforms than in our running example presented in the paper, such as, Arduino[16]. We intend to conduct additional analysis on other scenarios and platforms and collect feedback to further advance the robustness and scalability of our system. We hope to foster the collaboration between the ML and the Semantic Web communities. Therefore, provisioning the framework and making the toolchain available for everyone is also one of our next steps.

References

1. Bejarano-Carbo, A., et al.: Millimeter-scale ultra-low-power imaging system for intelligent edge monitoring (2022). https://doi.org/10.48550/ARXIV.2203.04496, https://arxiv.org/abs/2203.04496
2. Charpenay, V., Käbisch, S., Kosch, H.: Introducing thing descriptions and interactions: an ontology for the web of things. In: 2016 International Semantic Web Conference, pp. 55–66. Springer, Kobe (2016)
3. Compton, M., et al.: The SSN ontology of the W3C semantic sensor network incubator group. J. Web Semant. **17**, 25–32 (2012). https://doi.org/10.2139/ssrn.3198991
4. Doyu, H., Morabito, R., Brachmann, M.: A TinyMLaaS ecosystem for machine learning in IoT: overview and research challenges. In: 2021 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), pp. 1–5. IEEE, Hsinchu, Taiwan (2021). https://doi.org/10.1109/vlsi-dat52063.2021.9427352
5. Duan, S., Xu, X., Ren, S.: A brain-inspired low-dimensional computing classifier for inference on tiny devices (2022). https://doi.org/10.48550/ARXIV.2203.04894, https://arxiv.org/abs/2203.04894
6. Garimella, P.K.: IT-OT integration challenges in utilities. In: 2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS), pp. 199–204. IEEE (2018). https://doi.org/10.1109/cccs.2018.8586807
7. Giordano, M., et al.: CHIMERA: A 0.92 TOPS, 2.2 TOPS/W edge AI accelerator with 2 MByte on-chip foundry resistive RAM for efficient training and inference. In: 2021 Symposium on VLSI Circuits, pp. 1–2. IEEE, Kyoto, Japan (2021). https://doi.org/10.23919/VLSICircuits52068.2021.9492347

---

[16] https://www.arduino.cc/.

8. Gomez, J., et al.: Distributed on-sensor compute system for AR/VR devices: a semi-analytical simulation framework for power estimation (2022). https://doi.org/10.48550/ARXIV.2203.07474, https://arxiv.org/abs/2203.07474

9. Ihirwe, F., Di Ruscio, D., Mazzini, S., Pierini, P., Pierantonio, A.: Low-code engineering for internet of things: a state of research. In: Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, pp. 1–8 (2020). https://doi.org/10.1145/3417990.3420208

10. Janowicz, K., Haller, A., Cox, S.J., Le Phuoc, D., Lefrançois, M.: SOSA: a lightweight ontology for sensors, observations, samples, and actuators. J. Web Semant. **56**, 1–10 (2019). https://doi.org/10.1016/j.websem.2018.06.003

11. Jiao, B., et al.: A 0.57-GOPS/DSP object detection PIM accelerator on FPGA. In: Proceedings of the 26th Asia and South Pacific Design Automation Conference, ASPDAC 2021, pp. 13–14. ACM, New York, USA (2021). https://doi.org/10.1145/3394885.3431659

12. Litman, M., Field, D.: Mendix as a solution for present gaps in computer programming in higher educationD. AMCIS **1**(1), 1 (2018)

13. Mehdi, G., et al.: Semantic rule-based equipment diagnostics. In: d'Amato, C., et al. (eds.) ISWC 2017. LNCS, vol. 10588, pp. 314–333. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68204-4_29

14. Mitchell, M., et al.: Model cards for model reporting. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* 2019, pp. 220–229. ACM, New York, USA (2019). https://doi.org/10.1145/3287560.3287596, https://doi.org/10.1145/3287560.3287596

15. Nguyen, A., Weller, T., Färber, M., Sure-Vetter, Y.: Making neural networks FAIR. In: Villazón-Terrazas, B., Ortiz-Rodríguez, F., Tiwari, S.M., Shandilya, S.K. (eds.) KGSWC 2020. CCIS, vol. 1232, pp. 29–44. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-65384-2_3

16. Rashid, H.A., Ovi, P.R., Busart, C., Gangopadhyay, A., Mohsenin, T.: TinyM^{2}Net: a flexible system algorithm co-designed multimodal learning framework for tiny devices (2022). https://doi.org/10.48550/ARXIV.2202.04303, https://arxiv.org/abs/2202.04303

17. Ren, H., Anicic, D., Runkler, T.A.: TinyOL: TinyML with online-learning on microcontrollers. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE, Shenzhen, China (2021). https://doi.org/10.1109/ijcnn52387.2021.9533927

18. Ren, H., Anicic, D., Runkler, T.A.: How to manage tiny machine learning at scale: an industrial perspective (2022). https://doi.org/10.48550/ARXIV.2202.09113, https://arxiv.org/abs/2202.09113

19. Rojas, J.A., et al.: Leveraging semantic technologies for digital interoperability in the European railway domain. In: Hotho, A., et al. (eds.) ISWC 2021. LNCS, vol. 12922, pp. 648–664. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88361-4_38

20. Sagar, S., et al.: Modeling smart sensors on top of SOSA/SSN and WoT TD with the Semantic Smart Sensor Network (S3N) modular ontology. In: ISWC 2018: 17th Internal Semantic Web Conference, pp. 163–177 (2018)

21. Sanchis, R., García-Perales, O., Fraile, F., Poler, R.: Low-code as enabler of digital transformation in manufacturing industry. Appl. Sci. **10**(1), 12 (2020). https://doi.org/10.3390/app10010012, https://www.mdpi.com/2076-3417/10/1/12

22. Song, J., Lin, F.: PocketNN: integer-only training and inference of neural networks without quantization via direct feedback alignment and pocket activations in pure C++ (2022). https://doi.org/10.48550/ARXIV.2201.02863, https://arxiv.org/abs/2201.02863

23. Thuluva, A.S., Anicic, D., Rudolph, S., Adikari, M.: Semantic node-RED for rapid development of interoperable industrial IoT applications. Semant. Web **11**(6), 949–975 (2020). https://doi.org/10.3233/sw-200405

24. Thuluva, A.S., Bröring, A., Medagoda, G.P., Don, H., Anicic, D., Seeger, J.: Recipes for IoT applications. In: Proceedings of the Seventh International Conference on the Internet of Things, IoT 2017. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3131542.3131553

25. Udoh, I.S., Kotonya, G.: Developing IoT applications: challenges and frameworks. IET Cyber-Phys. Syst. Theory Appl. **3**(2), 65–72 (2018). https://doi.org/10.1049/iet-cps.2017.0068

26. Vartak, M., et al.: ModelDB: a system for machine learning model management. In: Proceedings of the Workshop on Human-In-the-Loop Data Analytics, HILDA 2016. ACM, New York, USA (2016). https://doi.org/10.1145/2939502.2939516

27. Waszkowski, R.: Low-code platform for automating business processes in manufacturing. IFAC-PapersOnLine **52**(10), 376–381 (2019). https://doi.org/10.1016/j.ifacol.2019.10.060