Article

A Holonic-Based Self-Learning Mechanism for Energy-Predictive Planning in Machining Processes

Seung-Jun Shin [1,*], Young-Min Kim [2] and Prita Meilanitasari [2]

[1] Division of Interdisciplinary Industrial Studies, Hanyang University, Seoul 04763, Korea
[2] Graduate School of Technology and Innovation Management, Hanyang University, Seoul 04763, Korea; yngmnkim@hanyang.ac.kr (Y.-M.K.); dintamio@hanyang.ac.kr (P.M.)
* Correspondence: sjshin@hanyang.ac.kr; Tel.: +82-2-2220-2358

check for updates

Abstract: The present work proposes a holonic-based mechanism for self-learning factories based on a hybrid learning approach. The self-learning factory is a manufacturing system that gains predictive capability by machine self-learning, and thus automatically anticipates the performance results during the process planning phase through learning from past experience. The system mechanism, including a modeling method, architecture, and operational procedure, is structured to agentize machines and manufacturing objects under the paradigm of Holonic Manufacturing Systems. This mechanism allows machines and manufacturing objects to acquire their data and model interconnection and to perform model-driven autonomous and collaborative behaviors. The hybrid learning approach is designed to obtain predictive modeling ability in both data-existent and even data-absent environments via accommodating machine learning (which extracts knowledge from data) and transfer learning (which extracts knowledge from existing knowledge). The present work also implements a prototype system to demonstrate automatic predictive modeling and autonomous process planning for energy reduction in milling processes. The prototype generates energy-predictive models via hybrid learning and seeks the minimum energy-using machine tool through the contract net protocol combined with energy prediction. As a result, the prototype could achieve a reduction of 9.70% with respect to energy consumption as compared with the maximum energy-using machine tool.

Keywords: cyber-physical production systems; self-learning factory; holonic manufacturing systems; machine learning; transfer learning; predictive analytics

1. Introduction

Manufacturing intelligence reinforces real-time understanding, reasoning, planning, and management of manufacturing processes with the pervasive use of sensor-based data analytics and modeling [1]. Such intelligence is nothing new in manufacturing; however, it is not mature despite much effort related to its implementation and utilization over the past decades [2]. Implementing manufacturing intelligence is becoming more important than ever due to the evolution of manufacturing technology (MT) itself and the convergence of MT with Internet of things and cyber-physical systems (CPS).

CPS have been recognized as a cutting-edge technology in implementing machine intelligence in various domains, as CPS are "physical and engineered systems whose operations are monitored, coordinated, controlled and integrated by a computing and communicating core" [3]. The concept of CPS is naturally being deployed to industrial automation in the manufacturing realm, and the manufacturing version of CPS is known as cyber-physical production systems (CPPS). CPPS seek to realize intelligence, connectedness, and responsiveness through autonomous and cooperative objects and sub-systems based on context awareness within and across all levels of production [4].

CPPS can be categorized based on their maturity levels, which consist of: visibility, transparency, predictive capability, and self-optimization [5]. CPPS ultimately pursue the acquisition of self-optimizing ability so that manufacturing machines are directly involved in problem-solving or optimization through autonomous and collaborative decision-making and communication with minimization of human intervention. This self-optimizing ability obviously requires predictive capability at the precedent level. Machines can achieve self-optimization only after they can predict their performance by themselves through learning algorithms and use this ability to enhance the accuracy and robustness of their decision-making through evolutionary learning. Such predictive capability can be realized if the machine can self-learn. Self-learning endows manufacturing systems (especially manufacturing objects like machines, material handling equipment, workpieces, work-in-process and products) with the ability to learn from history for future decisions [6].

From the perspective of CPPS implementation, CPPS require control architecture suites fit for autonomous and collaborative operation and control on manufacturing objects. Holonic Manufacturing Systems (HMS) represent one of the most promising architecture suites, with the same goals as those of CPPS [4]. This coincidence can be demonstrated in the Product–Resource–Order–Staff Architecture (PROSA) reference architecture. This referential architecture is structured to achieve both hierarchical and heterarchical control by employing holons (autonomous and cooperative objects in manufacturing systems) and their holarchy (a system of holons) for efficiency in resource utilization, stability against disturbances, and flexibility during changes [7]. To pursue manufacturing intelligence, we suggest that holonic-based systems should be reshaped to obtain learning ability within the complex and dynamic nature of manufacturing environments. Even good stationary structures and mechanisms can hardly accommodate huge numbers of manufacturing conditions which are rapidly changing. Without learning, it is extremely difficult to identify concrete behaviors and activities that will improve the performance of manufacturing systems [8].

Traditionally, self-learning largely depends on creating predictive models derived by machine-learning techniques (e.g., regression, decision tree, Artificial Neural Network (ANN), support vector machine, and genetic algorithms). Machine-learning techniques are used to acquire the knowledge needed to make future decisions from historical training examples [9]. It is known that they enhance the validity of machine-specific models by using real and historical data even in dynamic and complex manufacturing environments. Machine-learning determine cause-and-effect relationships from the training datasets that have been collected from previous manufacturing operations. As cause-and-effect relationships are derived into mathematical representation under certain conditions and constraints, machine-learned models can faithfully work as predictive models by anticipating an effect from an input of cause values.

However, traditional machine learning has a drawback. It does not work unless training datasets exist. Manufacturing environments cannot always create or keep training datasets due to difficult data collection, data loss, data becoming outdated, or even data missing from manufacturing operations that have not been run. Collecting new datasets by performing additional manufacturing operations is desirable; however, it is time-consuming and is sometimes impossible. Nevertheless, the self-learning ability should be obtained and maximized even in such data-absent environments, and transfer learning can be a complementary means of achieving the machine's self-learning. Transfer learning is a technique to extract knowledge from source tasks and apply the knowledge to a target task to reduce the effort required to collect training datasets [10]. As transfer learning involves knowledge extraction from existing models, it allows machines to create knowledge-transferred models in the data-absent environment. Eventually, the adaptive convergence of machine learning and transfer learning enables machines to implement their self-learning ability in both data-existent and even data-absent environments.

In the metal cutting industry, energy consumption becomes a major metric for improving energy-efficiency and environmental performance. According to a survey [11], the manufacturing sub-sectors of fabricated metal products and machinery where the metal cutting industry involves

as a part consume 41,869 and 23,424 million kWh of net electricity in the United States of America, respectively. As these values respectively occupy 5.5% and 3.1% of the entire manufacturing sector, reducing energy consumption in the metal-cutting industry is important for improving environmental performance. In this context, machining in a machine tool affects energy consumption and varies the energy difference in terms of machining power and time by about 66% [12]. Thus, process planning for energy efficiency works as a useful means for reducing the energy consumed during the execution of machine tools because machining sequences and process parameters decided during process planning significantly influence the performance of machine tool operations [13].

Accordingly, much of the literature has elucidated the relational models between process planning decisions and energy consumption based on theoretical and experimental modeling approaches [14]. A theoretical approach uses the theory of metal-cutting with some coefficient assumptions; however, it has limits in predicting energy values correctly due to the gap between assumptive and real coefficient values. An experimental approach can be subdivided into statistical and learning approaches. A statistical approach generates statistical models based on Design of Experiments, which aims at generating response surfaces with a small set of experimental data. This approach derives polynomial equations for energy prediction; however, it only works within the restricted experimental condition. A learning approach uses real data from machining operations for creating machine-learned energy models and shows high accuracy of energy prediction; however, it is limited to creating such models in a data-absent environment, as mentioned above.

In view of the above, a holonic-based approach is necessary to gain the predictive capability through self-learning for reducing energy in machining processes. As holons result from the application of object-oriented concept, they can work as decentralized individuals who independently operate for how-to-create and how-to-use models. These object-oriented holons can adaptively and evolutionarily create learned models based on their associated data and thus can cope with the variability of data, which frequently take place in manufacturing systems due to the changes in manufacturing setup, condition and environment. Furthermore, holons' mutual cooperation via their message exchanges pursues performance optimization centralized on a holarchy. A plausible scenario for energy reduction in machining is that the machines abstracted by holons automatically create energy-predictive models through learning techniques, and predict their energy values using the models. In succession, the machines autonomously and cooperatively make an optimal decision for reducing energy consumption during the process planning phase.

For such purposes, we designed a holonic-based mechanism for self-learning factories based on a hybrid-learning approach. We also implemented a prototype system to perform predictive process planning for energy reduction in milling processes. The hybrid-learning approach is proposed to obtain the ability of self-learned predictive modeling in both data-existent and even data-absent environments via accommodating traditional machine learning and transfer learning. The holonic-based mechanism, consisting of a modeling method, system architecture, and operational procedure, is designed to provide an autonomous and collaborative decision-making environment through the virtual agentization of machines and their associated objects under the paradigm of HMS. This mechanism provides interconnections between data/models and virtual agents. Thus, we can create and apply energy-predictive models automatically on machines with minimal human intervention. The implementation demonstrates how individual machine tools utilize real data or existing models for creating their learned models, predict energy based on their own models, and automatically negotiate between themselves to find the best machine tool that can minimize energy consumption in milling machining.

Section 2 reviews the relevant literature, and Section 3 introduces the concepts of a self-learning factory and hybrid learning. Section 4 presents the holonic-based mechanism. Section 5 demonstrates a prototype system with discussions, and Section 6 summarizes our conclusions.

2. Related Works

This section reviews the literature relevant to HMS and learning ability. Our mechanism builds upon the concept of HMS from a systematic perspective and the application of learning-based analytics from a methodological perspective, respectively.

2.1. Holonic Manufacturing Systems

HMS originated from PROSA, which adopted a holonic organization to achieve stability against disturbances, flexibility in changes, and efficiency in resource utilization [7]. The PROSA identified major keywords as defined below [8].

- Holon: An autonomous and cooperative building block for transforming, transporting, storing, and validating information and physical manufacturing objects. Basic holons consist of product, resource, and order holons, whereas staff holons assist the basic holons.
- Holarchy: A system of holons that cooperates to achieve a goal. It defines the basic rules for cooperation of holons, thereby limiting their autonomy.
- Autonomy: The capability of a holon to create and control the execution of its own plans and strategies.
- Cooperation: A process whereby a set of holons develops and executes mutually acceptable plans.

Extensive knowledge of HMS can be found in outstanding reviews including [8,15–17]. The reviews interestingly imply that Multi-Agent Systems (MAS) are a commonly-used and efficient technology to implement HMS due to the suitability of implementing the modularity, decentralization, and complexity of holons and their holarchy [17]. Here, an agent is a computational system situated in a dynamic environment with the capability of exhibiting autonomous and intelligent behavior, while MAS operate the community of interacting agents as a whole [16]. Thus, it makes sense that the conceptual frame of HMS needs to be transformed to programmable outcomes using MAS technology. Note that the present work also adopts this view due to the reasons given above.

Previous literature has attempted to develop and apply holonic-based systems to enhance target Key Performance Indicators (KPI) in broad applications. The following describes the purposes of HMS implementations in individual applications [15].

- Automation: The low- and (or) high-level control architecture to synchronize physical and software control units for flexibility at the machine or shop floor levels.
- Task allocation: Task assignment involving the distribution of tasks to available resources with the use of Contract Net Protocol (CNP), a negotiation procedure between a manager and a set of candidate contractors about the assignment of a task [18]. Task allocation can be a part of planning and scheduling in some sense.
- Fault-tolerance: Detection of failure, diagnosis of failure, and determination of reasonable recovery actions.
- Real-time control: The system control that reacts within precise time constraints, being classified into hard (missing deadline results in catastrophic consequences) or soft (meeting deadline is desirable but missing a deadline will not cause serious damage).
- Planning and scheduling (the application of the present work): Optimal planning and scheduling of available resources in the production or process level.

Table 1 lists the studies that have endeavored to develop holonic-based systems for enhancing the target KPI (e.g., productivity, flexibility, reconfigurability and fault-free operation). Staff holons are designed to efficiently carry out evaluation, mediation, management, and coordination to assist the basic holons, which concentrate on achieving goals. However, the previous studies did not much focus on data-driven modeling methodologies where the process of acquiring data and creating models becomes critical in structuring holons' functionalities and behaviors. As presented in Table 1, the

previous studies are limited in identifying pivotal mediators that can interconnect historical data and create data-driven models with basic holons for predictive process planning. These studies are also limited in specifying operational and negotiating procedures between holons based on their recognition of model-based prediction.

MAS-driven studies have recently contributed to improve energy efficiency in manufacturing in accordance with the increase of energy reduction requirements. Alotaibi et al. developed a MAS prototype to optimize bi-objective functions (energy and tardiness) in a flexible job shop [19]. Marchiori et al. presented a dynamical approach for energy trades in steel production with the use of autonomous software agents [20]. Giret et al. proposed a software engineering approach for designing sustainable intelligent control systems based on multi-agent and holonic principles [21]. However, their studies depend on a deterministic or discrete event method and do not deal with energy-predictive models based on a data-driven method, which can deliver better predictability and adaptability of models at the machine level.

Table 1. A summary of previous literature on Holonic Manufacturing Systems (HMS).

Application	Citation	Objective	Holon Characteristics (Basic/Staff Holon)
Automation	[22]	A logic control system that supports autonomous and cooperative actions on Automated Guided Vehicles (AGV)	- Resource (B) [1]: the set of devices consisting of robots, workpiece suppliers, and conveyors - Order (B): contains the data relevant to top-level holon orders - Product (B): represents the abstraction of constituent parts - Scheduling (S) [1]: provides scheduling data to other holons - Traffic controller (S): surveys AGV traffic and advises AGV holons to sort out conflicts and avoid congestion
	[23]	An adaptive control system that operates using a role-based mechanism to maintain performance even when faced with perturbations	- Holon agent (B): a resource or an intelligent product, which can inform, communicate, decide and act independently - Optimization mechanism (S): a societal or environmental optimization mechanism to ensure global performance
	[24]	A high-level architecture with integration of both behavioral and structural self-organization for adaptive production control	- Product (B): represents products and the knowledge for production - Task (B): manages real-time execution of orders - Operational (B): represents resources (e.g., robots and operators) - Supervisor (S): introduces optimization into the system
Task allocation	[25]	A machine allocation algorithm that employs the genetic algorithm control for schedule generation and the shortest processing time-based Contract Net Protocol (CNP) for schedule negotiation	- Job (B): represents single jobs - Machine (B): represents machines - Job management (S): manages job holons and informs decomposition into decomposition holons - Decomposition (S): allows for cluster generation and scheduling assignment - Cluster (S): groups a number of machine holons - Genetic Algorithm (S): determines job process sequences in a meta-heuristic method
	[26]	A software design methodology that formulates workflow adaptation problems based on the CNP	- Task (B): represents tasks to be processed - Actor (B): represents workers or customers in the system
Fault-tolerance	[27]	A distributed architecture that performs disturbance handling and predicts mean time between failures	- Resource (B): monitors the device's status for predictive maintenance - Quality control (S): performs detection through quality verification - Task (S): monitors production orders - Supervisor (S): performs meta-monitoring by aggregating the information under holon clusters
	[28]	A service-oriented architecture that operates mechanisms for fault treatment with reconfiguration of dispersed manufacturing	- Product (B): manages requests of products, searches for corresponding holons, and creates a work order - Task (B): manages recipes - Operation (B): represents equipment and humans - Supervisor (S): coordinates the services of all holons

Table 1. *Cont.*

Application	Citation	Objective	Holon Characteristics (Basic/Staff Holon)
Real-time control	[29]	An adaptive control and distributed approach using IEC 61499 function blocks in machining groups	- Product (B): requests manufacturing to planning holons - Planning (B): assigns product operations into machines - Main Control (B): sequences operations and calculates the processing time of production - Operation (B): consists of: control layers to encapsulate agent and function blocks, interface layers to contain virtual simulation and a logical adapter, and machine layers to encapsulate a real machine and its virtual model
	[30]	A holonic hybrid control system that interacts discrete planning and continuous processes	- Product (B): stores and communicates recipes of products - Order (B): determines future planning and holarchy of the system - Resource (B): performs device online control
Planning/scheduling	[31]	An integrated planning and scheduling algorithm that dynamically generates optimal process plans and schedules	- Job (B): represents jobs to be manufactured - Machine tool (B): represents machine tools - Machining process (B): represents machining processes by machine tools - Production engineering (S): initializes the basic holons and modifies job specifications for disturbance handling - Job order (S): represents manufacturing tasks - Coordination (S): determines suitable assignment of job holons using mathematical models
	[32]	A Petri-net based methodology that computes sequential flexibility for dead-lock free planning and scheduling	- Product (B): possesses product's process information - Resource (B): provides manufacturing capability - Order (B): coordinates the execution of production plans - Directory facilitator (S): provides static information on the system as an information server
	[33]	A hybrid metaheuristic-based holonic multi-agent model for flexible job shop scheduling and robot routing	- Robot-system (B): represents robots used for operation - Order-system (B): executes orders within each job - Machine-system (B): represents machine system tools - Scheduler (B): prepares the best promising reasons of the search space - Cluster (B): guides the search to the global optimum solution of the problem
	[34]	A rescheduling mechanism for unpredicted orders and unavailable resource appearance	- Product/Order (B): issues goals according to orders to be solved as managers - Resource (B): represents machines to carry out operations - Contractor (S): performs negotiations among resource holons within the CNP

[1] (B): Basic holon, (S): Staff holon.

2.2. Learning-Based Analytics

Learning ability is recognized as an indispensable feature of manufacturing intelligence [35]. As machine learning inherits the ability to learn, the applications of machine learning dramatically increased in manufacturing domains over the last two decades, and proved suitable in prediction, optimization, control, maintenance, and troubleshooting. This suitability stems from the advantages of machine-learning techniques that handle high dimensional problems, increase the usability of machine-learning practice, discover unknown knowledge, and adapt automatically to dynamic and complex environments [36].

The learning ability has also been incorporated into agent-based manufacturing systems [37]. Kadar and Monostori presented resource/system-level learning to improve the performance of distributed systems by expanding the adaptive characteristics of agents [38]. Shen et al. proposed a learning mechanism for identifying organizational knowledge and selective interaction propagation from emergent system behavior, and it was used for adjusting distributed schedules and planning dynamically [6].

The learning ability has been widely applied into energy-efficient machining as well. Previous works have demonstrated that machine-learning techniques are powerful for predicting and optimizing energy consumption through utilizing the prior knowledge of a concerned system [39]. For example, Garg et al. applied a multi-gene genetic programming approach to generate the model structure and coefficients automatically for energy prediction and optimization in milling machining [40]. Bhinge et al. presented a data-driven approach for energy prediction in milling machining through the application of Gaussian process regression [41], and Liu et al. used a tree-based gradient boosting method, which is a machine learning method to combine weak models into a single strong model in an iterative fashion, to predict specific cutting energy in milling [39].

Despite such efforts, a common problem remains in that values of certain attributes are not available or are missing in the dataset [36]. The recent emergence of transfer learning appears to overcome this problem in manufacturing. Transfer-learning applications are increasing and include fault detection and condition causality in product quality management, fault diagnosis and condition-based maintenance in machine maintenance, and tool tip dynamics prediction in machine chatter [42,43].

Consequently, the motivation of the present work is to develop a HMS mechanism for gaining learning ability, where basic holons can interconnect data, create data-driven models, and determine their behaviors autonomously and collaboratively for energy-efficient machining through predictive process planning. The convergence of machine learning and transfer learning provides a basis for proactive decision-making about the future behaviors of agent-based manufacturing systems, thereby resulting in learning ability in complex and dynamic environments.

3. Self-Learning Factory and Hybrid Learning

This section introduces the concept of a self-learning factory and a hybrid-learning approach, respectively. Section 3.1 explains the conceptual structure and process of a self-learning factory. Section 3.2 describes the theoretical methodology of the hybrid-learning approach.

3.1. Self-Learning Factory

Manufacturing systems operate in dynamic and real-time environments and are frequently confronted with unexpected events such as machine failure. In this circumstance, MAS have been applied to facilitate adaptive, flexible and efficient use of manufacturing resources. However, determining concrete behaviors and activities in MAS a priori is challenging because the following things should be known: the environmental requirements that will emerge in the future, which agents are available, and how those agents need to interact in response to these requirements. Such challenges should be overcome by endowing the agents with the ability to improve the future performance of manufacturing systems through experience [6]. In the present work, a self-learning factory is the

manufacturing system that allows manufacturing objects themselves to learn from past experience, perform predictive simulations and analytics based on the learned-experience, and thus proactively determine their behaviors and activities for improving, sustaining or recovering their target KPI.

Figure 1 presents the concept of a self-learning factory. It consists of a physical and cyber pairwise factory, which mirrors the physical factory and uses virtual agents for representing their physical objects. The cyber factory collects manufacturing data acquired from physical objects. It processes data to generate training datasets and manufacturing context information. Here, the manufacturing context means a machining condition that specifies which machine, material, machining feature, operation and strategy are applied when a certain dataset is generated. The manufacturing context information can be used a model identifier for categorizing the entire training dataset into individual datasets because different process conditions create disparate models. For example, models for a machine need to be different from those of another machine because both machines have different capabilities and performances. It then creates models from training datasets using learning techniques and stores them in a knowledge database (model repository). Here, it can adaptively choose machine learning or transfer learning, depending on whether training datasets exist or not (more details in Section 3.2). The cyber factory makes predictive planning and control decisions based on learned knowledge and models, and eventually feeds such decisions forward to the physical objects located in the physical factory. This cycle repeats, and the cyber factory evolutionarily improves the robustness of knowledge and models, thereby allowing for more accurate planning and control in physical factories.
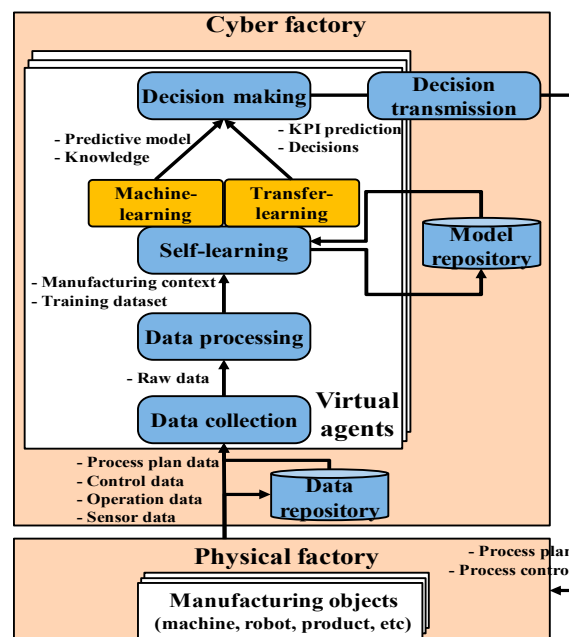

Figure 1. The concept of a self-learning factory. KPI: Key Performance Indicators.

3.2. Hybrid Learning

Manufacturing data are very important because data-driven knowledge creation is the foundation for the self-learning factory. Figure 2 presents data flow on a computer-aided chain in machining processes. Machining processes require part geometries, production plans (macro-level plans for managing a shop floor), process plans (micro-level plans for machining a part), and Numerical Control (NC) programs. Supplementary information like part libraries and machine and cutting tool specifications aid in efficient planning and control by providing technical requirements about products and resources. Here, the data associated with specifications, planning, and control work as causative data because they characterize commands and instructions by which machines must operate. While machine tools run the machining, they generate machine-monitoring data to represent their actions

and movements along with timestamps. After or during machining, inspection equipment records values that are used to check whether machining was satisfactorily completed as designed or not. Machine monitoring and inspection data can be resultant data because they result from the machine's actual operations commanded and instructed by the causative data. Specifically, process plan and NC program data significantly influence machine-monitoring data because a machine tool takes the actions designated by the NC programs, which are outcomes of process plans [44].
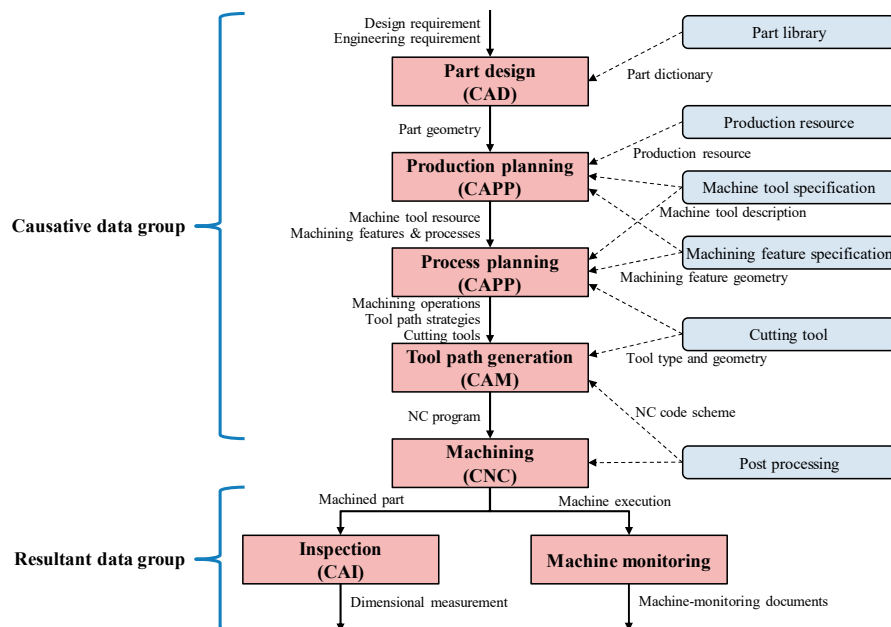

Figure 2. A computer-aided process chain and its data flow. CAD: Computer-Aided Design; CAPP: Computer-Aided Process Planning; CAM: Computer-Aided Manufacturing; CNC: Computerized Numerical Control; CAI: Computer-Aided Inspection.

From a learning perspective, causative data can correspond to input variables (x variables) as training datasets consist of x–y pairwise data instances; meanwhile, the resultant data are included as output variables (y variables) [37]. Additionally, certain causative data are involved in identifying the manufacturing contexts because they specify machining conditions. Hence, process plan and NC program data configure manufacturing context information or x variable data instances, whereas machine-monitoring data are related to y variable data instances. Training datasets can be constructed by integrating machine-monitoring data instances with their corresponding process plan and NC program data instances. Such training datasets are the primary requirements for machine learning and are used to compute their causal relationship by learning techniques.

When implementing the self-learning factory, it is necessary to achieve the self-learning ability through creating predictive models by means of appropriate learning approaches. Predictive models allow machines to forecast KPI under uncertainties, thereby helping the KPI optimization through their self-aware abilities [45]. The traditional learning that uses machine-learning techniques shows excellence at creating predictive models, as reviewed in Section 2.2. However, it does not work unless training datasets exist. To overcome this limitation of the traditional learning, we apply hybrid learning. Hybrid learning can be defined as a learning method where traditional machine learning creates predictive models in a data-existent environment; on the other hand, transfer learning does in a data-absent environment.

Figure 3 presents the concept of hybrid learning for creating energy prediction models in machining processes. Note that our problem is supervised learning because the x and y variables are supervised by humans and desired outputs are supplied during training. When training datasets exist, traditional learning computes a mathematical function, $y = f(X) + \varepsilon$ (ε: error term), based on learning x–y pairwise

training datasets. The upper part of Figure 3 shows how traditional learning is used to create an energy model using an ANN, which is useful for energy prediction in machining [46]. This model can calculate an anticipated energy value based on the input of process parameters (feedrate, spindle speed, and cutting depth) in a certain manufacturing context because ANN makes the $x-y$ relationship numerically known. Such models can provide reliable prediction capability because they build on training datasets that come from real data.
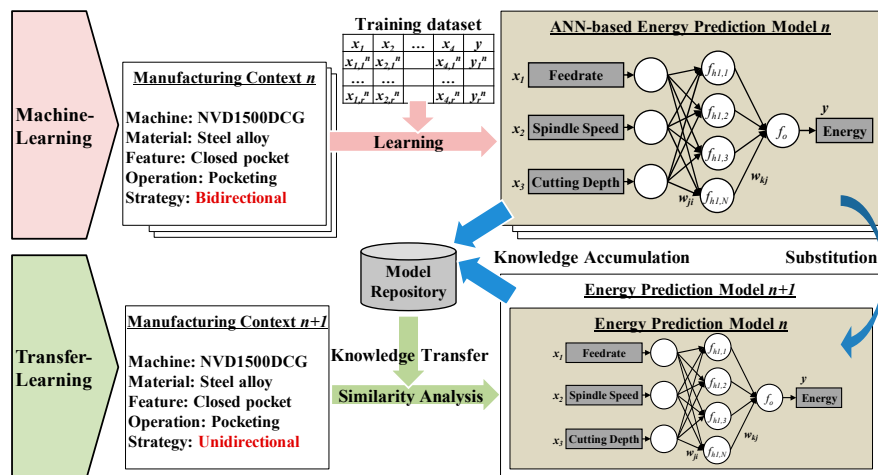

Figure 3. The concept of hybrid learning. ANN: Artificial Neural Network.

Transfer-learning can work when training datasets are unavailable. This transfer learning can create substituent models by transferring learned knowledge (existing models) as it builds upon the similarity between models. A target manufacturing context (target task) captures a substituent model that has the best similarity among existing models (source tasks), as presented in the lower part of Figure 3. Transfer learning unavoidably requires prior knowledge, where the similarity between models has been investigated in a certain manufacturing system (domain). The prior knowledge can be obtained from a preliminary analysis of the target KPI (here, energy). Table 2 shows an example of the similarity of strategies in 2.5 dimensional pocketing machining. This similarity comes from the previous work [47], which observes that unidirectional x-axis up/down milling and unidirectional y-axis strategies have the similar energy pattern in pocket machining due to the dependency of cycle time; on the other hand, bidirectional x-axis, contour, and spiral strategies do. Model similarities can be graded in terms of high, middle or low levels, depending on their energy pattern likeness. When creating a substituent model, one of several models that have a high-level of similarity can be selected and then be substituted for the model that needs to be created (the selection method explained in Section 4.1.2). For example, a contour or spiral strategy model can be substituted for the model of bidirectional strategy due to their high-level of similarity.

The adaptive convergence of machine learning and transfer learning enables self-learning ability regardless of the degrees of freedom in the data. While models are continuously created by hybrid learning, enormous knowledge can be accumulated to ensure predictive capability in a huge number of manufacturing contexts.

Table 2. A similarity matrix of energy patterns in 2.5 dimensional pocketing strategies.

Strategy.	Unidirectional x-axis down Milling	Unidirectional x-axis up Milling	Bidirectional x-axis	Unidirectional y-axis	Contour	Spiral
Unidirectional x-axis down milling		High	Low	High	Low	Low
Unidirectional x-axis up milling	High		Low	High	Low	Low
Bidirectional x-axis	Low	Low		Low	High	High
Unidirectional y-axis	High	High	Low		Low	Low
Contour	Low	Low	High	Low		High
Spiral	Low	Low	High	Low	High	

4. Mechanism

This section presents the mechanism for implementing the self-learning factory based on the hybrid-learning approach. The mechanism includes a modeling method, system architecture and operational procedure, and it focuses on predictive process planning for energy reduction in the cyber part of the self-learning factory.

4.1. Modeling Method

Predictive process planning requires models so that machines make proactive and autonomous decisions through model-based anticipation. The hybrid-learning approach needs to be fully specified because it should be implemented to compile the knowledge needed for automatic creation and use of models. Figure 4 shows high-level methods of the hybrid-learning approach. Sections 4.1.1 and 4.1.2 explain the methods of machine learning and transfer learning, respectively.
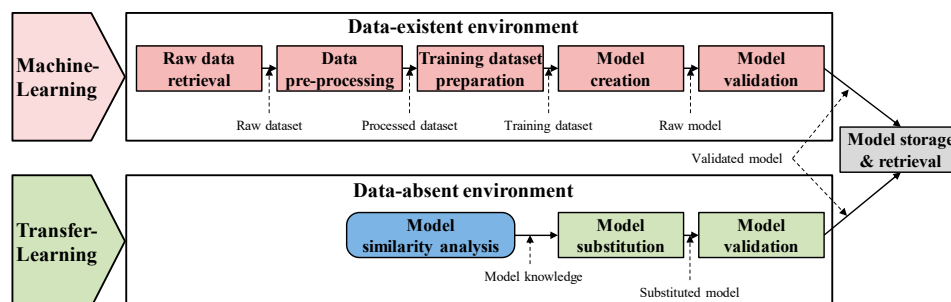
Figure 4. The modeling method of a hybrid-learning approach

4.1.1. Machine-Learning Method

The machine-learning method handles manufacturing data and data-driven models with the use of machine-learning techniques. It consists of: (1) raw data retrieval, (2) data pre-processing, (3) training dataset preparation, (4) model creation, (5) model validation, and (6) model storage and retrieval.

(1) Raw data retrieval involves the search and retrieval of raw data stored in a data repository for collecting data instances in training datasets. As explained in Section 3.2, the process plan, NC program and machine-monitoring data need to be searched and retrieved through a certain search method. A metadata-based search is useful as the metadata indicate the data about the data and serve as a map for locating data instances [48]. Once data instances are tagged with metadata attributes as header information, they can be effectively detected through mapping metadata-tagged data instances with the data queries utilizing the metadata attributes. We design the attributes of the metadata by

considering generality and accessibility. Generality assures that basic information about data instances will be represented across various data formats and dispersed data sources. Accessibility increases the availability of data searches even when some attributes are null. The attributes of the metadata of data can be identified as follows.

　　　Metadata of data = {UUID, Group ID, Creator, Source, Duration, Means, Purpose, Creation}.

- UUID: Universally Unique Identifier.
- Group ID: an identifier for grouping instances.
- Creator: an identifier indicating who creates instances.
- Source: an identifier indicating where instances are stored.
- Duration: a period of time for gathering instances.
- Means: an identifier indicating how instances are obtained.
- Purpose: data attributes to be requested.
- Creation: a timestamp of data creation.

　　　Figure 5 shows an example of the data retrieval using the metadata of data. When a set of raw data associated with process planning (formalized as ISO14649 [49]), NC programing (conforming to Fanuc codes), and machine monitoring (represented by MTConnect [50]) needs to be retrieved, 'O9131' (the NC program name) can work as 'group ID'. In Figure 5a, if the metadata contain 'group ID' and 'purpose', the relevant data instances can be retrieved because 'O9131' (red italic letters) is encoded at 'FILE_DESCRIPTION' in the header section and 'purpose' corresponds to the entity of 'PROJECT' in the data section. In Figure 5b, 'purpose' can request a list of 'CODE BLOCKS' in the NC program named 'O9131'. Figure 5c illustrates the data retrieval from an MTConnect document when data instances regarding 'position' and 'wattage' attributes during a period of time are necessary.

　　　(2) Data pre-processing re-produces high-quality data from raw data and handles them as designated for preparing training datasets. The data pre-processing basically includes data cleaning, integration, transformation, and reduction [51].

　　　Data cleaning resolves missing, noisy, outlying, duplicate, or incorrect data. Raw data unavoidably include sparse, imprecise, faulty, missing, or null data due to the dynamics of manufacturing systems and the limited capability of measurement devices [52]. These uncleaned data cause an increase in data uncertainty and result in negative impact on data-driven learning. Data cleaning produces so-called good data by keeping the data uncertainty under control, thereby increasing the reliability of data-sensitive learning.

　　　Data integration combines heterogeneous data sources or separate formats into a single dataset for the desired learning analysis. For example, data instances retrieved from three different data formats in Figure 5 should be integrated into a tabular training dataset to connect the x and y variables. Data integration can be achieved by the backward tracing that scans from an MTConnect document and an NC program to an ISO14649 program. Here, a key attribute should be identified as the linking point for backward tracing, for which 'position' can be chosen as this key. Since 'position' indicates the coordinates of a cutting tool, a value of 'power' matched with a certain position can be obtained. A NC code block associated with the given position can be traced because the block obviously commands cutting tool movement involving the position. In turn, a machining operation associated with the NC block can be found because the former creates a group of NC blocks where the latter gets involved.

　　　Data transformation converts data instances into the desired format, scale or unit that is more useful for the learning analysis. For example, real data values about feedrate, spindle speed, cutting depth, and power (blue and underlined letters in Figure 5a) need to be adjusted to a 0–1 scale through minimum–maximum normalization due to their different scales. In addition, it is necessary to convert a power unit to an energy unit. This is because a power meter typically measures power values, as shown in Figure 5c, while the y variable in our model uses energy units, which are scalar quantities. We adopt the delta-energy unit, which can be calculated by multiplying power with a sampling rate of

measured power [41]. For example, the sampling rate is given by 0.365 s (the average sampling time of the power meter used in our case study).
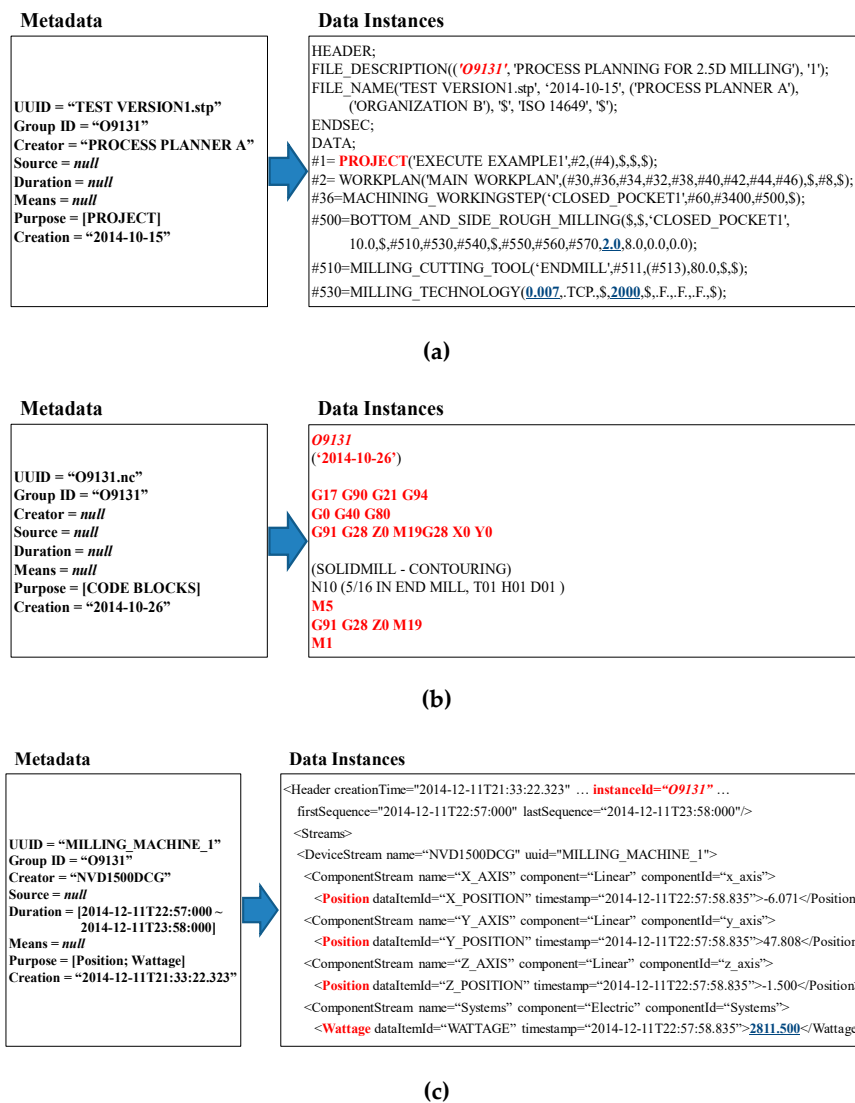
Metadata

Data Instances

```
UUID = "TEST VERSION1.stp"
Group ID = "O9131"
Creator = "PROCESS PLANNER A"
Source = null
Duration = null
Means = null
Purpose = [PROJECT]
Creation = "2014-10-15"
```

```
HEADER;
FILE_DESCRIPTION(( 'O9131', 'PROCESS PLANNING FOR 2.5D MILLING'), '1');
FILE_NAME('TEST VERSION1.stp', '2014-10-15', ('PROCESS PLANNER A'),
    ('ORGANIZATION B'), '$', 'ISO 14649', '$');
ENDSEC;
DATA;
#1= PROJECT('EXECUTE EXAMPLE1',#2,(#4),$,$,$);
#2= WORKPLAN('MAIN WORKPLAN',(#30,#36,#34,#32,#38,#40,#42,#44,#46),$,#8,$);
#36=MACHINING_WORKINGSTEP('CLOSED_POCKET1',#60,#3400,#500,$);
#500=BOTTOM_AND_SIDE_ROUGH_MILLING($,$,'CLOSED_POCKET1',
    10.0,$,#510,#530,#540,$,#550,#560,#570,2.0,8.0,0.0,0,0.0);
#510=MILLING_CUTTING_TOOL('ENDMILL',#511,(#513),80.0,$,$);
#530=MILLING_TECHNOLOGY(0.007,.TCP.,$,2000,$,.F.,.F.,,.F.,$);
```

(a)

Metadata

Data Instances

```
UUID = "O9131.nc"
Group ID = "O9131"
Creator = null
Source = null
Duration = null
Means = null
Purpose = [CODE BLOCKS]
Creation = "2014-10-26"
```

```
O9131
('2014-10-26')

G17 G90 G21 G94
G0 G40 G80
G91 G28 Z0 M19G28 X0 Y0

(SOLIDMILL - CONTOURING)
N10 (5/16 IN END MILL, T01 H01 D01 )
M5
G91 G28 Z0 M19
M1
```

(b)

Metadata

Data Instances

```
UUID = "MILLING_MACHINE_1"
Group ID = "O9131"
Creator = "NVD1500DCG"
Source = null
Duration = [2014-12-11T22:57:000 ~
        2014-12-11T23:58:000]
Means = null
Purpose = [Position; Wattage]
Creation = "2014-12-11T21:33:22.323"
```

```
<Header creationTime="2014-12-11T21:33:22.323" … instanceId="O9131" …
   firstSequence="2014-12-11T22:57:000" lastSequence="2014-12-11T23:58:000"/>
 <Streams>
 <DeviceStream name="NVD1500DCG" uuid="MILLING_MACHINE_1">
  <ComponentStream name="X_AXIS" component="Linear" componentId="x_axis">
   <Position dataItemId="X_POSITION" timestamp="2014-12-11T22:57:58.835">-6.071</Position>
  <ComponentStream name="Y_AXIS" component="Linear" componentId="y_axis">
   <Position dataItemId="Y_POSITION" timestamp="2014-12-11T22:57:58.835">47.808</Position>
  <ComponentStream name="Z_AXIS" component="Linear" componentId="z_axis">
   <Position dataItemId="Z_POSITION" timestamp="2014-12-11T22:57:58.835">-1.500</Position>
  <ComponentStream name="Systems" component="Electric" componentId="Systems">
   <Wattage dataItemId="WATTAGE" timestamp="2014-12-11T22:57:58.835">2811.500</Wattage>
```

(c)

Figure 5. Data retrieval using a metadata-based search (**a**) ISO14649 program; (**b**) Numerical Control (NC) program; (**c**) MTConnect document. UUID: Universally Unique Identifier.

Data reduction may involve the removal of redundant data instances or a reduction in data dimensions to alleviate computational burdens or obtain straightforward learning results. The present dataset forms a data tuple of {feedrate, spindle speed, cutting depth}-{delta energy} at every sampling rate. We need to reduce the data dimension to {feedrate, spindle speed, cutting depth}-{energy} in terms of a manufacturing context because our models seek to output an energy value using the input of certain process parameters in a manufacturing context, as described in Section 3.2. For this purpose, all delta energy values within a manufacturing context are aggregated into a single energy value in the manufacturing context.

(3) Training dataset preparation decomposes the entire pre-processed dataset into individual training datasets separated by manufacturing contexts. Different manufacturing contexts require different models. This comes from the disparate power patterns caused by different cutting force distributions. For example, the power pattern for unidirectional strategy is different from that for bidirectional strategy due to their different tool movements and their different cutting forces. Table 3 presents an example of training datasets in two different manufacturing contexts (bidirectional and

contour machining strategies) within the same pocketing operation. These two datasets are used to create two different models. For example, when the feedrate is set to 0.333, spindle speed to 0.5, and cutting depth to 1, its corresponding energy value equals to 0.188, which is an aggregated value of individual delta energy values consumed by operating the bidirectional strategy for the pocketing. Note that the numerical values are normalized to a 0–1 scale based on original values.

Table 3. An example of training datasets.

Manufacturing Context					x Variables			y Variable
Machine	Material	Feature	Operation	Strategy	Feedrate	Spindle speed	Cutting depth	Energy
NVD1500DCG	Steel alloy	Closed pocket	Pocketing	Bi-directional	0.333	0.5	1	0.188
					0.667	0	0	0.546
					0.667	1	0	0.227
					0.667	1	1	0.000
					0.333	0.5	0	0.796
				Contour	1	0.5	0	0.256
					0.333	0.5	1	0.269
					0.667	0	0	0.386

(4) Model creation involves the generation of predictive models through learning training datasets by machine-learning techniques. As noted in Section 3.2, our model is supervised learning and thus machine-learning techniques can be used to derive mathematical functions that determine the relationship between the *x* and *y* variables. Equation (1) expresses an ANN-based function for energy prediction [53]. Figure 6 shows the structure of an energy prediction model (the graphical structure is presented in Figure 3) and its example where the attributes of an ANN function are instantiated. The manufacturing context enrolls model identification, and the numerical function performs the energy calculation based on the input of the process parameters.

$$y = f_O(\sum_{j=0}^{p} w_{Oi} f_h(\sum_{i=0}^{q} w_{ji} x_i)) + \varepsilon \tag{1}$$

where y: energy, x: process parameter (feedrate, spindle speed, and cutting depth), p and q: the numbers of neurons at each layer, w_{oi} and w_{ji}: weight values, f_o and f_h: activation functions, and ε: learning error
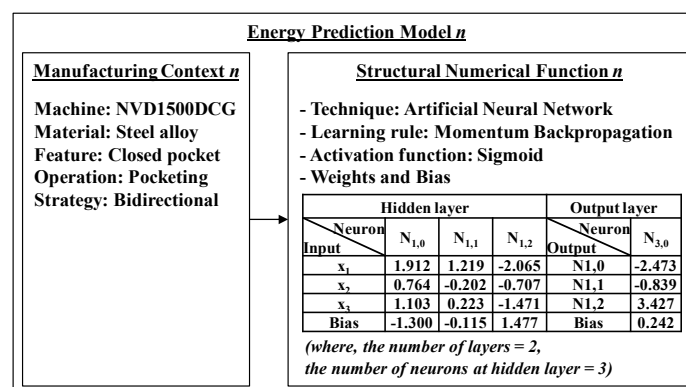

Figure 6. Structure of an energy prediction model.

(5) Model validation involves the quantification of model significance and reliability to validate model conformance. This process checks whether model performance satisfactorily meets a threshold by measuring learning error (the deviation between training data and a numerical function) and prediction error (the deviation between the numerical function and real values to be predicted). Root mean square error (RMSE) is widely used as a performance metric for measuring the learning error [54]. Cross validation is useful for measuring the prediction error. It splits the full dataset into training and test data folds, measures the trained model's performance using the test data fold, and then repeats this procedure by changing the roles of the data folds [54].

(6) Model storage and retrieval involves the storage of validated models in a model repository with their structural forms, and retrieval of the models when requested. Such numerical functions expressed in Equation (1) are quite hard to store in and retrieve from the database. The tabular model representation illustrated in Figure 6 makes this model storage and retrieval efficient. As common relational database systems store and retrieve data records in tabular form, the attributes of ANN functions can be identified as columns and their instances can be recorded as rows in tables. A metadata-based search is also useful as the metadata act as model navigators, as explained in Section 4.1.1 (1). The metadata of a model also need to consider accessibility (as with the metadata of the data) because accessibility is the common sense of storage and retrieval in a database. However, the metadata of the model need to be designed in accordance with specificity because manufacturing contexts depend on and vary with characters of manufacturing systems (e.g., types and complexity of production). The models requested need to be accurately retrieved, and thus the metadata of the model should be able to represent the manufacturing context in a straightforward manner. The metadata of a model for machining processes can be identified as follows.

Metadata of model = {UUID, Group ID, Creator, Source, Means, Creation, Machine, Material, Feature, Operation, Strategy}.

- Machine: an identifier for the machine tool that creates a model.
- Material: an identifier for a workpiece material.
- Feature: an identifier for a machining feature.
- Operation: an identifier related to a machining operation.
- Strategy: an identifier related to a machining strategy that identifies the tool path pattern.

4.1.2. Transfer Learning Method

Transfer learning enables indirect model acquisition through knowledge transfer from existing models when data do not exist, as addressed in Section 3.2. Our method is inductive transfer learning in that the source and target domain (machining process) is the same, but the target task (manufacturing context) is different from the source tasks. On the assumption that the similarity analysis has been investigated, the transfer-learning method consists of: (1) model substitution, (2) model validation, and (3) model storage and retrieval. We skip (3) because it is the same as in Section 4.1.1 (6). We will further discuss the assumption in Section 5.3 (3). It is worth mentioning that traditional machine learning needs to be prior to transfer learning because the former builds on real data, whereas the latter is based on transferred knowledge. When training datasets are available and can be learned, transfer-learned models need to be replaced by machine-learned models to ensure data-driven predictive capability.

(1) Model substitution involves the generation of an alternative energy model by selecting the model that is most similar to the target manufacturing context. Figure 7 shows two methods of model creation: cloning and competing. When there is only one model with a high-level of similarity, the cloning just copies and pastes the original model to a new model, as shown in Figure 7a. When the number of such models is greater than one, the competing is required to choose the best model based on the criteria including default, preference and likeness, as illustrated in Figure 7b. For example, if a new model for a contour strategy is requested and two bidirectional and spiral strategies indicate a high-level of similarity with the former strategy, one of the latter models needs to be chosen. The spiral

strategy model can be chosen if it turns out to be more like a contour strategy model with regard to their machining power distributions.
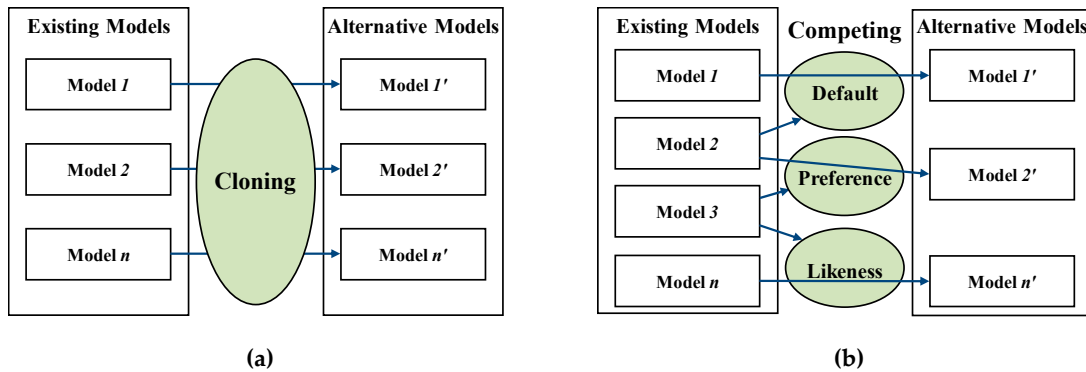

Figure 7. Two methods of transfer learning; (**a**) cloning, (**b**) competing.

(2) Model validation involves quantification of model significance and reliability to validate model conformance. However, validating transfer-learned models is harder because the conformance of a transfer-learned model may not be assured in the target task, although the original model proved to be significant and reliable in the source task. The most obvious method of validation is to measure prediction error by gathering real test data in the target manufacturing context. Reverse validation is recommended when a few of datasets exist for the target manufacturing context [55]. It approximates the difference between the estimated and true conditional distributions in the context of data limitation, although it still requires a minimum dataset at the target task. In reverse validation, a transfer-learned model is re-learned by combining $\{X_s, Y_{s,pred}\}$ (output dataset of the original model) and $\{X_t, Y_t\}$ (real dataset gathered in the target task). In turn, the difference between $Y_{t,pred}$ (output of the new model) and Y_t (real output) is measured to quantify the model approximation for the true conditional distribution (*s*: dataset in the source tasks, *t*: dataset in the target task, *pred*: predicted value).

4.2. System Architecture

Section 4.1 explained the modeling methods and described how to create and use models. It is necessary to identify objects and their functions to allocate such methods from a software architecture perspective. We designed a holonic-based system architecture, as shown in Figure 8. The PROSA architecture underlies this architectural design and can be used to pursue goal-oriented systemization through virtualizing object agents that have autonomous and collaborative capabilities.
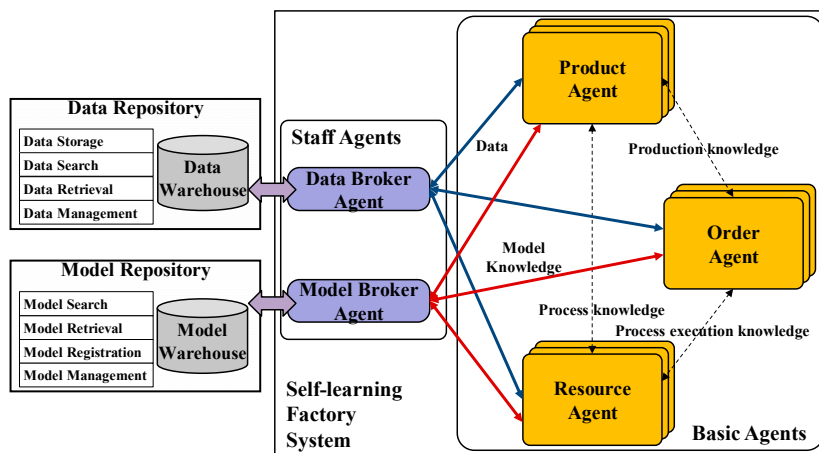

Figure 8. A system architecture for a self-learning factory.

In the PROSA architecture, basic agents consisting of product, order and resource agents mutually exchange production knowledge, process knowledge and process execution knowledge. A product agent performs the functions of request, allocation, confirmation, and supervision of tasks. An order agent is an orchestrator that takes charge of Calls-For-Proposals (CFP), bid evaluation and selection, task allocation, and task progress supervision. Meanwhile, a resource agent receives proposals, checks availability, creates bids, accepts allocations and executes tasks.

In our architecture, data and models should be interconnected within the HMS architecture for integrating the modeling method described in Section 4.1. The proposed architecture thereby gains the capabilities of predictive model-based bid submission for resource agents and predictive value-based bid evaluation for order agents. For this purpose, a data broker agent and a model broker agent are added as staff agents.

The data broker agent acts as a mediator connecting basic agents and a data repository. The data broker thus helps basic agents acquire manufacturing context information, training data and task details. It receives the metadata of data from basic agents when these agents need to gather data to create models or check availability. It returns the resulting data instances to the basic agents through the metadata-based search in the data repository. Meanwhile, the model broker agent is a mediator to connect basic agents and a model repository. It stores models in the model repository once basic agents create models using the acquired data. It searches and returns the models requested by the basic agents when the latter need to use the former. Likewise, the metadata of model is applied to enable the metadata-based search in model requests, searches, and returns.
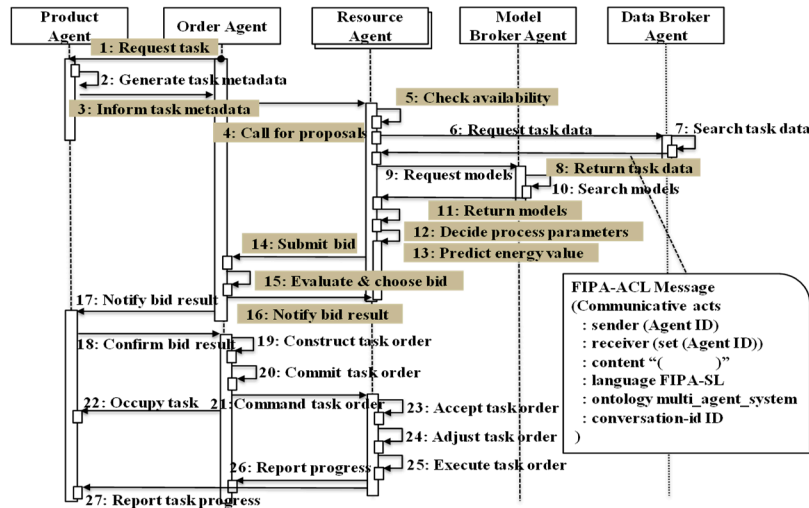
4.3. Operational Procedure

This sub-section describes the operational procedure to specify agents' activities and interactions in a sequential order based on the architecture proposed above. Figure 9 shows the operational procedure represented by a sequence diagram in Unified Modeling Language. This figure is focused on model creation and usage of resource agents.

Figure 9a illustrates the procedure for model creation, substitution and registration to prepare the self-learning ability. If the target model already exists in the model repository (5), this procedure is terminated (7). If not (8.1), the procedure is invoked and starts with a training data request (10). If training datasets are available (13.1), models are created using the machine-learning method (16). If training datasets are not available (13.2), models are substituted from existing models using the transfer-learning method (24). The models created by the two different methods are requested to register (26) and are then registered in the model repository (28).

Figure 9b shows the procedure for model usage to apply the self-learning ability. This procedure builds upon CNP but extends to accommodate the activities and interactions associated with model-based bidding and evaluation. An order agent requests the task taken for fabricating a product (1), and a product agent provides task metadata to the order agent (3). The order agent issues CFP to resource agents (4). The resource agents check their availability with respect to their capability (whether they can fabricate or not) and idleness (whether they are occupied or not) (5). Available resource agents receive the task details (technical specification of the task) using the task metadata (8), and receive models using the metadata of the model extracted from the task metadata (11). They automatically determine process parameters within their allowable ranges and capacities (12). In turn, they anticipate energy values for the task using the models received (13), and then submit their bids where predictive energy values are recorded (14). The order agent evaluates the resources' bids based on energy values (15), and then chooses and notifies the resource agent who submits the minimum energy value (16). The remainder follows the traditional CNP. While agents communicate, they comply with the Foundation for Intelligent Physical Agents–Agent Communication Language (FIPA-ACL), which defines a set of interaction protocols and their individual communicative acts to coordinate multi-message actions [56].
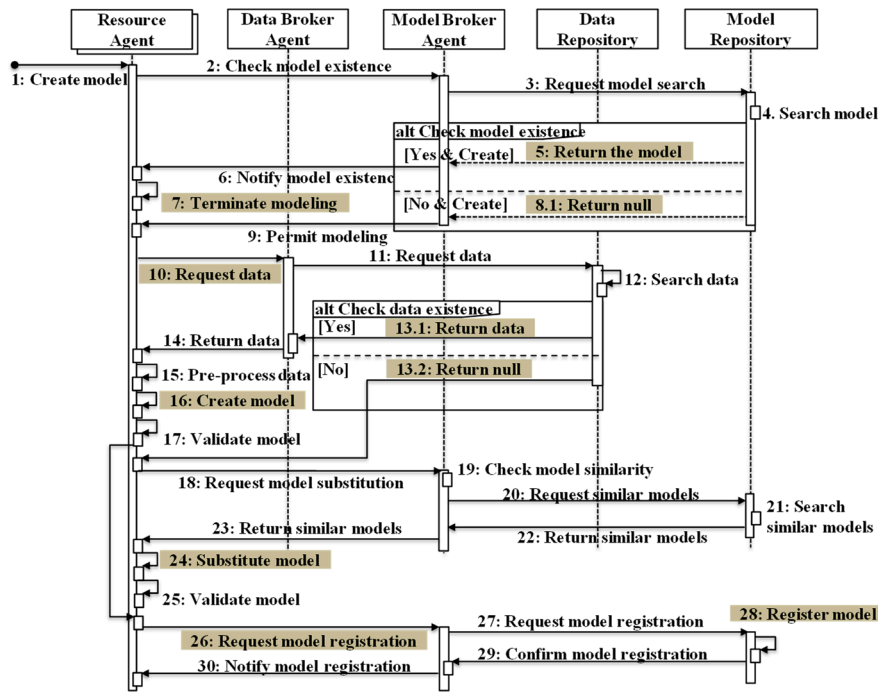
(a)

(b)

Figure 9. An operational procedure for a self-learning factory (**a**) Model creation, substitution, and registration; (**b**) Contract Net Protocol and model usage. FIPA-SL: Foundation for Intelligent Physical Agents – Semantic Language.

5. Implementation

We implement a prototype system to show the feasibility of the self-learning factory. The prototype demonstrates automatic predictive modeling and autonomous process planning for energy reduction in milling machining. Section 5.1 describes implementation scenarios, and Section 5.2 explains prototype implementation. Section 5.3 discusses implementation results.

5.1. Implementation Scenarios

Figure 10 shows a test part containing 13 machining conditions represented by {machining feature; machining operation; machining strategy}. Here, a machining condition corresponds to a

manufacturing context. Implementation scenarios consist of: (1) model creation, substitution, and registration, and (2) CNP and model usage, as explained in Section 4.3.
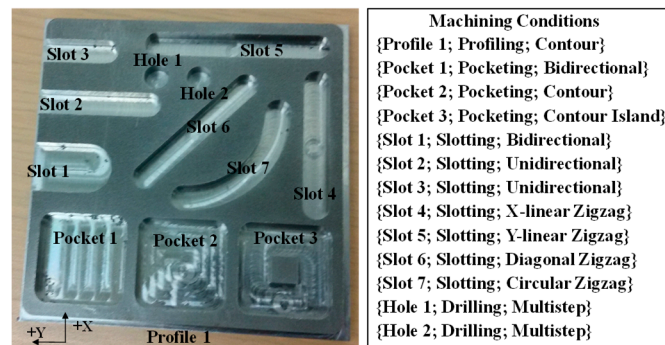
Figure 10. A test part and a set of machining conditions.

Figure 11 presents the two scenarios (the simplification of Figure 9a,b). In Figure 11a, a machine tool generates models using hybrid learning and registers them into the model repository for the next scenario. When the machine asks the model broker for checking the existence of the models associated with the machining conditions, the model broker returns the relevant models if they exist. Otherwise, the machine requests data to the data broker who returns the data requested. If the data exist, the machine creates energy models using machine learning and then requests model registration to the model broker who notifies the registration confirmation to the machine. If the data do not exist, the machine requests similar models to the model broker. The model broker then searches similar models based on model similarity and returns them to the machine. The machine creates alternative energy models using transfer learning and registers the models in the same way. Here, a model for the machining condition {Pocket 1; Pocketing; Spiral}, which is assumedly to be absent in the model repository, is created.

In Figure 11b, a CFP is initiated by an order if a product needs to be machined and informs the relevant tasks. Five machine tools compete for a task. Machines 4 and 5 refuse this task due to their unavailability because Machine 4 is a turning machine and Machine 5 was occupied by another task. The remaining three machines (Machine 1, Machine 2, and Machine 3) decide their process parameters within their allowable ranges or preferences. Process parameters are assumedly determined as follows: Machine 1 (feedrate: 0.0127 mm/tooth, spindle speed: 1750 Revolution per Minute (RPM), cutting depth: 1.5 mm), Machine 2 (feedrate: 0.0178, spindle speed: 2000, cutting depth: 1.0), and Machine 3 (feedrate: 0.0127, spindle speed: 2000, cutting depth: 1.5). They use their energy models to anticipate the energy values consumed during the execution of the given machining conditions. Once their energy values are estimated, the machines send their bids including their predictive energy values to the order. The order evaluates the energy values submitted by the three machines. The order accepts one of the machine tools if its proposal is the minimum energy value.
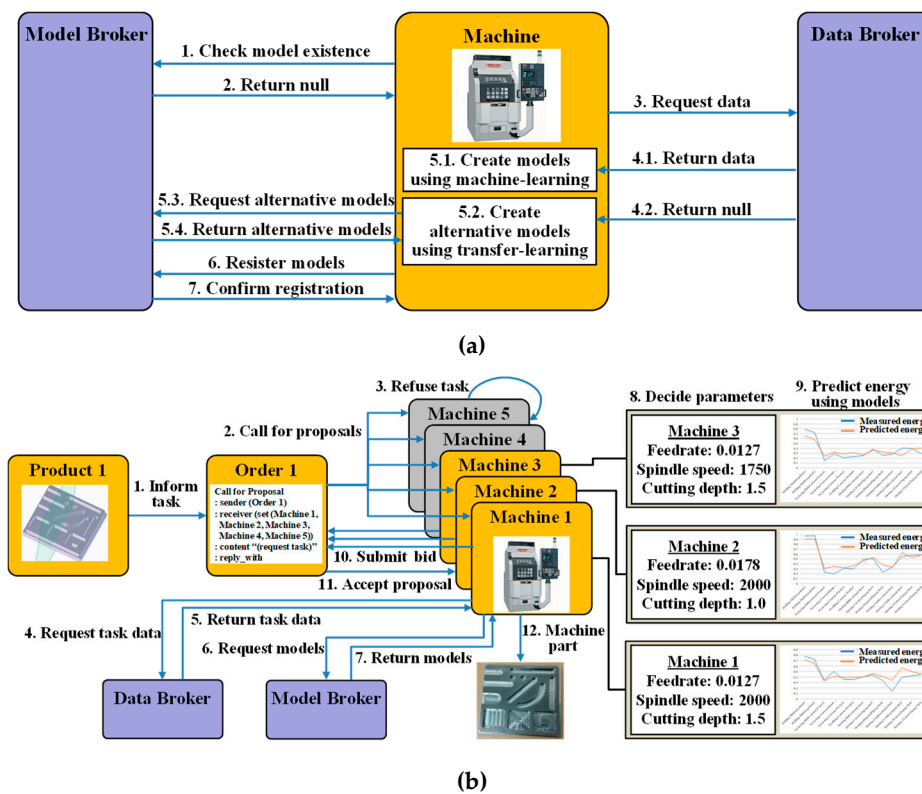
(a)

(b)

Figure 11. Implementation scenarios for (**a**) Model creation, substitution and registration; (**b**) model usage and machine selection.

5.2. Prototype Implementation

For energy modeling, we fabricated 12 test parts in a milling machine. Table 4 presents a list of process parameters for the 12 individual parts. These parameters were randomly determined within the experimental safety and allowable ranges of the machine and cutting tool used. Figure 12 shows an implementation architecture. We generated ISO14649 programs manually, while NC programs were generated using computer-aided manufacturing software, and MTConnect documents were collected in a physical part to represent the machine-monitoring data heterogeneously sourced from an NC and a power meter. The installations of this experiment included the machine (Mori Seiki NVD 1500 DCG), NC (Fanuc 0i), workpiece (Steel 1018, 10.16 cm × 10.16 cm × 1.27 cm), cutting tool (solid carbide flat-end mill, 8-mm diameter, four flutes), and power meter (high-speed power meter from System insights). Note that only one machine is used due to our experimental limitation. We implemented a prototype system in a cyber part based on the mechanism of the self-learning factory, as explained in Section 4. The installations of this implementation included an integrated development environment (Eclipse Java Oxygen), agent platform (Java Agent Development framework (JADE)), JADE execution and deployment (EJADE), data and model repositories (MySQL), and a Java-based ANN framework (Neuroph).

Table 4. List of process parameters.

Trial	Feedrate (mm/tooth)	Spindle Speed (RPM)	Cutting Depth (mm)
1	0.0127	1500	1.5
2	0.0127	2000	1.5
3	0.0127	1750	1
4	0.0229	1750	1
5	0.0127	1750	2
6	0.0178	1500	1
7	0.0178	2000	1
8	0.0178	2000	2
9	0.0178	1750	1.5
10	0.0076	1750	1.5
11	0.0152	1750	1.5
12	0.0127	1750	1.5
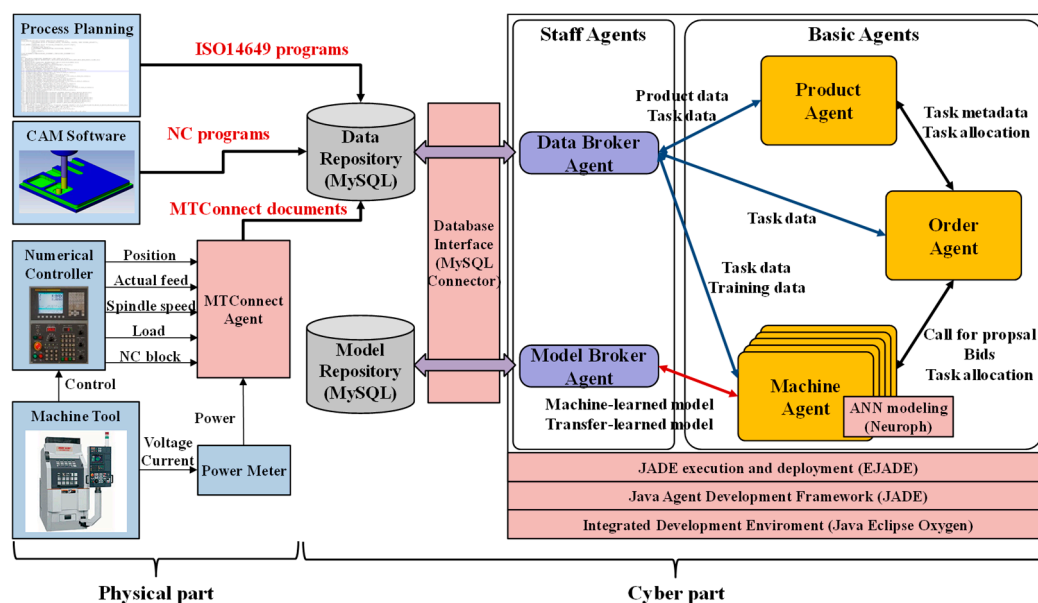
Figure 12. Implementation architecture of the prototype system.

Figure 13 illustrates the screen shots for the implementation of model creation, substitution and registration corresponding to Figure 11a. The screen shots, captured from the JADE sniffer agent, represent FIPA-ACL message exchanges and interactions across individual agents with respect to time, while a computer automatically proceeds (we only click the start button). Note that the arrows only indicate external message exchanges with communicative acts between agents, while internal works inside agents are hidden. In Figure 13a, the machine learning works to create models associated with the 13 machining conditions. The ANN technique is used for this purpose, and the attributes of the ANN-based energy models include (see example in Figure 6) the learning rule (momentum backpropagation), activation function (sigmoid), the number of layers (2), the number of neurons at a hidden layer (3), learning rate (0.3), maximum error (0.01), maximum iteration (1000), and momentum (0.2). As shown in Figure 13b, the transfer learning is activated because the data broker cannot find data in the data repository and then refuses data return. An energy model for {Pocket 1; Pocketing; Spiral}, as described in Section 5.1, is alternatively created through cloning the energy

model for {Pocket 1; Pocketing; Bidirectional} due to their high level of similarity. In these ways, energy models are created and registered in the model repository for the next use.
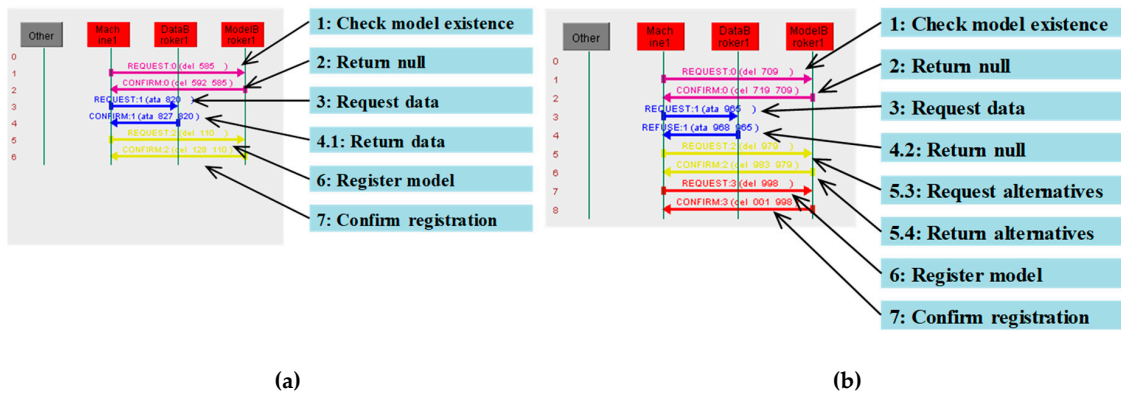

(a)   (b)

Figure 13. Implementation result: model creation, substitution and registration (**a**) Machine learning-based; (**b**) Transfer learning-based.

Figure 14 shows the screen shot for the implementation of CNP and model usage. The order agent is an initiator in the frame of CNP, which comprises an initiator and participants for requesting a task and performing the task, respectively. The order agent communicates with not only the product agent (a participant) for issuing a task but also the machine agents (participants) for assigning the task. The order agent processes its operations aligning with the scheduling of one-shot, cyclic or conditional behaviors for communicating with the participants using FIPA-ACL messages (see Figure 9b). The order agent sends the messages to the target participants and receives the messages from them based on the behavioral scheduling because it can write or read the FIPA-ACL messages that include a sender, receivers, communicative acts (i.e., a tag for communicative acts; e.g., call for proposal, accept/reject proposal, inform and refuse), contents, conversation ID, and so forth.
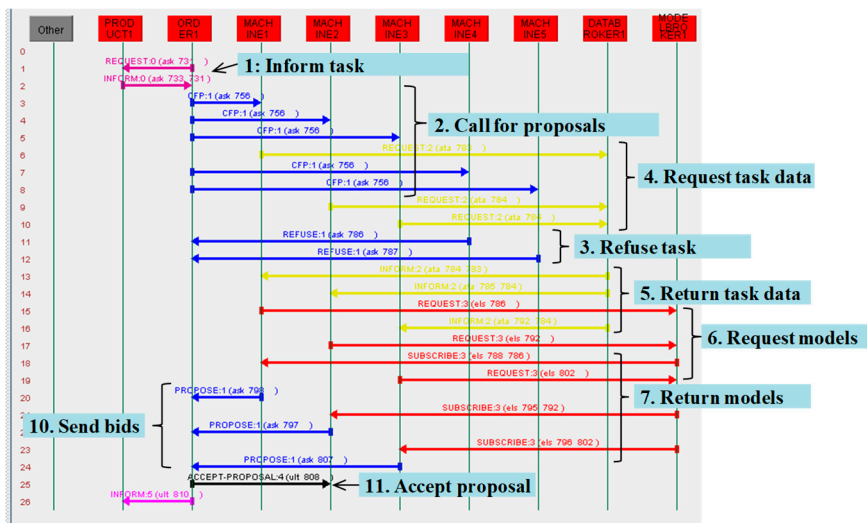

Figure 14. Implementation result: model usage and machine selection.

Order 1 calls for proposals for selecting a machine who can perform the task informed from Product 1. Machines 4 and 5 refuse this task due to their unavailability as explained in Section 5.1. The remaining three machines receive the energy models of the 13 machining conditions and input the determined process parameters. These three machines predict energy values for running the 13 machining conditions and the predictive energy values are, respectively: 11,825 kJ (Machine 1), 11,700 kJ (Machine 2), and 12,957 kJ (Machine 3). These machine agents propose their bids including

these predictive energy values to the order agent. The order agent evaluates their energy values to seek the machine agent who submits the minimum energy value in its bid. Because Machine 2 proposes the minimum energy value, the order agent accepts the proposal from Machine 2, transmits the bidding result and instructs Machine 2 to take the task. This selection of the minimum energy-using machine tool (Machine 2) achieves 9.70% energy reduction, compared with the maximum energy-using one (Machine 3).

We measure real energy values from actual machining to check the accuracy of their corresponding predicted values derived from the ANN-based energy models. The real measured energy values were recorded as 11,382 kJ (Machine 1), 11,044 kJ (Machine 2), and 12,580 kJ (Machine 3). These different values come from the application of different process parameters given in Section 5.1. The total relative error, which measures the percentage of predicted energy values – real energy values)/real energy values, were 3.89%, 5.94%, and 3.00%, respectively.

5.3. Discussion

(1) Experimental limitation: our implementation could select the minimum energy-using machine tool among three machines through predicting and competing their energy values. Using three different machine tools is desirable because individual machine tools make different machine-specific energy values due to the differences in their capabilities and performances. However, a single machine had to be used due to our experimental limitations. It will be more realistic to use different machine tools for creating machine-specific energy models through instantiating different values in the attribute 'machine' of the manufacturing context. Alternatively, transfer learning can be applied to create machine-specific models by reflecting the difference of capabilities and performances between the target and source machines (see an example in the below (3)). In addition, virtual simulators can be useful for limited experimental environments. Some machining simulators can generate machine-specific power values affected by machine's capabilities and performances [57,58].

(2) Increase of practicability: our implementation has been made within a single order on a single process. This may be far from the reality in common manufacturing systems where multiple processes deal with various products and orders. Thus, gaining practicability remains critical. In other words, the target application is demanded to extend toward production planning considering multiple products and orders in a production line. It is expected that the adoption of MAS technology makes the practicability achievable because MAS use unique identification and take their autonomous and collaborative actions regardless of the number of product, order and machine agents. The proposed approach needs to be extended to the production planning by adding more product, order, and machine agents, although the difficulty and complexity of implementation increase.

(3) Uncertainty of transfer learning: our implementation shows the feasibility of the acquisition of self-learning ability by machine and transfer learning techniques. Transfer learning creates an energy model for the target manufacturing context {Pocket 1; Pocketing; Spiral}, which was not machined in our experiment, by cloning the energy model for the source manufacturing context {Pocket 1; Pocketing; Bidirectional}. However, we could not quantify significance nor validate conformance of the transferred model because it was not machined just as stated. The model validation of transfer learning remains as a future work. Our transfer-learning approach builds upon the similarity between the two manufacturing contexts and thus the similarity needs to be analyzed and verified in advance. If such similarity is not verified, transfer learning may not work properly and thus need to consider alternative means besides the similarity analysis. Reverse validation needs to be applied if a few of datasets are generated in the target manufacturing context, as described in Section 4.1.2 (3). Otherwise, the properties that characterize a difference between target and source manufacturing contexts can be added as variables of transfer learning. For example, an energy model for a machine tool can be transferred from that for another machine. The former model should be different to the latter one because these machines have different property values in basic power, rotation torque, and motor efficiency, which affect power and energy in machining. These properties can be additional input variables in the

structure of machine-learning models so that transfer learning can derive machine-dependent results through learning the influence of different values of those properties.

(4) Implementation challenge: energy-efficient machining has become a massive trend in some countries; however, it is still far from reality in other countries where many small-and-medium sized manufacturers fabricate products with paying cheap industrial electricity costs. These countries deal with time and quality as critical performances and are less concerned with energy consumption because they regard the energy cost as an endurable expense. Nevertheless, researchers need to keep their efforts on implementing and deploying cost-effective and data-accessible solutions for energy-efficient machining as the metal-cutting industry affects a large portion of the total energy consumption over the world. The use of open sources helps increase cost effectiveness for implementing such solutions. The implementation tools that we used in the prototype system are all open sources, which are publicly accessible without payment (payment may be required for commercial purposes). This implementation strategy can reduce solution development expenses to the reasonable cost level and help the deployment of such solutions toward small-and-medium sized manufacturers. The use of interoperable and open data interfaces comfortably supports the availability of data collection as data are critical for implementing manufacturing intelligence. Recent standardized interfaces including MTConnect and Open Platform Communications–Unified Architecture facilitate data accessibility. These interfaces provide open source tools as well and thus are quite useful for making a data bridge between physical and cyber factories.

6. Conclusions

In the present work, we designed and implemented a holonic-based mechanism for a self-learning factory based on a hybrid-learning approach. The concept of the self-learning factory was proposed to allow manufacturing objects to learn past experience using their real data, to perform predictive analytics and to determine their behaviors and activities for improving a target KPI. The holonic-based mechanism identified a modeling method, system architecture, and operational procedure to implement an autonomous and collaborative prediction environment through the virtual agentization of manufacturing objects under the paradigm of HMS. The hybrid-learning approach was designed to acquire predictive capability independently with the degrees of freedom in the data through the accommodation of machine learning and transfer learning. This hybrid learning can be used to build up a massive knowledge base through the accumulation of models, thereby gaining self-learning ability in manufacturing systems. A prototype demonstrated the feasibility of the proposed mechanism via predictive process planning for energy reduction in milling machining. Autonomous and collaborative activities of manufacturing agents are carried out on a computer to select the minimum energy-using machine tool while minimizing human intervention.

The limitations of the present work are as follows: (1) Our target is limited to process planning for a single product and process and thus cannot ensure the feasibility of the proposed mechanism in a more complex production line, (2) our implementation is restricted by the use of a single machine and thus does not embody more realistic scenarios by multiple machine tools, (3) our experiment does not show the validity of transferred models due to our experimental limitations, and (4) our implementation excludes control and feedback of the cyber and physical parts in a real-time manner as CPPS obviously require mirrored synchronization between the both parts. We plan to overcome these limitations in future work.

Author Contributions: All authors conceived the research idea and the methods of this study. S.-J.S. designed the concept and mechanism and implemented the prototype; Y.-M.K. contributed to specifying the mechanism of the modeling methods; P.M. supported the prototype implementation and analyzed the experiment results.

References

1. Davis, J.; Edgar, T.; Porter, J.; Bernaden, J.; Sarli, M. Smart manufacturing, manufacturing intelligence and demand-dynamic performance. *Comput. Chem. Eng.* **2012**, *47*, 145–156. [CrossRef]
2. Zuehlke, D. SmartFactory—Towards a factory-of-things. *Annu. Rev. Control* **2010**, *34*, 129–138. [CrossRef]
3. Rajkumar, R.; Lee, I.S.; Sha, L.; Stankovic, J. Cyber-physical systems: The next computing revolution. In Proceedings of the 47th ACM/IEEE Design Automation Conference, Anaheim, CA, USA, 13–18 June 2010; pp. 731–736.
4. Monostori, L.; Kadar, B.; Bauernhansl, T.; Kondoh, S.; Kumara, S.R.T.; Reinhart, G.; Sauer, O.; Schuh, G.; Sihn, W.; Ueda, K. Cyber-physical systems in manufacturing. *CIRP Ann.* **2016**, *65*, 621–641. [CrossRef]
5. Schuh, G.; Anderl, R.; Gausemeier, J.; ten Hompel, M.; Wahlster, W. *Industrie 4.0 Maturity Index—Managing the Digital Transformation of Companies*; ACATECH Study; Herbert Utz Verlag: Munich, Germany, 2017.
6. Shen, W.; Maturana, F.; Norrie, D.H. Enhancing the performance of an agent-based manufacturing system through learning and forecasting. *J. Intell. Manuf.* **2000**, *11*, 365–380. [CrossRef]
7. Brussel, H.V.; Wyns, J.; Valckenaers, P.; Bongaerts, L.; Peeters, P. Reference architecture for holonic manufacturing systems: PROSA. *Comput. Ind.* **1998**, *37*, 255–274. [CrossRef]
8. Shen, W.; Hao, Q.; Yoon, H.J.; Norrie, D.H. Applications of agent-based systems in intelligent manufacturing: An updated review. *Adv. Eng. Inform.* **2006**, *20*, 415–431. [CrossRef]
9. Priore, P.; De la Fuente, D.; Puente, J.; Parreno, J. A comparison of machine-learning algorithms for dynamic scheduling of flexible manufacturing systems. *Eng. Appl. Artif. Intell.* **2006**, *19*, 247–255. [CrossRef]
10. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
11. Manufacturing Energy Consumption Survey. Available online: https://www.eia.gov/consumption/manufacturing/data/2014/index.php (accessed on 24 September 2019).
12. Hu, S.; Liu, F.; He, Y.; Hu, T. An on-line approach for energy efficiency monitoring of machine tools. *J. Clean. Prod.* **2012**, *27*, 133–140. [CrossRef]
13. Kara, S.; Li, W. Unit process energy consumption models for material removal processes. *CIRP Ann.* **2011**, *60*, 37–40. [CrossRef]
14. Shin, S.J.; Woo, J.Y.; Rachuri, S.; Meilanitasari, P. Standard data-based predictive modeling for power consumption in turning machining. *Sustainability* **2018**, *10*, 598. [CrossRef]
15. Babiceanu, R.F.; Chen, F.F. Development and applications of Holonic manufacturing systems: A survey. *J. Intell. Manuf.* **2006**, *17*, 111–131. [CrossRef]
16. Monostori, L.; Vancza, J.; Kumara, S.R.T. Agent-based systems for manufacturing. *CIRP Ann.* **2006**, *55*, 697–720. [CrossRef]
17. Leitão, P. Agent-based distributed manufacturing control: A state-of-the art survey. *Eng. Appl. Artif. Intell.* **2009**, *22*, 979–991. [CrossRef]
18. Odell, J.J.; Van Dyke Parunak, H.; Bauer, B. Representing Agent Interaction Protocols in UML. In *Agent-Oriented Software Engineering*; Ciancarini, P., Wooldridge, M.J., Eds.; Springer: Berlin, Germany, 2000; pp. 121–140.
19. Alotaibi, A.; Lohse, N.; Vu, T.M. Dynamic Agent-based Bi-objective Robustness for Tardiness and Energy in a Dynamic Flexible Job Shop. *Procedia CIRP* **2016**, *57*, 728–733. [CrossRef]
20. Marchiori, F.; Belloni, A.; Beninie, M.; Cateni, S.; Colla, V.; Ebeld, A.; Lupinelli, M.; Nastasi, G.; Neuer, M.; Pietrosanti, C.; et al. Integrated dynamic energy management for steel production. *Energy Procedia* **2017**, *105*, 2772–2777. [CrossRef]
21. Giret, A.; Trentesaux, D.; Salido, M.A.; Garcia, E.; Adam, E. A holonic multi-agent methodology to design sustainable intelligent manufacturing control systems. *J. Clean. Prod.* **2017**, *167*, 1370–1386. [CrossRef]
22. Lind, M.; Roulet-Dubonnet, O. Holonic shop-floor application for handling, feeding and transportation of workpieces. *Int. J. Prod. Res.* **2011**, *49*, 1441–1454. [CrossRef]
23. Adam, E.; Berger, T.; Sallez, Y.; Trentesaux, D. Role-based manufacturing control in a holonic multi-agent system. *Int. J. Prod. Res.* **2011**, *49*, 1455–1468. [CrossRef]
24. Barbosa, J.; Leitão, P.; Adam, E.; Trentesaux, D. Dynamic self-organization in holonic multi-agent manufacturing systems: The ADACOR evolution. *Comput. Ind.* **2015**, *66*, 99–111. [CrossRef]
25. Wang, K.; Choi, S.H. A holonic approach to flexible flow shop scheduling under stochastic processing times. *Comput. Oper. Res.* **2014**, *43*, 157–168. [CrossRef]

26. Hsieh, F.S.; Lin, J.-B. A self-adaptation scheme for workflow management in multi-agent systems. *J. Intell. Manuf.* **2016**, *27*, 131–148. [CrossRef]

27. Leitão, P. A holonic disturbance management architecture for flexible manufacturing systems. *Int. J. Prod. Res.* **2011**, *49*, 1269–1284. [CrossRef]

28. Silva, R.; Blos, M.; Junqueira, F.; Filho, D.S.; Miyagi, P. A service-oriented and holonic control architecture to the reconfiguration of dispersed manufacturing systems. In Proceedings of the 5th Doctoral Conference on Computing, Electrical and Industrial Systems (DoCEIS), Costa de Caparica, Portugal, 7–9 April 2014.

29. Jovanovic, M.; Zupan, S.; Prebil, I. Holonic control approach for the "green"-tyre manufacturing system using IEC 61499 standard. *J. Manuf. Syst.* **2016**, *40*, 119–136. [CrossRef]

30. Indriago, C.; Cardin, O.; Rakoto, N.; Castagna, P.; Chacon, E. H2CM: A holonic architecture for flexible hybrid control systems. *Comput. Ind.* **2016**, *77*, 15–28. [CrossRef]

31. Nejad, H.T.N.; Sugimura, N.; Iwamura, K. Agent-based dynamic integrated process planning and scheduling in flexible manufacturing systems. *Int. J. Prod. Res.* **2011**, *49*, 1373–1389. [CrossRef]

32. Quintanilla, F.G.; Cardin, O.; L'Anton, A.; Castagna, P. A Petri net-based methodology to increase flexibility in service-oriented holonic manufacturing systems. *Comput. Ind.* **2016**, *76*, 53–68. [CrossRef]

33. Nouri, H.E.; Driss, O.B.; Ghédira, K. Simultaneous scheduling of machines and transport robots in flexible job shop environment using hybrid metaheuristics based on clustered holonic multiagent model. *Comput. Ind. Eng.* **2016**, *102*, 488–501. [CrossRef]

34. Pascal, C.; Panescu, D. On rescheduling in holonic manufacturing systems. *Comput. Ind.* **2019**, *104*, 34–46. [CrossRef]

35. Monostori, L.; Brussel, H.V.; Westkampfer, E. Machine learning approaches to manufacturing. *CIRP Ann.* **1996**, *45*, 675–712. [CrossRef]

36. Wuest, T.; Weimer, D.; Irgens, C.; Thoben, K.D. Machine learning in manufacturing: Advantages, challenges, and applications. *Prod. Manuf. Res.* **2016**, *4*, 23–45. [CrossRef]

37. Monostori, L. AI and machine learning techniques for managing complexity, changes and uncertainties in manufacturing. *Eng. Appl. Artif. Intell.* **2003**, *16*, 277–291. [CrossRef]

38. Kadar, B.; Monostori, L. *Approaches to Increase the Performance of Agent-Based Production Systems*; Engineering of Intelligent Systems; Springer: Heidelberg, Germany, 2001; pp. 612–621.

39. Liu, Z.; Guo, Y. A hybrid approach to integrate machine learning and process mechanics for the prediction of specific cutting energy. *CIRP Ann.* **2018**, *67*, 57–60. [CrossRef]

40. Garg, A.; Lam, J.S.L.; Gao, L. Energy conservation in manufacturing operations: Modelling the milling process by a new complexity-based evolutionary approach. *J. Clean. Prod.* **2015**, *108*, 34–45. [CrossRef]

41. Bhinge, R.; Park, J.; Law, K.H.; Dornfeld, D.A.; Helu, M.; Rachuri, S. Toward a Generalized Energy Prediction Model for Machine Tools. *J. Manuf. Sci. Eng.* **2017**, *139*, 041013. [CrossRef]

42. Bang, S.H.; Ak, R.; Narayanan, A.; Lee, Y.T.; Cho, H.B. A survey on knowledge transfer for manufacturing data analytics. *Comput. Ind.* **2019**, *104*, 116–130. [CrossRef]

43. Chen, G.; Li, Y.; Liu, X. Pose-dependent tool tip dynamics prediction using transfer learning. *Int. J. Mach. Tools Manuf.* **2019**, *137*, 30–41. [CrossRef]

44. Shin, S.J.; Woo, J.Y.; Rachuri, S. Energy efficiency of milling machining: Component modeling and online optimization of cutting parameters. *J. Clean. Prod.* **2017**, *161*, 12–29. [CrossRef]

45. Lee, J.; Lapira, E.; Bagheri, B.; Kao, H. Recent advances and trends in predictive manufacturing systems in big data environment. *Manuf. Lett.* **2013**, *1*, 38–41. [CrossRef]

46. Li, L.; Liu, F.; Chen, B.; Li, C.B. Multi-objective optimization of cutting parameters in sculptured parts machining based on neural network. *J. Intell. Manuf.* **2015**, *26*, 891–898. [CrossRef]

47. Aramcharoen, A.; Mativenga, P.T. Critical factors in energy demand modelling for CNC milling and impact of toolpath strategy. *J. Clean. Prod.* **2014**, *78*, 63–74. [CrossRef]

48. Han, S.H.; Choi, Y.; Yoo, S.B.; Park, N.K. Collaborative engineering design based on an intelligent STEP database. *Concurr. Eng. Res. Appl.* **2002**, *10*, 239–249. [CrossRef]

49. Xu, X.W.; Wang, H.; Mao, J.; Newman, S.T.; Kramer, T.R.; Proctor, F.M.; Michaloski, J.L. STEP-compliant NC research: The search for intelligent CAD/CAPP/CAM/CNC integration. *Int. J. Prod. Res.* **2005**, *43*, 3703–3743. [CrossRef]

50. Vijayaraghavan, A.; Dornfeld, D. Automated energy monitoring of machine tools. *CIRP Ann.* **2010**, *59*, 21–24. [CrossRef]

51. MIT Critical Data. *Secondary Analysis of Electronic Health Records*; Springer International Publishing: Cham, Switzerland, 2016; pp. 115–141.

52. Nannapaneni, S.; Mahadevan, S.; Rachuri, S. Performance evaluation of a manufacturing process under uncertainty using Bayesian networks. *J. Clean. Prod.* **2016**, *113*, 947–959. [CrossRef]

53. Shin, S.J.; Woo, J.; Rachuri, S.; Seo, W. An energy-efficient process planning system using machine-monitoring data: A data analytics approach. *Comput. Aided Des.* **2019**, *110*, 92–109. [CrossRef]

54. Witten, I.H.; Frank, E. *Data Mining—Practical Machine Learning Tools and Techniques*; Elsevier: San Francisco CA, USA, 2005.

55. Zhong, E.; Fan, W.; Yang, Q.; Verscheure, O.; Ren, J. Cross validation framework to choose amongst models and datasets for transfer learning. In *Machine Learning and Knowledge Discovery in Databases*; Springer: Heidelberg, Germany, 2010; pp. 547–562.

56. Bellifemine, F.; Caire, G.; Greenwood, D. *Developing Multi-Agent Systems with JADE*; John Wiley & Sons, Ltd.: Chichester, UK, 2007.

57. Larek, R.; Brinksmeier, E.; Meyer, D.; Pawletta, T.; Hagendorf, O. A discrete-event simulation approach to predict power consumption in machining processes. *Prod. Eng. Res. Dev.* **2011**, *5*, 575–579. [CrossRef]

58. Shin, S.J.; Woo, J.; Kim, D.B.; Kumaraguru, S.; Rachuri, S. Developing a virtual machining model to generate MTConnect machine-monitoring data from STEP-NC. *Int. J. Prod. Res.* **2016**, *54*, 4487–4505. [CrossRef]