Journal of Intelligent Manufacturing
https://doi.org/10.1007/s10845-023-02158-5

Evaluation of process planning in manufacturing by a neural network based on an energy definition of hopfield nets

Jan Michael Spoor[1] · **Jens Weber**[2]

Received: 12 January 2023 / Accepted: 30 May 2023

Abstract
During the planning stages of new factories for the Body-In-White assembly, the processes used per production system need to be defined. Each production system uses a specific combination of processes, with each process belonging to a main process group. The combination of the processes and groups is subject to restrictions. Since the amount of possible combinations is too large to individually check for restrictions, we propose a Neural Network using an energy measurement derived from Hopfield networks. The proposed network memorizes former correct combinations and provides a recommendation score on how likely a new planned configuration is. Since processes can be paired with processes from their own group or with themselves, the Neural Network is modified to allow loops for joining vertices with themselves. This modification is achieved by adjusting the energy function of Hopfield networks to measure the activation of the combinations of clusters, meaning the edges, and not the activation of vertices during the training phase. We implemented the network for the process planning of factories of a leading European automotive manufacturer, and the results using correct, incorrect, and random process combinations indicate a strong capability of detecting anomalous process combinations.

Keywords Anomaly detection · Process planning · Hopfield neural networks · Expert systems

Introduction

The automotive industry always had a focus on automation and consequently became a leader in the field of automation, i.e., the automation of the assembly stages was driven forward with high energy (Bauernhansl et al., 2014). In detail, different levels of flexibility are necessary to cover the wide range of assembly processes and hence these processes have currently different automation levels. Most notably, the Body-In-White assembly is nowadays considered fully automated. Therefore, not only the automatization of the production but also the automatization of the planning of the production becomes of interest in manufacturing and the automotive industry in particular.

One field of research in this area which has not received sufficient attention in the past from research and industry is the computer-assisted assembly system configuration (Hagemann & Stark, 2018). In the last few years, increased efforts have been made to tackle the automation or at least the assistance of the early planning phases. This research paper tackles the detection of faults in the early process planning (PP) of production systems. If processes for a production system configuration are planned, whether automated, semi-automated, or manual, there is the need for a validation if this process combination is a valid setup of the production system (Schmidt et al., 2014). Currently, this is often a manual approach, evaluated by domain experts. To speed up this process and at least create a semi-automated solution for the validation of process combinations, a recommendation score of how likely it is that a newly planned configuration should appear might improve the overall planning procedure by highlighting the most suspicious cases and accepting common process combinations. Hence, planning experts at a leading European automotive manufacturer initiated a project

✉ Jan Michael Spoor
jan.spoor@kit.edu

Jens Weber
weberj@dhbw-loerrach.de

[1] Institut für Informationsmanagement im Ingenieurwesen (IMI), Karlsruhe Institute of Technology, Kriegsstraße 77, 76133 Karlsruhe, Baden-Württemberg, Germany

[2] Faculty of Technology, Baden-Wuerttemberg Cooperative State University Lörrach, Hangstraße 46-50, 79539 Lörrach, Baden-Württemberg, Germany

Published online: 27 June 2023

to create a method to cross-check and validate process combinations of newly planned factories and production systems for the Body-In-White assembly. The target is to create a recommendation on the likelihood of a new process combination after the proposed combination has been entered and to detect anomalous or novel combinations. This should then improve the steering and validation of the PP.

This task is complicated by frequently occurring limitations of production planning: wrong process combinations are often disregarded early in the production planning process and not documented or archived (Spoor et al., 2022). Therefore, only correct process combinations are given. If anomalous process combinations need to be detected, this must be done exclusively based on data classified as normal. Furthermore, the large variety of processes makes rule-based approaches difficult, and the database of used process combinations is limited by the amount of production systems in use. Linear or nonlinear programming using an optimization function is not possible because: first, the set-up of an useful optimization function is not feasible since besides costs and cycle times also hard-to-quantify targets must be included such as operability, safety, or processing quality; and second, the modeling of the restrictions of the optimization function would require an impracticable effort similar to rule-based approaches. Hence, a Neural Network (NN) using an energy measurement based on the Ising or Hopfield model is proposed to memorize correct sets of combinations and evaluate new combinations based on learned patterns.

First, the given use case and problem description are further presented and the state-of-the-art approaches for PP are described. Subsequently for the presented use case, an approach for PP validation using a NN and an energy measurement based on the Ising model is introduced and mathematically formulated in the methodology section. In the results section, the application of the proposed NN using the training and testing data of validated former production systems is presented. Subsequently, the resulting recommendations for the validation of process combinations are evaluated. The used methodology and results are further discussed, and afterwards an outlook on continuing research topics is given.

Use case and problem description

The given process combinations are a data set of 8, 674 formerly used and (assumed to be) correct combinations. No examples of false process combinations are given. Therefore, the approach must leverage only true negative data and be able to separate new process combinations that are not compatible with the known validated process combinations.

The total given amount of individual processes within the conducted analysis is 586. The analyzed processes are grouped into 12 process clusters containing similar types of tasks, applications, and overall uses. The clustering was conducted by the domain experts. If new processes are added in future planning phases, these would have to be assigned to the existing clusters.

Not all processes might be combinable together within the same production system, or they might need a specific additional process to allow this combination. Therefore, a set of restrictions of process combinations per production system applies. These restrictions also apply to combinations of processes with themselves and within the same group. Since processes and process groups can be combined with themselves but might also have restriction when combining them with themselves, it is necessary to model these restrictions.

For the validity check, only the combinability of a process or a process group with other processes, itself, or another process group is relevant. It is trivial to see that with 586 individual processes the spectrum of possible combinations is too large for a rule-based or even manual approach. However, even when using the 12 process clusters, the amount of combinations is too large for an individual or rule-based validation of each combination since there exists a total of 708, 576 possible process cluster combinations. This amount of combinations is corrected for impossible combinations, e.g., if process cluster connection (A, B) and (B, C) is active, the connection (A, C) must also apply, excluding some possible set-ups. Due to computational limitations, in order to reduce complexity in the solution, and since not all 586 processes occur equally often in the data set, the analysis was conducted for the 12 process clusters. This means that the given process combinations might result in duplicates after the processes were mapped to each cluster if, e.g., different process combinations result in the same process cluster combination. To illustrate the frequency of these combinations, the resulting duplicates are not removed from the data set. This results in a data set containing $< 1.2\%$ of possible combinations.

Using only the 12 process clusters, a combination can be modeled using a network graph with $V = 12$ vertices and up to $E_{max} = 78$ undirected edges, including loops since process groups can be combined with themselves. Most importantly, if two process groups in a process combination are applied, the resulting graph must have an edge between the two vertices representing the process groups and if the same process group is applied twice, the graph must have a loop in the corresponding vertex. Each data set can therefore be represented as an undirected simple graph permitting loops. If not only combinations but also sequences are of importance, the network becomes a directed graph with the edges indicating if the connected vertices precede (or succeed) each other. Sadly, sequence information was not sufficiently documented in the observed data set.

The overall goal is to assign a weight to each edge, resulting in a graph representing the likelihood of the process

combination using all 12 vertices for the process groups and all 78 edges representing the combinations. The full graph is represented as an undirected multigraph with loops, often called pseudograph. If the weights of the edges are neglected, each possible process combination, i.e., a former process combination in the given data set, is a subgraph of this overall described network. The graph structure does not change if new processes are added. Only if a new process cluster is set-up, the graph structure changes by adding another vertex with the corresponding edges. Since the process clusters are defined based on a higher abstraction level, it should be most likely possible to assign all new processes to the existing process clusters. Only in case of a distributive change of the technologies used, a new process cluster might be necessary. This is not a concern, since if this distributive technological change occurs, previous knowledge would not be usable anyway and all planning models and methods would also have to be renewed.

In conclusion, this problem description and use case can be formulated as an anomaly detection of unusual, wrong, and novel graph structures and networks using only a set of true negative network structures for training purposes. The recommendation on the likelihood of an observed combination can be evaluated using an anomaly score. A sketch of the proposed methodology is given in Fig. 1.

In addition, the resulting combination scores, anomalies, and recommendations must have a good interpretability for an application in order to show the domain experts why and how the specific results are applied. Only if the results of the model are understandable for experts with mainly engineering expertise, it can be safely applied. Within the PP phase the domain experts need to understand the reasoning to appropriately alter their PP and to understand the possible mistakes within their planning approach.

State-of-the-art methods for automated process planning in production systems

Within an overview of the whole research topic of PP, Leo Kumar (2017) describes the development of expert systems (ES) for computer-aided process planning (CAPP) as an important future research direction. ES are defined as computer programs utilizing knowledge and deduction measures to solve problems which otherwise would require human expert interaction. Therefore, we can classify the introduced problem as an area of application for ES in CAPP.

In the automated planning and validation of processes in production systems, rule-based approaches are most commonly used, e.g., currently at the mentioned automotive manufacturer. Alternatively, computational optimization and search algorithms are often applied (Hagemann & Stark, 2018). Hagemann and Stark (2020) use a combinatorial optimization algorithm to fully automated compute the best configuration of a planned production system minimizing the investment costs. Other approaches to further automate the early planning, design, and set-up of assembly lines are given by Michalos et al. (2015) and Michels et al. (2018).

Principally in PP, it is proposed to use methods of data mining for a knowledge discovery in former successful production systems, but this approach is limited by the required high standard for the used data models and an extensive initial effort (Kretschmer et al., 2017). Faster methods can be developed not by deterministically setting up rules but rather by extracting rules for the CAPP from feedback data of the production system using statistical methods of knowledge discovery. This is useful because the development of CAPP can be accelerated and created more dynamically by statistically modeling the process interdependencies instead of a fixed initial rule set (Schuh et al., 2017). Hence, our proposed approach leverages this statistical modeling to circumvent a comprehensive assessment of rule sets to cover all possible process combinations. More diverse research and development is done in the field of PP for more component- or product-centric than production system-centric applications. Methods for solving PP problems are in these cases split into exact methods and approximate methods. Exact methods use branch & bound methods and mathematical programming but are limited by the NP-hard complexity of the problems. Approximate methods include among others genetic algorithms (Liu et al., 2021). Similar methods compared to the NN proposed here in the area of CAPP are used in the PP subfield of setup planning by Ming and Mak (2000) utilizing Kohonen self-organizing neural networks to create valid restrictions of features, approach directions, procedure relationships, and tolerances. Hopfield networks are used to evaluate the operation sequence problem and the setup sequence problem by mapping them beforehand into the traveling salesman problem.

Within the field of business process monitoring, which tackles among others the prediction of running business processes using historic process data, similar use cases are researched. NN, i.e., Long Short-Term Memory (LSTM) are applied to predict the next following process based on evaluated running processes. Successful methods using LSTM are presented by Evermann et al. (2017) using embedded wording and by Tax et al. (2017). Deep neural networks considering in addition the interdependencies among the sequential event data are applied by Mehdiyev et al. (2020). Language-based models with an attention-based transformer are also possible approaches for predicting process sequences (Moon et al., 2021). The problem presented in this contribution differs in the sense that the interest is not on the prediction but only on the validation of a given process combination and on finding anomalous or novel set-ups during PP activities.

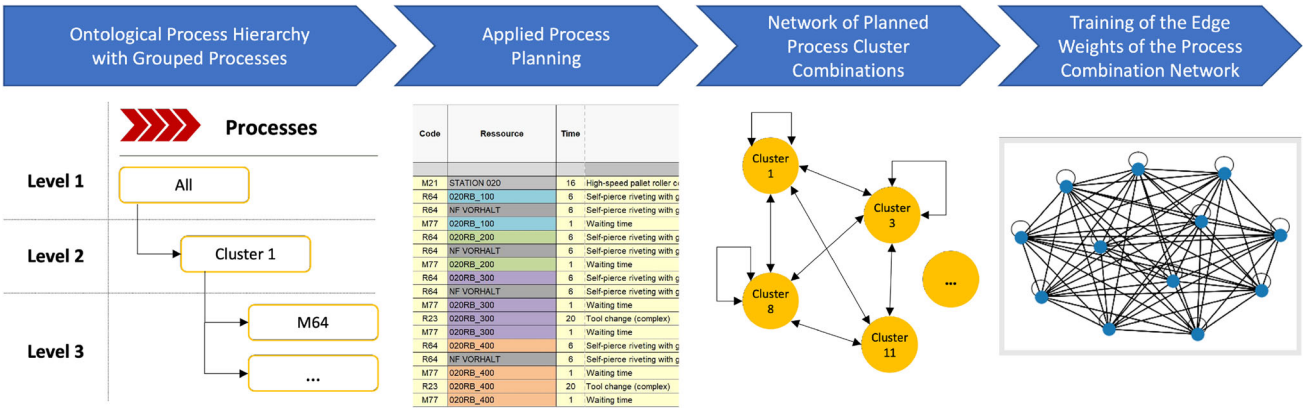


Fig. 1 The proposed methodology follows four consecutive steps: 1. similar processes are grouped in clusters, 2. process planners design a valid process combination, 3. the process combination is translated into a combination network, 4. the network's edge weights representing the likelihood of a process combination are trained

Furthermore, no sequential information is given in this particular data set but should in principle be applicable.

To the authors' best knowledge, currently no approach using NNs for PP validation exists. Also, an application of an energy measure using the Ising model for process planning or monitoring is currently not researched. However, a validation of process combinations using former applied valid processes can be seen as a form of pattern recognition. Approaches based on memorized patterns using the Ising model were successfully conducted. Nonetheless, they were within a different domain and used set-ups of the NN not considering loops (Stošić & Fittipaldi, 1997). Another closely-related method to train a network using only correct data and memorizing patterns are Autoencoders, often also called Autoassociator networks (Larochelle et al., 2009). Autoencoders proved themselves useful for anomaly detection, e.g., as applied by Chen et al. (2018) using only correct instances for the training phase. In the following section, a novel approach for this kind of anomalous pattern recognition, exemplary with an application for PP validation, is proposed. The proposed NN might also be useful in different domains and for anomaly detection of graph structures in general.

Methodology

Proposed neural network

Since only the combinations of the process clusters from an amount of V different clusters are relevant and not whether a process group itself is used, the graph representing a process combination is modeled using a symmetric $V \times V$ connection matrix M. Each element m_{ij} of the connection matrix can indicate either an active connection or no active connection between cluster i and cluster j.

$$m_{ij} = \begin{cases} +1, \; active\; edge \\ -1, \; inactive\; edge \end{cases} \tag{1}$$

Each given process combination of the data set is expressed using a Boolean connection matrix M. An entry $m_{ij} = -1$ indicates no edge between vertex i and vertex j of the graph. The set of active process clusters with the clusters numbered from 1 to V is given as P. If two vertices i and j are active, the edge m_{ij} between these vertices must be active, and if two processes of the same cluster i are in the set of process cluster combinations P, a loop in edge m_{ii} must be applied. Therefore, the set-up of the connection matrix undergoes restrictions. For reference, the algorithm for the set-up of a connection matrix is given in appendix A using algorithm 4.

The relation between the process clusters representing the combinability is expressed by an undirected multigraph with loops where the vertices are the process clusters and the weight of the edges represents the combinability between cluster i and cluster j. Since each edge is always assigned a weight, this graph has $E = \sum_{k=1}^{V} k = \frac{V(V+1)}{2}$ edges. The graph is represented by a symmetric $V \times V$ weight matrix W using individual weights w_{ij} for each combinability between vertex i and vertex j. Since all vertices are connected among each other, the proposed model design of the network is similar to a Fully Recurrent Neural Network.

$$-1 < w_{ij} < +1 \; \forall i, j \tag{2}$$

If sequences are analyzed, the network can be adjusted to a directed multigraph. In this case, the connection and weights indicate if a process i is a successor (or predecessor) of the process j. The matrices are then no longer symmetrical, but additional evaluational power is added to the model. The following overall model and procedure stays the same.

A negative weight represents a negative correlation for these vertices, i.e., these two process clusters are most likely not combined together. A positive weight represents combinations of vertices which are usually active together.

This network of weights is stimulated by an input of the connection matrix, and the weights are then updated to decrease its energy while stimulated. The energy of the stimulated network is calculated using an adaptation from the energy computation in Hopfield networks (Hopfield, 1984) or resp. in the Ising model (Brush, 1967), applying the active or inactive connection m_{ij} instead of the individual active or inactive vertices. This enables loops by circumventing the restriction that $w_{ii} = 0$. The energy resulting from the weight network when stimulated by a connection matrix using a correction value θ and the Kronecker delta function $\delta[n]$ to count the active vertices is given as follows:

$$H = -\sum_{i=0}^{V}\sum_{j=i}^{V} w_{ij}m_{ij} - \theta * \delta[m_{ij} - 1] \tag{3}$$

An equilibrium energy for the network without stimulation by an connection matrix is defined as follows:

$$H_{eq} = -\sum_{i=0}^{V}\sum_{j=i}^{V} w_{ij}^2 \tag{4}$$

This equilibrium is equivalent to the Ising model without external forces applied. The weight w_{ij} is hereby used as an estimator of the mean probability if a connection is active instead of a real applied connection m_{ij}.

The correction value is the average effect on the system energy if any combination is active. A negative correction value indicates that only a limited amount of combinations are active at the same time, while a positive correction value indicates that combinations are more likely to be active than inactive.

$$\theta = \frac{2}{V(V+1)}\sum_{i=0}^{V}\sum_{j=i}^{V} w_{ij} \tag{5}$$

For each element of the training set, the network will be stimulated and is then optimized to reduce the energy of the simulated state in the direction of the equilibrium energy (minus the correction value). The energy difference between the equilibrium and the stimulated state is given as follows:

$$\begin{aligned} G &= H - H_{eq} \\ &= -\sum_{i=0}^{V}\sum_{j=i}^{V} w_{ij}m_{ij} - w_{ij}^2 - \theta * \delta[m_{ij} - 1] \\ &= -\sum_{i=0}^{V}\sum_{j=i}^{V} w_{ij}(m_{ij} - w_{ij}) - \theta * \delta[m_{ij} - 1] \end{aligned} \tag{6}$$

To optimize the difference in energy between the two states by updating the weight, a simple gradient can be set up to update each weight individually using a training rate $\alpha \in (0, 1)$ as step sizes of adaption per iteration t:

$$w_{ij}^t = w_{ij}^{t-1} + \alpha\left(m_{ij} - w_{ij}^{t-1}\right) \tag{7}$$

This weight adjustment is similar to the weight adjustment rules for learning in Autoassociator networks using the Widrow-Hoff rule (Abdi et al., 1996). The first novelty of this approach is the adjustment of the computation of the energy function of Hopfield networks in equation (3) to allow loops by updating weights based on the edges, not vertices, as described in equation (7). This allows process combinations with itself.

The vertices are not activated one after another if the weights of the predecessor vertices exceed the activation threshold. Instead, the edges of each applied training data graph are imprinted in the weights of the whole network. This distances the model from the conventional working of neurons but nonetheless proves its usefulness in the application presented in the results section. In order to not penalize inactive combinations, only weights of i and j with active process clusters are updated fulfilling the condition for cluster i of $\sum_{j=0}^{V} m_{ij} > -V$ and analogous for cluster j.

Two adjustments can be made to refine the training phase and add commonly applied regularization techniques:

1. An overall decay $d \in [0, 1)$ of the weights towards zero if only limited training data is given for the connection.
2. A reduced training rate $r \in [0, 1]$ after each iteration of a training data amount T so that the changes of the weights are limited in later iterations.

Decay is commonly used in the training of NNs to prevent overfitting. A reduction of the training rate is often applied to enable a high learning rate at the beginning of the training phase to prevent the optimization being trapped in a local optima and then reduce the learning rate later to stabilize the solution around the found optimum. Thus, these concepts can be also applied to the proposed NN. Using these adjustments,

the change in weights per iteration step t is given as follows:

$$w_{ij}^{t} = w_{ij}^{t-1} + \alpha \left(1 - r\frac{t}{T}\right)\left(m_{ij} - w_{ij}^{t-1}\right) - d * w_{ij}^{t-1} \quad (8)$$

In addition, a correction value using the average weights does not consider the frequency of active and inactive combinations. To take infrequently active combinations into account as well, a biased correction value is useful. If combinations of clusters with themselves are more common in the data set, a simple but useful biased correction value is the average weight of all entries of the connection matrix. This approach weights combinations of different clusters twice. This approach is useful, if the diagonal entries yield significantly higher weight values than the other entries; thus, falsely indicating that active connections are more likely. Therefore, the biased correction value is smaller and penalizes a high amount of active combinations more.

$$\theta_{biased} = \frac{1}{V^2}\sum_{i=0}^{V}\sum_{j=0}^{V} w_{ij} \quad (9)$$

In general, a biased correction value can be constructed using a weighted average of the connection matrix entries. The weighted average should hereby compensate for an imbalance of the training data.

The computational complexity of the proposed method is $\mathcal{O}(TV^2)$ and hence mostly depends on the amount of selected vertices representing the clusters to check for combinability. Using this set-up and an initial weight matrix with $w_{ij}^{t=0} = 0\ \forall i, j$, a weight matrix is created which indicates, when stimulated, if the given input is memorized or unknown.

Validation and recommendation procedure

As a heuristic to separate anomalous states from correct states and if a decay $d = 0$ is used, it is proposed to use the equilibrium energy after the training as an initial test measure which should classify most correct data as true negatives. An unknown or anomalous input creates a stimulated state with an energy higher than the equilibrium energy, while a memorized common and most likely correct state will result in energy values below the equilibrium energy. The energy acts as a similarity measure between the unknown input and the memorized inputs without the necessity to previously memorize every possible pattern.

This is a second novelty of the proposed approach since it enables a probabilistic memory of multiple former correct process combinations by comparing the resulting energies, instead of an iterative adjustment of the network towards a single memorized state as commonly used in Hopfield networks. Therefore, no patterns are stored and recalled in the NN, but a limited amount of patterns are imprinted during the training phase, resulting in weighted edges of the trained network representing the combinability. These trained pattern then enable a similarity measure between the trained and suggested inputs by comparing the energy measurement using the trained NN. Thus, it is not necessary to train all possible patterns beforehand, but rather to train the NN with the known correct combinations and then evaluate the energy as a similarity measure between the trained and tested inputs.

For anomaly detection, a commonly used approach is the application of Autoencoders. One disadvantage of Autoencoders is the overall black-box characteristic of these types of models, since it is not directly visible which adjustment is necessary to create a data point which is a correct state, or, *vice versa*, which factors influenced the model to classify a data point as an outlier. In contrast, the proposed NN gives a direct evaluation as to which combination of vertices influenced its decision the most by evaluating the impact of the stimulated energy by each active connection.

If a vertex i is currently within the given process combination set P, its overall impact I_i on the measurement of the energy is given as follows:

$$I_i = \sum_{j=0}^{V} w_{ij} + \left(\sum_{j=0}^{V} w_{ij}m_{ij} - \theta * \delta[m_{ij} - 1]\right) \quad (10)$$

The impact of only an active loop is as follows:

$$I_i = w_{ii} + w_{ii}m_{ii} - \theta * \delta[m_{ii} - 1] \quad (11)$$

To reduce the stimulated energy and thus to create a more similar process combination P compared to the memorized process combinations, the vertex should be added or removed, which creates the largest impact on the energy function. This is iteratively applied until the energy of the adjusted combination decreases below a selected threshold. This process is similar to the state changes of Hopfield networks.

Algorithm 1 Training Phase

Input:
Data set X of size $T \times V \times V$
Parameter:
Training rate α
Decay d
Reduced training rate r
Output:
Trained weights matrix W of size $V \times V$
Trained correction value θ

1: Set $t = 1$
2: **for** $t \leq T$ **do**
3: Set $i = 1$
4: **for** $i \leq V$ **do**
5: Set $j = i$
6: **for** $j \leq V$ **do**
7: **if** $\sum_{j=0}^{V} m_{ij} > -V$ **or** $\sum_{i=0}^{V} m_{ij} > -V$ **then**
8: Set $w_{ij} = w_{ij} + \alpha \left(1 - r\frac{t}{T}\right)\left(m_{ij} - w_{ij}\right) - d * w_{ij}$
9: Set $w_{ji} = w_{ij}$
10: **end if**
11: Set $j = j + 1$
12: **end for**
13: Set $i = i + 1$
14: **end for**
15: Set $t = t + 1$
16: **end for**
17: Set $\theta = \frac{2}{V(V+1)} \sum_{i=0}^{V} \sum_{j=i}^{V} w_{ij}$

For the iterative removal of processes from the set of active process combinations, the equations (10) and (11) apply to select the processes with the highest impact for exclusion. In case of loops, this is conducted by first excluding all but one instance of the process from the list of processes. If adding a process results in a loop, this is analogously computed using equation (11). If new processes currently not within the set of processes are added, the procedure becomes more complicated since the connection matrix changes.

$H(P)$ is defined as the resulting energy by process combination P. If a process p is added, the resulting set becomes $P^* = P + \{p\}$. This new process combination results in a new connection matrix M by applying algorithm 4, which then enables an evaluation of the energy using equation (3). The impact becomes as follows:

$$I_i^* = H(P) - H(P^*) \tag{12}$$

Overall, equation (12) is applied for all iterations during a recommendation procedure as comparison. For each currently included vertex in the set P, the energy is computed if it becomes deactivated. If a process is currently only once in the set of processes, the impact needs to be evaluated for the addition of a loop and for complete exclusion of the process from set P. For each process p currently not in the set P, the energy is computed if the process gets included.

If the vertex i with the highest I_i^* is selected and a corresponding process added or removed from the combination list, a new combination matrix M applies, and the energy decreases. This adapts a given process combination iteratively by adding or removing processes towards a more common and memorized combination.

Proposed implementation and algorithm

Within the application in PP, the overall procedure is sketched in Fig. 2.

To train the NN model using equation (8), algorithm 1 is used, which is in Fig. 2 applied in step 1. To evaluate a given process combination using the trained NN model and the energy definition given in equation (3), algorithm 2 is used, which is in Fig. 2 applied in step 2. To create a list of recommendations using a given process combination using the trained NN model and the energy difference definition given in equation (12), algorithm 3 is used, which is in Fig. 2 applied in step 3.

In algorithm 3, three cases for each cluster per iteration are evaluated. First, if a cluster is included twice in the combination list and thus, a loop exist, the cluster is removed once such that the loop is removed (line 10 to 12). Second, if a cluster is included once, it is evaluated if removing the cluster entirely from the combination list (line 13 to 16) or adding it a second time and creating a loop decreases the energy level more (line 16 to 21). Third, if the cluster is not within the list of combinations, it is included once (line 22 to 24).

Algorithm 2 Testing Phase

Input:
Process connection matrix M of size $V \times V$
Parameter:
Trained weights matrix W of size $V \times V$
Trained correction value θ
Critical energy H_{crit}
Output:
Measured energy H
Statement if process classifies as outlier

1: Set $a = 0$
2: Set $i = 1$
3: Set $H = 0$,
4: **for** $i \leq V$ **do**
5: Set $j = i$
6: **for** $j \leq V$ **do**
7: Set $H = H - w_{ij} * m_{ij}$
8: **if** $m_{ij} = 1$ **then**
9: Set $a = a + 1$
10: **end if**
11: Set $j = j + 1$
12: **end for**
13: Set $i = i + 1$
14: **end for**
15: Set $H = H + a * \theta$
16: **if** $H > H_{crit}$ **then**
17: Outlier = TRUE
18: **end if**

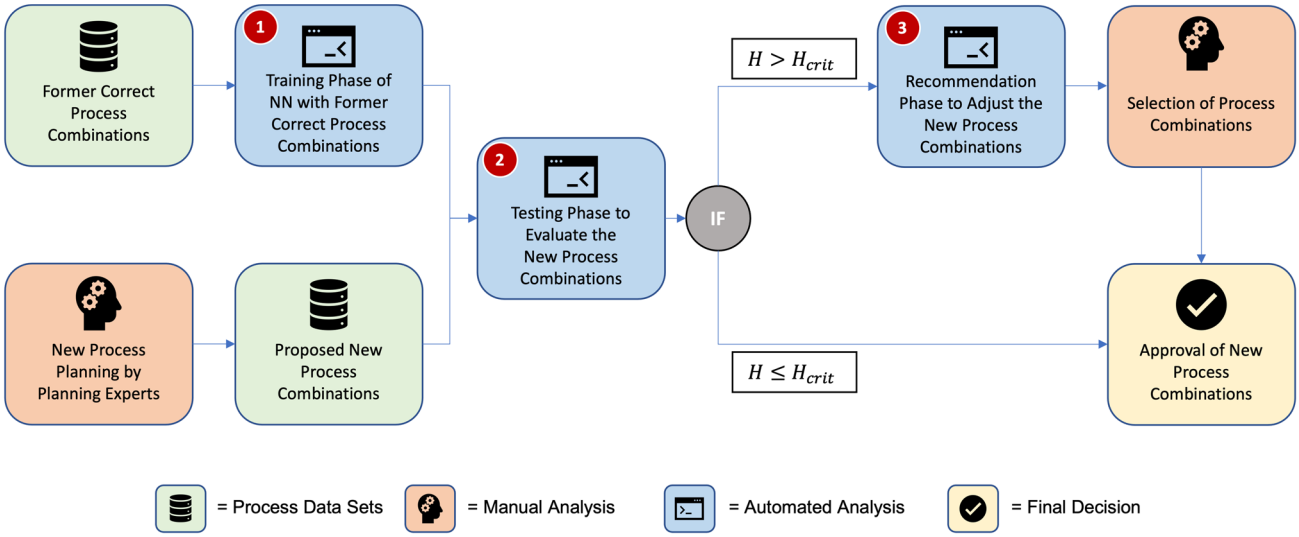


Fig. 2 Sketch of the applied concept for usage and implementation in manufacturing of the proposed NN during PP by a planning expert

Results

Data sources and description

The used data set contains, as briefly described in the problem statement, 8,674 process combinations currently implemented and applied within the production systems of two plants from different locations and both currently running at a manufacturing site of a leading European automotive manufacturer. These data include processes from all stages of the shell construction of a premium car series in the Body-In-White assembly, i.e., construction of the body frame, including assembly of the front, substructure, and rear carriage, the assembly of the side panels, and the assembly of doors and hatches. Therefore, these data cover the whole range of operations over multiple manufacturing lines using real applied processes of the Body-In-White manufacturing. Thus, the analysis is capable of representing and evaluating the performance of the proposed NN for a holistic and real application in production systems of the Body-In-White assembly as implemented today. All processes are transformed into a connection matrix by algorithm 4 prior to the analysis. An overview of the data is given in Table 1.

The applied process combinations run on a variety of unique manufacturing cells, each using from one up to twelve robots as resources per cell and multiple additional auxiliary resources, as such as, e.g., tanks and conveyor belts. For this manuscript, the analysis focuses on the applied process combinations within the manufacturing system. The robot layout as an additional restriction of a combination of processes and resources is not evaluated but could easily be added in the analysis by using vertices and edges representing resource and process combinations. This project focuses on a pure process planning use case with a known and implemented set-up of resources.

Table 1 Description of the used data set

Instance	Amount	Comment
Process combinations	8,674	
Unique processes	586	
Years	6	from 2017 to 2022
Plants	2	full Body-In-White assembly

Algorithm 3 Recommendation Phase

Input:
Process combination set P
Parameter:
Trained weights matrix W of size $V \times V$
Trained correction value θ
Critical energy H_{crit}
Output:
List of recommendations P^* of size $n+1$
List of energy improvements H^* of size $n+1$

Set $n = 0$, $removed =$ FALSE
Compute M from set P using algorithm 4
Compute H from the connection matrix M using algorithm 2
Set $P_0^* = P$
Set $H_0^* = H$
while $H_n^* > H_{crit}$ **do**
  Set $I_{max} = 0$
  Set $i = 1$
  for $i \leq V$ **do**
    if $\|P_n^* \cap \{i\}\| > 1$ **then**
      Set $P^* = P_n^* - \{i\}$
    end if
    if $\|P_n^* \cap \{i\}\| = 1$ **then**
      if $removed =$ FALSE **then**
        Set $P^* = P_n^* - \{i\}$
        Set $i = i - 1$, $removed =$ TRUE
      else
        Set $P^* = P_n^* + \{i\}$
        Set $removed =$ FALSE
      end if
    end if
    if $\|P_n^* \cap \{i\}\| = 0$ **then**
      Set $P^* = P_n^* + \{i\}$
    end if
    Compute M' from set P^* using algorithm 4
    Compute H' from the connection matrix M' using algorithm 2
    Compute $I_i = H - H'$
    if $I_i > I_{max}$ **then**
      Set $I_{max} = I_i$
      Set $P_{opt} = P^*$
      Set $H_{opt} = H'$
    end if
    Set $i = i + 1$
  end for
  Set $n = n + 1$
  Set $P_n^* = P_{opt}$
  Set $H_n^* = H_{opt}$
end while

As discussed previously, only combinations that are assumed to be correct are available, which requires a training with only true negative network structures. Therefore, we created 30 false process combinations supported by industry experts and created a set of random combinations for a validation of the anomaly detection in the testing phase.

Hyperparameter optimization

To find a useful parametrization, a hyperparameter optimization of the training rate α, decay d, and reduced training rate r is applied. For this purpose, a grid search is used. Each set of parameters is evaluated with 30 Monte Carlo cross-validations and a training set size of 90% is applied. The NN is trained using algorithm 1. A more detailed discussion and interpretation of the training procedure is given in the following section.

For the testing of the NN, algorithm 2 is applied using the biased correction value. The results of the testing phase are evaluated by the rate of misestimating correct process combinations as false positives, using a given constant detection rate of 90.0% of false process combinations as true positives (TPR). The lower the false positive rate (FPR), the better the parametrization. The results of the grid search are given in Table 2. A detailed discussion and interpretation of the training procedure is given in the following sections.

The grid search suggests that the NN is comparably stable regarding its parametrization. For small changes in the parameters, no significant effect is measurable. Only in cases of high values for training rate or decay, an effect is significant and the variance depending on the training data increases. The grid search has most likely not found the global minima, but the possible improvement in the global optimum of the FPR is assumed to be rather low in case of the given data set. Thus, the parameters can be freely selected in the range of $0.01 \leq \alpha \leq 0.1$, $0 \leq d \leq 0.04$, and $0 \leq r \leq 0.5$. Changes of the parameterization within this limits will change the resulting equilibrium and stimulated energy levels but will still enable a distinct threshold for the critical energy level to separate correct and incorrect process combinations. Since decay and reduced training rate do not have a significant influence on the performance for training rates $\alpha \leq 0.1$, they can be set to zero to reduce complexity of the proposed NN and to apply equation (7) during the training phase. It should be noted, that other data sets and use cases might benefit from these parameters.

Table 2 Effect of parametrization evaluated by the FPR using a given 90.0% TPR

Training rate	Decay	Reduced training rate $r = 0$	$r = 0.1$	$r = 0.25$	$r = 0.5$
$\alpha = 0.01$	$d = 0$	$(1.7 \pm 0.4)\%$	$(1.6 \pm 0.5)\%$	$(1.4 \pm 0.4)\%$	$(1.5 \pm 0.4)\%$
	$d = 0.02$	$(1.6 \pm 0.5)\%$	$(1.5 \pm 0.5)\%$	$(1.6 \pm 0.4)\%$	$(1.4 \pm 0.4)\%$
	$d = 0.04$	$(1.8 \pm 0.6)\%$	$(1.6 \pm 0.7)\%$	$(1.5 \pm 0.5)\%$	$(1.3 \pm 0.5)\%$
	$d = 0.1$	$(1.7 \pm 0.6)\%$	$(1.6 \pm 0.5)\%$	$(1.8 \pm 0.8)\%$	$(1.7 \pm 0.8)\%$
$\alpha = 0.05$	$d = 0$	$(1.3 \pm 0.5)\%$	$(1.4 \pm 0.5)\%$	$(1.5 \pm 0.5)\%$	$(1.4 \pm 0.4)\%$
	$d = 0.02$	$(1.5 \pm 0.5)\%$	$(1.7 \pm 0.6)\%$	$(1.8 \pm 0.6)\%$	$(1.4 \pm 0.5)\%$
	$d = 0.04$	$(1.6 \pm 0.6)\%$	$(1.5 \pm 0.5)\%$	$(1.4 \pm 0.6)\%$	$(1.6 \pm 0.6)\%$
	$d = 0.1$	$(1.9 \pm 1.0)\%$	$(1.8 \pm 1.1)\%$	$(1.8 \pm 1.0)\%$	$(1.5 \pm 0.7)\%$
$\alpha = 0.1$	$d = 0$	$(1.4 \pm 0.5)\%$	$(1.5 \pm 0.6)\%$	$(1.4 \pm 0.5)\%$	$(1.5 \pm 0.6)\%$
	$d = 0.02$	$(1.8 \pm 0.8)\%$	$(1.7 \pm 0.7)\%$	$(1.4 \pm 0.6)\%$	$(1.6 \pm 0.6)\%$
	$d = 0.04$	$(1.8 \pm 0.7)\%$	$(1.6 \pm 0.8)\%$	$(1.7 \pm 0.8)\%$	$(1.4 \pm 0.5)\%$
	$d = 0.1$	$(2.1 \pm 1.1)\%$	$(2.0 \pm 1.1)\%$	$(1.9 \pm 1.1)\%$	$(1.6 \pm 0.7)\%$
$\alpha = 0.25$	$d = 0$	$(2.7 \pm 1.5)\%$	$(1.8 \pm 1.3)\%$	$(1.9 \pm 0.8)\%$	$(1.9 \pm 1.0)\%$
	$d = 0.02$	$(2.0 \pm 1.0)\%$	$(2.1 \pm 1.2)\%$	$(2.1 \pm 1.1)\%$	$(1.7 \pm 0.9)\%$
	$d = 0.04$	$(2.4 \pm 1.3)\%$	$(2.0 \pm 1.0)\%$	$(1.9 \pm 0.9)\%$	$(1.6 \pm 0.7)\%$
	$d = 0.1$	$(2.9 \pm 1.5)\%$	$(2.6 \pm 1.7)\%$	$(2.6 \pm 1.9)\%$	$(2.1 \pm 0.9)\%$

Using as parameters values $\alpha = 0.05, d = 0$, and $r = 0.0$, the effect of the training size T is evaluated. The NN is trained and tested using different training and testing sizes and then evaluated using as prior the FPR for a given 90.0% TPR. The results are given in Table 3. The biased correction value is applied and 30 Monte Carlo cross-evaluations are conducted.

Even with around 5,000 training data, the networks performance does not significantly change and the training is stable. Thus, this amount of data is sufficient to conduct a useful anomaly detection task and all tested training sizes show comparable results. Overall, the performance of the model for the given data set is independent of the selected training size and thus, a training size of 90% can be used for the application of the model. Nonlinear relationships between training size and parameters have not been evaluated separately, but since the effects of parametrization on the FPR are rather small, no significant effect is assumed.

The last parametrization is the selection of the biased or regular correction value θ. In the given use case many loop connections exit since process clusters are very often combined with themselves; thus, the biased threshold is useful in training and evaluation of the network. Using the found parameters and a training size of 90%, the FPR is $(1.9 \pm 0.6)\%$ for the regular correction value using a 90.0% TPR. If this value is compared to the results in Table 3, it is concluded that the biased threshold is useful in driving the performance of the evaluation.

Overall, the parametrization is rather stable and enables an efficient set-up of anomaly detection tasks without an extensive hyperparameter optimization. In different use cases and datasets, a hyperparameter optimization might be more important. While the parameters might be further optimizable by a pattern search or similar methods, the results indicate a high stability and the results are more driven by the training data sets, since most false positive rates of the used parameters are within one standard deviation.

Procedure of training phase

For the training phase the data set is shuffled and split into 90/10 training and testing data. Since the data set is shuffled, the results vary slightly between each run, but in multiple conducted tests using a Monte Carlo cross-validation the overall results and performance of the used models stayed relatively unchanged as demonstrated in the hyperparameter optimization. The remaining $T = 7,807$ process combinations are then used to iteratively change the weights as per equation (8). The initial weights are set as zero for all edges. The parameters are selected based on the conducted hyperparameter optimization. Since decay and reduced learning rate have no significant impact for values of $0.01 \leq \alpha \leq 0.1$, they are set to zero such that the iterative weight change simplifies to equation (7). The applied parameters for the training are summarized in Table 4.

To visualize the training progress, the energy in the equilibrium and the energy in the stimulated state per iteration t are compared using equation (3) and (4) with the correction value continuously updated after each iteration. The stimulated state can be averaged over the last 100 iterations to create a magnitude of the variance and mean values during training. The visualization of the training phase is given in Fig. 3.

Table 3 Effect of different training sizes evaluated by the FPR using a given 90.0% TPR

	Training size to data set size ratio				
	50 %	60 %	70 %	80 %	90 %
T	4,337	5,205	6,072	6,940	7,807
FPR	$(1.5 \pm 0.5)\%$	$(1.6 \pm 0.4)\%$	$(1.6 \pm 0.5)\%$	$(1.5 \pm 0.5)\%$	$(1.3 \pm 0.5)\%$

Table 4 Applied parameters for the training phase

Parameter	Symbol	Value
Training size	T	7,807
Training rate	α	0.05
Decay	d	0.0
Reduced training rate	r	0.0

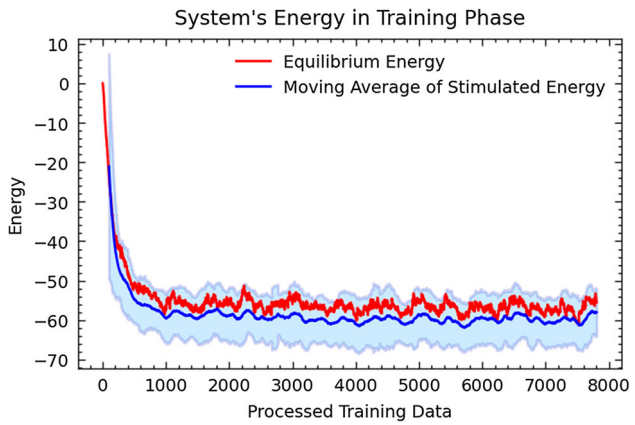


Fig. 3 Equilibrium and stimulated state's energy during training phase. The moving average of the stimulated states is calculated per 100 training data. The blue shaded region is the 3-σ region of the moving average

It is visible that after around 1,000 processed training data the energy stabilizes and does not change any longer in a significant manner. Also, the moving average of the stimulated states stabilizes and remains constant with a stable 3-σ area. Therefore, using the applied parameters and despite the limited amount of training data, it is possible to conduct a learning phase which stabilizes. Also, the learning progress is robust over changes in the parameters, indicating an overall stable model. Smaller training sizes still enable a stabilization of the energy, as shown in the hyperparameter optimization for different training sizes. Therefore, other splits then 90/10 can be used but result in limited changes.

The upper end of the 3-σ area overlaps with the equilibrium energy. If the suggested heuristic is applied using the equilibrium energy as criterion for the detection of anomalous process combinations, the area between the upper 3-σ interval and the equilibrium gives an indication of the expected amount of false positives during the testing phase. Some uncommon process combinations will be detected as anomalous, but the main amount of data, containing most likely common or typical combinations, will pass as true negatives.

Results of testing phase

First, the biased correction value is computed as $\theta_{biased} = -0.79$. The biased correction value is used since combinations of clusters with themself are disproportionately present in the data set. Also, the biased correction value increases the predictive performance of the model compared to an unbiased value. The negative value of the correction value indicates that an active connection is overall unlikely and most connections will be inactive. This results in a penalization of process combinations between multiple different process clusters which the biased correction value might not correct circumferentially. This should not be interpreted as an error but as part of the results since it only suggests that production system set-ups with multiple different process groups are unlikely.

For the testing phase, the 10%, in total 867, remaining correct process combinations are used as validated test data to check the rate of false positives. In addition, an equal amount of random process combinations is generated. A random amount of 2 to 5 process clusters are activated. If a cluster is activated twice, it counts as a connection to itself. Furthermore, 30 validated false process combinations are used to test the model's capabilities of detecting true negative combinations. For all testing data, the energy is calculated and visualized in Fig. 4 using box plots with whiskers.

The proposed heuristic using the equilibrium energy suggests a threshold of $H_{crit} = -55.5$, classifying all combinations above as anomalous. As indicated in Fig. 3, the amount of false positives in the tested correct process combinations is very limited, with only 2.9% using the equilibrium energy as threshold. Certain outliers exist which are (assumed to be) correct combinations but are classified as anomalous. However, compared to the random and false process combinations, the number is highly limited. Also, the mean energy level of $\overline{H}_{true} = -59.0 \pm 1.9$ is significantly lower than the threshold. While the random process combinations are often under the threshold, they are visibly above the correct validated process combinations with a mean energy level of $\overline{H}_{rand} = -53.6 \pm 3.0$, indicating that they are only with a low significance above the energy level of the correct processes. This might result from generating random data, which might

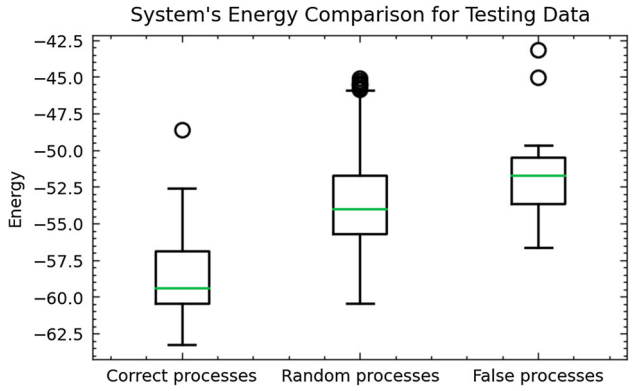


Fig. 4 Median energy levels per state of validated correct process combinations, random combinations, and validated incorrect combinations. The boxes show the lower to upper quartile. The median is marked with a line. The whiskers indicate the 1.5 times interquartile range above/below the quartile. Data outside of the whiskers are marked as dots

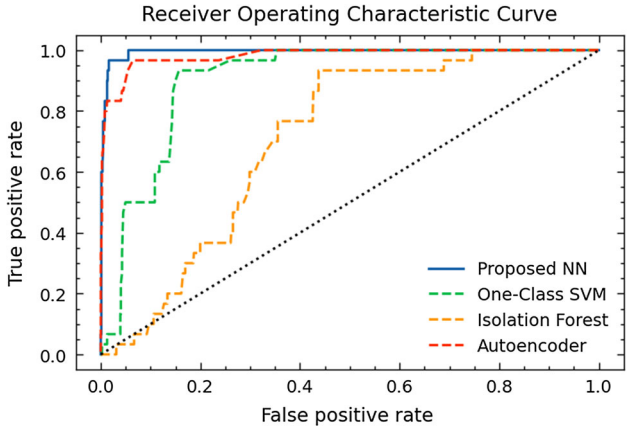


Fig. 5 Receiver operating characteristic for the trained network using verified false states. An Autoencoder, One-Class SVM, and Isolation Forest are used for benchmarking

create simply correct or at least inconspicuous samples. Also, the random process combinations benefit from the limitation of a maximum of 5 active clusters. Hence, the effect of many unlikely cluster combinations as suggested by the negative correction value is circumvented.

In conclusion, by using the equilibrium energy as a threshold, 96.7% of validated false processes are classified correctly as true positives. If more false positives are accepted, since this ratio is quite low when using the equilibrium energy, this ratio increases. Also, the energy level between the false and correct combinations differs significantly with an energy level of $\overline{H}_{false} = -51.8 \pm 2.8$. The combinations with a very high energy are process combinations with many different active clusters. The lower whiskers of the box plot barely reach the upper quartile of the correct data. This indicates that the true positives and true negatives can be separated quite efficiently.

Performance evaluation and benchmarking

The capabilities for outlier detection of the proposed NN are further analyzed using a receiver operating characteristic (ROC) curve in Fig. 5.

As shown, the correct and false data can be classified very efficiently, and using a stricter measure then the equilibrium energy can further drive the performance of the analysis. It is possible to detect 90.0% of false process combinations as true positives with only misestimating 1.3% of correct process combinations as false positives using a threshold of $H_{crit} = -54.7$. Using this energy level, most of the random data are also classified as anomalous which should be correct due to their random nature. Therefore, it is concluded that the suggested model can be applied to the defined task and is able to reliably detect false process combinations while passing correct combinations.

As the main competitor in these scenarios, an Autoencoder is selected for benchmarking. In addition, the data set is benchmarked using a One-Class Support Vector Machine (SVM) as proposed by Schölkopf et al. (2001) and an Isolation Forest as proposed by Liu et al. (2008). The methods are chosen as benchmarks since they are able to detect anomalies in data sets without a prior training data set with labeled classifications. For this purpose, the connection matrix M must be converted into a Boolean feature vector containing all 78 possible process combinations.

The applied methods were trained using the same training data set as the NN and then evaluated by their anomaly score comparing the test data set and the validated false process combinations. The Autoencoder was set up using the implementation by Zhao et al. (2019). After testing different parameters, a four layer network with 64/32/32/64 neurons with an ReLU activation function for the hidden states, a sigmoid activation function for the output layer, as used per default by Zhao et al. (2019), and a contamination rate of 1% is applied. The One-Class SVM was specified to use an RBF kernel with a $\gamma = \frac{1}{78\sigma}$. The Isolation Forest was set up with a contamination of zero and also a contamination calculated as originally proposed. Since the calculated contamination suggested a slightly better performance, it was used in the final evaluation. This result might derive from considering that within the training data set some uncommon (or even falsely classified as correct) combinations might be still included. Therefore, a calculation of the contamination is beneficial. The One-Class SVM and Isolation Forest are applied using the implementation by Pedregosa et al. (2011).

The results are given in Fig. 5. It is assumed that a One-Class SVM performs well using only true negative examples during training but might in this case be limited by the binary nature of the connection m_{ij}. The performance of the One-Class SVM is therefore worse for all possible false

positive rates than our proposed model. The Isolation Forest is also limited by the binary connection and performs worse then the One-Class SVM and the resulting ROC is also significantly exceeded by our proposed NN. However, the Autoencoder is capable of a similarly good detection of outliers, but overall the proposed NN results in a slightly better ROC. The Autoencoder even performs on par with our proposed method in areas of lower than 82% true positive rates. Using more validated false process combinations might improve the significance of the comparison and result in a better recommendation, but these are limited by the nature of the problem.

Nevertheless, the Autoencoder in addition lacks the easily interpretable weights of our proposed NN. Since the final weights are defined in the range of -1 to 1, they can be easily shown and discussed with domain experts and create an explainability which the Autoencoder cannot provide. A value close to 1 suggests a high combinability and combinations which are present in most process set-ups, while a value close to -1 suggests combinations which are very unlikely to occur. By looking up the values of the weight matrix, it can easily be explained why a result received a certain energy level. Using equation (10) and (11) for each currently active process, its impact on the energy level and therefore its impact on the classification procedure can be evaluated. This easily interpretable model is further highlighted by the recommendation process, which suggests unlikely processes combinations or missing processes in the current process combination.

In addition, the proposed NN yields a more beneficial computational complexity than an Autoencoder. Therefore, we assume that our proposed model is highly beneficial in applications where results are aligned with experts outside of computer science and reasoning must be applied on the results, requiring less of a black-box approach. In conclusion, the benchmarking shows that our proposed approach is very beneficial for the presented use case which cannot be simultaneously well-evaluated and interpreted in an equivalent manner using common state-of-the-art methods of anomaly detection.

Applied recommendation phase

If the recommendation system is applied, processes get activated or deactivated iteratively to improve the energy rating. Since in the use case presented here the correction value is negative, the network will in most cases prefer less processes and therefore mostly delete processes from the proposed process combinations entered in the system. On the other hand, since process combinations within the same cluster are very common, processes of the same process cluster might be added. The progress of recommendation is illustrated by the energy change per iterative step in Fig. 6.

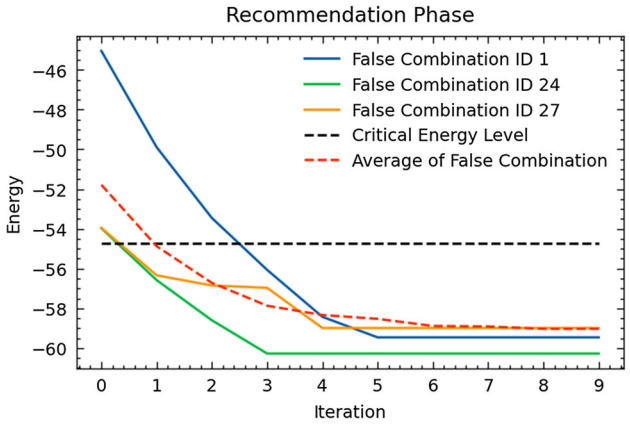


Fig. 6 Energy level after multiple iterations of recommendation with single activations or deactivations of processes per iterative step

Table 5 Recommended process cluster adjustments per iterative step

Iteration	Process Combinations		
	ID 1	ID 24	ID 27
0	6, 7, 10, 11, 12*	1, 3, 12*	6, 8, 10*
1	6, 10, 11, 12*	1, 3, 12, 12	8, 10
2	10, 11, 12*	1, 1, 3, 12, 12	10
3	10, 11, 12, 12	1, 1, 12, 12	1, 10
4	11, 12, 12	1, 1, 12, 12	1, 1, 10
5	12, 12	1, 1, 12, 12	1, 1, 10
...	12, 12	1, 1, 12, 12	1, 1, 10

*The energy of the combination is above the critical value

It is visible that after a few adjustments to the process combination lists the values are below the threshold $H_{crit} = -54.7$ and would classify as correct process combinations. The single recommendation steps are listed in Table 5.

For the presented examples, the recommended process combinations iterate towards the most likely combinations, which differ between cases. Depending on the configurations or starting values, the recommendations iterate towards different energy levels. The resulting recommendations are therefore local minima, but are not necessarily global minima. Also, the robustness of the recommendations is limited since changes in the training data sometimes result in a swapping of the best and second best proposed adjustment. Nevertheless, the recommendations all result in more likely combinations. Thus, in practice, the planning expert conducting the recommendation phase should be shown a list of all feasible recommended adjustments for manual selection. In the analyzed use case, all false processes combinations changed to assumed correct combinations after a maximum of 3 steps. In larger networks these recommendation phases can be more nuanced, longer, or complex.

Evaluation of an exemplary process combination

For a better understanding of the methodology and the application of the proposed NN, an exemplary process, which is a real currently running process at said automotive manufacturer, is analyzed in this section.

As an exemplary application, a more unique process combination is selected. This process combination runs on a manufacturing line of a plant in Germany. The corresponding manufacturing line of the plant analyzed here is located in the assembly of the side panels and named FS71. The first manufacturing cell of this line through which a car body passes runs the selected exemplary process combination.

The manufacturing line FS71, which includes the analyzed cell, is of particular interest since it includes among the most varied applications and special welding and gluing processes, as well as multiple tool exchanges. Thus, the said European automotive company uses this manufacturing line as a benchmark for testing new ideas and concepts of the internal digital factory research. The manufacturing cell's position in the factory layout is displayed by a red marking in Fig. 7.

The manufacturing cell can be displayed in more detail using the digital plant layout. This manufacturing cell includes as resources four robots (named 020RB 100 up to 020RB 400) and also a conveyor belt (named Station 020). The digital layout of the manufacturing cell is given in Fig. 8.

The process combination analyzed in this section starts with the delivery of the car body via the conveyor belt and then conducts a self-pierce riveting process including a gluing application by all robots. During the ongoing riveting process of the robots 020RB 100 and 020RB 200, the other robots 020RB 300 and 020RB 400 finish their riveting task earlier and come to a prior halt to start an exchange of their tools in a special tool change process. This tool change is marked by the planning experts as a complex and non-standard change. This tool change is necessary for a following process. After this tool change, 020RB 300 and 020RB 400 undergo a waiting time until 020RB 100 and 020RB 200 are also finished with the conducted self-pierce riveting process. After another waiting time, the process ends. The process chart is given in Fig. 9.

This process combination includes the process codes M21, R64, M77, and R23. The process code M21 is applied once, which is the delivery of the car body by the conveyor belt. The process code M21 is assigned to cluster 11. Cluster 11 includes all process without robots which handle ramping and clamping procedures. The process code R64 describes the self-pierce riveting process and is included eight times. The code R64 belongs to cluster 3 which includes all technical robot main processes, mainly welding applications. The process M77 is included six times and is a waiting time. This is covered by cluster 8, including all preprocessing tasks which do not include actions by robots or handling of the components. In conclusion, the code R23 of the tool exchange is included twice and belongs to cluster 2 which are processes running on robots but without a processing of the component.

To analyze this process combination, first the symmetric connection matrix must be set up as given by algorithm 4. Also, this process can be visualized by a network graph. The edges between cluster 2, 3, 8, and 11 are all applied; thus, the entries (2, 3), (2, 8), (2, 11), (3, 8), (3, 11), and (8, 11) and their symmetric counterparts of the connection matrix are given as 1. Also, the loops of cluster 2, 3, and 8 are active since these processes are applied more than once; thus, the second, third, and eighth diagonal entries of the connection matrix are 1. Only cluster 11 does not have a loop; therefore, the connection matrix has the value -1 for the eleventh diagonal entry. All other edges are inactive, and the connection matrix has the value -1 in all other entries.

If this process combination would be applied for a training phase, all weights where this connection matrix has entries of 1 would increase in this training step and all weights where the connection matrix has values of -1 would decrease, as defined by the learning rule in equation (8).

If this process is applied for an outlier test, the energy is measured as defined by equation (3) using the trained weight matrix and biased correction value. Since this process, respectively the line FS71, is described by the planning experts as a special case of particular interest due its high complexity, it is expected that the resulting energy might be higher then the average case of a process combination but should still be classified as a correct combination.

In conclusion, if this process combination is applied on the trained weight matrix from prior sections, the resulting energy is given by $H_{example} = -58.0$. This is indeed slightly higher than the average energy of correct combinations given in Fig. 4 by $\overline{H}_{true} = -59.0$ but still within the standard deviation of the true process combinations. Also, it is lower than the proposed $H_{crit} = -54.7$ by the analysis of the ROC curve in the section "Performance evaluation and benchmarking" and significantly lower than the energy levels of the random and false process combinations as seen in Fig. 4. Thus, this process would classify as a correct process, and its more interesting aspects indeed result in a slightly higher energy than the average correct process. Therefore, it can be concluded that the proposed NN gives a good estimation of the complexity of the applied process combinations and results in a meaningful outlier score using the energy evaluation.

Discussion

An important assumption of the approach is that past configurations can be used to validate entirely new combinations. Since we assume that this assumption holds in the presented

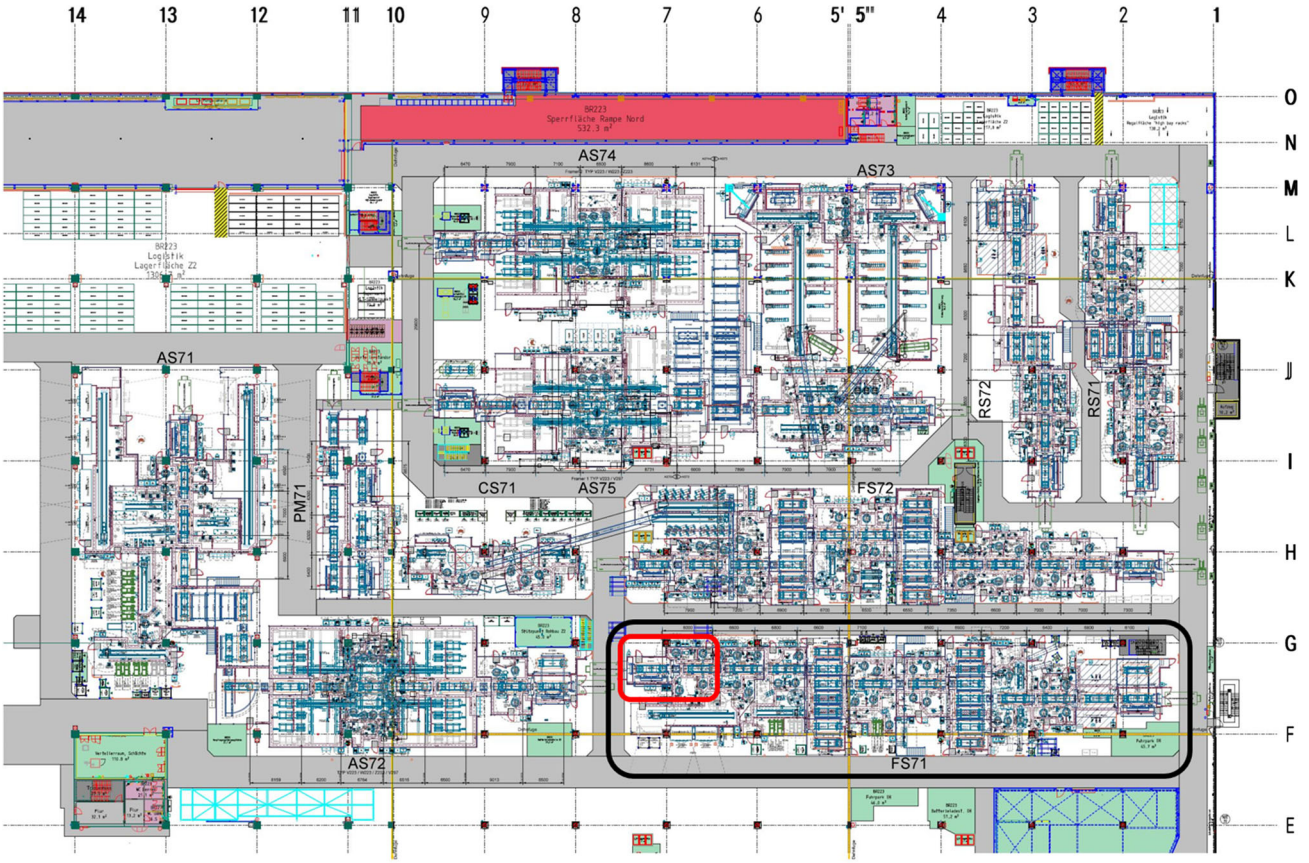


Fig. 7 Plant layout of the assembly of the side panels of one location. All here visible working stations and cells are applied in the analysis. The manufacturing line FS71 is located in the lower right of the map. The exemplary process combination is located at the manufacturing cell marked in red

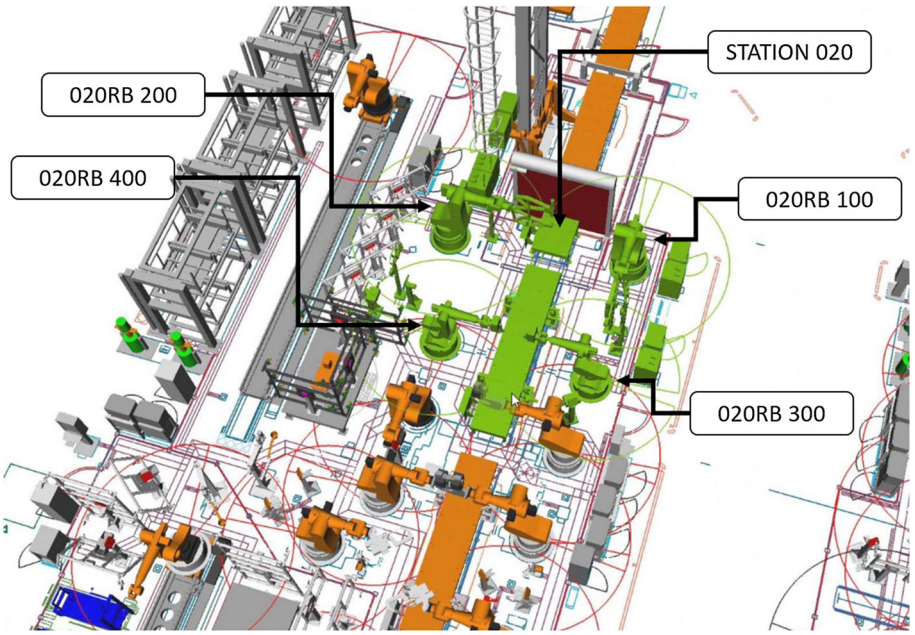
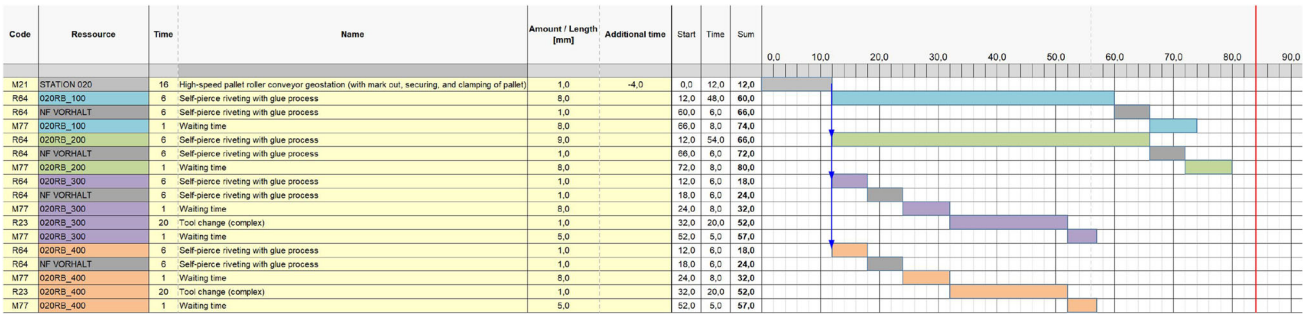


Fig. 8 Layout of the observed manufacturing cell, which is the first cell of the manufacturing line FS71, running the exemplary process. The resources of the corresponding manufacturing cell are marked in green. The robots and the conveyor belt are named with arrows



Code	Ressource	Time	Name	Amount / Length [mm]	Additional time	Start	Time	Sum
M21	STATION 020	16	High-speed pallet roller conveyor geostation (with mark out, securing, and clamping of pallet)	1,0	-4,0	0,0	12,0	12,0
R64	020RB_100	6	Self-pierce riveting with glue process	8,0		12,0	48,0	60,0
R64	NF VORHALT	6	Self-pierce riveting with glue process	1,0		60,0	6,0	66,0
M77	020RB_100	1	Waiting time	8,0		66,0	8,0	74,0
R64	020RB_200	6	Self-pierce riveting with glue process	9,0		12,0	54,0	66,0
R64	NF VORHALT	6	Self-pierce riveting with glue process	1,0		66,0	6,0	72,0
M77	020RB_200	1	Waiting time	8,0		72,0	8,0	80,0
R64	020RB_300	6	Self-pierce riveting with glue process	1,0		12,0	6,0	18,0
R64	NF VORHALT	6	Self-pierce riveting with glue process	1,0		18,0	6,0	24,0
M77	020RB_300	1	Waiting time	8,0		24,0	8,0	32,0
R23	020RB_300	20	Tool change (complex)	1,0		32,0	20,0	52,0
M77	020RB_300	1	Waiting time	5,0		52,0	5,0	57,0
R64	020RB_400	6	Self-pierce riveting with glue process	1,0		12,0	6,0	18,0
R64	NF VORHALT	6	Self-pierce riveting with glue process	1,0		18,0	6,0	24,0
M77	020RB_400	1	Waiting time	8,0		24,0	8,0	32,0
R23	020RB_400	20	Tool change (complex)	1,0		32,0	20,0	52,0
M77	020RB_400	1	Waiting time	5,0		52,0	5,0	57,0

Fig. 9 Process chart of the analyzed exemplary process. 020RB 100 is marked in blue, 020RB 200 is marked in green, 020RB 300 is marked in violett, and 020RB 400 is marked in orange

use case, this might not always be the case. In addition, a valid combination which was simply never performed before and also does not follow past patterns of combinations will most likely be classified incorrectly, as seen in the test data set as the false positive cases, since no domain knowledge is applied to model the real restrictions. The model is also limited by the availability of only process combinations and no complete ontology or process sequences. Within manufacturing, the use of an ontology, i.e., the product, process, and resource (PPR) model, can improve the planning and simulation of production systems (Agyapong-Kodua et al., 2014). Adding information concerning the process sequences or the process combinations with resources and products will most likely improve the significance of the model. Sequences can be modeled via directed edges between the processes, and resource and product combinations can be added using vertices. While the overall results indicate the usefulness of the model, two additional aspects in the mathematical set-up need to be discussed further.

First, the calculation of the correction value should in theory balance the energy change by activating any edge and create a comparability between combinations with different amounts of edges. The analysis shows that the high energy combinations are cases with many active edges. Therefore, the model penalizes large amounts of active edges. This is partly an intended phenomenon as combinations with a large amount of different clusters are only rare or unrealistic occurrences, thus these cases should be reasonably checked twice. Otherwise, the correction value might not create a real comparability between cases with different amounts of active edges since the correction value compensates using the mean of the weights, which does not comprise the full spectrum of weights. The introduction of a biased correction value attempts to mitigate this problem. Nevertheless, a direct comparison of process combinations with different amounts of active clusters might be limited, and only comparisons between process combinations using the same cluster amount might create results that are easy to interpret. A further analysis might be necessary to specify the best approach to compensate an activation of different amounts of edges when calculating the correction value.

Second, using the weights as the mean probability that a cluster gets activated and then using this mean as an input for an equilibrium energy state might simplify the analysis too much. A more nuanced equilibrium energy state could be based on the mean energy of the Ising model. This would result in a more complex set-up of the energy function but might further improve the results or enable a faster and even more stable training process. On the other hand, the resulting gradient using the difference between the activation and the weight is easily interpretable: a high positive weight is most beneficial if a positive value, an active edge, of the connection matrix is given and decreases the energy the most, while it penalizes a inactive edge the most. Despite being quite simple, this logic directly optimizes the energy for the most likely connection matrix, and based on the used training data set, the more common a connection matrix is, the lower the energy becomes.

Overall, the model is easily interpretable, does not require an intensive set-up of parameters, and results in a high sensitivity without an extensive amount of necessary training data. In addition, the model does not require domain and expert knowledge as input. Therefore, we conclude that the model is a viable method for memorizing correct combinations and giving estimations on how close new configurations, resp. combinations, are to an already memorized training data set. This also enables a useful method of anomaly detection for graph structures.

To compare our proposed model with, e.g., human learning, it can be interpreted that a new task, image, etc. that is similar to an already known task becomes less stressful. On the other hand, completely new impressions result in more stress in the form of more attention or tension required to process the task. In addition, the applied learning rate with more familiar combinations resulting in less adjustments of the network is also comparable to the human learning experience, as discussed within the context of Autoassociator networks by Sirois (2004).

Conclusions

The proposed NN is able to successfully validate process combinations with a high sensitivity. The main novelty of the proposed NN model is the usage of a connection matrix instead of a consecutive activation of each vertex. Also, the usage of the Ising model energy function is adjusted to satisfy the loop requirement, and an adjusted equilibrium energy function, which shows its usefulness in the evaluation, is proposed. In future research, we plan a further optimization of the model and applied parameters and a more in-depth benchmarking for multiple different data sets to further prove its effectiveness.

This automated evaluation of process combinations is a novel approach in CAPP, as within this domain mostly rule-based and model-based optimization approaches are used. Therefore, this contribution tackles an under-researched field, opens new possible applications of CAPP, and hence can advance the overall field of automated planning in the manufacturing industry, in particular the automotive industry. Furthermore, the approach can potentially be extended to validate not only combinations of processes but also process, product, and resource combinations to add additional ontologies and cover the planning procedures along the entire PPR model. Also, applications within business processes and in other domains are possible, e.g., to detect anomalous process transactions that violate policies and normal procedure. In addition, the proposed model applied for anomaly detection comparing different graph structures or networks might be applicable in a wide variety of domains outside the field of PP. A generalized possible use case, which could potentially utilize the model, is if a set of combinations is valid or contains anomalous combinations. The authors are considering applications in NLP, i.e., an automated validation of whether descriptions and statements in written text contradict each other.

The authors plan to implement the proposed NN model into a CAPP software at said automotive manufacturer and accompany and observe this implementation closely to gain further insight on the performance and future applications. Using the gained knowledge of this proposed model, we plan to enhance CAPP in the future by developing NN models capable not only of validation but also of proposing certain processes and whole set-ups based on a given incomplete set of processes. This might require the adjustment of the network to a directed multigraph with loops since some processes might be possible to combine after a certain process is already applied but not the other way round. This should then enable a semi-automated CAPP where certain core processes are manually determined, and the NN proposes fitting processes required to complete the full process combination without the set-up of a comprehensive rule set.

Acknowledgements The research was prepared within the framework of the doctoral program of the Institut für Informationsmanagement im Ingenieurwesen at the Karlsruhe Institute of Technology. The authors thank all colleagues and former colleagues who provided data, concepts, and development support of the research work within Mercedes-Benz Group AG. The authors particularly thank Olaf Buckmann, Sascha Frede, and Verena Fröhlich for supporting the data acquisition.

Funding Open Access funding enabled and organized by Projekt DEAL. This research is funded by the Mercedes-Benz Group AG.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Appendix A: set-up of a connection matrix

The connection matrix using the applied restrictions, which required edges between all active process groups and loops if a process group is active twice, is computed by algorithm 4:

Algorithm 4 Create a connection matrix

Input:
Process combination set P
Output:
Connection matrix M of size $V \times V$

Set $i = 0$, $j = 0$, $a = 0$, $b = 0$
for $i \leq V$ **do**
  for $j \leq V$ **do**
    Set $m_{ij} = -1$
    Set $j = j + 1$
  end for
  Set $i = i + 1$
end for
for $a \leq \|P\|$ **do**
  Set i as the a-th element of P
  Set $b = a + 1$
  for $b \leq \|P\|$ **do**
    Set j as the b-th element of P
    Set $m_{ij} = 1$
    Set $m_{ji} = 1$
    Set $b = b + 1$
  end for
  Set $a = a + 1$
end for

References

Abdi, H., Valentin, D., Edelman, B., et al. (1996). A widrow- hoff learning rule for a generalization of the linear auto-associator. *Journal of Mathematical Psychology, 40*(2), 175–182. https://doi.org/10.1006/jmps.1996.0017

Agyapong-Kodua, K., Haraszkó, C., & Németh, I. (2014). Recipe-based integrated semantic product, process, resource (ppr) digital modelling methodology. *Procedia CIRP, 17*, 112–117. https://doi.org/10.1016/j.procir.2014.03.118

Bauernhansl, T., Hompel, M., & Vogel-Heuser, B. (Eds). (2014). Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung, Technologien, Migration. Springer Vieweg, Wiesbaden, Germany.

Brush, S. G. (1967). History of the lenz-ising model. *Reviews in Modern Physics, 39*, 883–893. https://doi.org/10.1103/RevModPhys.39.883

Chen, Z., Yeo, C. K., Lee, B. S., et al. (2018). Autoencoder-based network anomaly detection. In: 2018 Wireless Telecommunications Symposium (WTS) (pp. 1–5). https://doi.org/10.1109/WTS.2018.8363930

Evermann, J., Rehse, J. R., & Fettke, P. (2017). Predicting process behaviour using deep learning. *Decision Support Systems, 100*, 129–140. https://doi.org/10.1016/j.dss.2017.04.003

Hagemann, S., & Stark, R. (2018). Automated body-in-white production system design: Data-based generation of production system configurations. In: Proceedings of the 4th International Conference on Frontiers of Educational Technologies, New York, NY, USA, ICFET '18 (pp .192–196). https://doi.org/10.1145/3233347.3233373

Hagemann, S., & Stark, R. (2020). An optimal algorithm for the robotic assembly system design problem: An industrial case study. *CIRP Journal of Manufacturing Science and Technology, 31*, 500–513. https://doi.org/10.1016/j.cirpj.2020.08.002

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences, 81*(10), 3088–3092. https://doi.org/10.1073/pnas.81.10.3088

Kretschmer, R., Pfouga, A., Rulhoff, S., et al. (2017). Knowledge-based design for assembly in agile manufacturing by using data mining methods. *Advanced Engineering Informatics, 33*, 285–299. https://doi.org/10.1016/j.aei.2016.12.006

Larochelle, H., Bengio, Y., Louradour, J., et al. (2009). Exploring strategies for training deep neural networks. *Journal of Machine Learning Research, 10*(1), 1–40.

Leo Kumar, S. P. (2017). State of the art-intense review on artificial intelligence systems application in process planning and manufacturing. *Engineering Applications of Artificial Intelligence, 65*(1), 294–329. https://doi.org/10.1016/j.engappai.2017.08.005

Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, ICDM'08 (pp. 413–422). https://doi.org/10.1109/ICDM.2008.17

Liu, Q., Li, X., & Gao, L. (2021). Mathematical modeling and a hybrid evolutionary algorithm for process planning. *Journal of Intelligent Manufacturing, 32*(3), 781–797. https://doi.org/10.1007/s10845-020-01703-w

Mehdiyev, N., Evermann, J., & Fettke, P. (2020). A novel business process prediction model using a deep learning method. *Business & Information Systems Engineering, 62*(2), 143–157. https://doi.org/10.1007/s12599-018-0551-3

Michalos, G., Fysikopoulos, A., Makris, S., et al. (2015). Multi criteria assembly line design and configuration—An automotive case study. *CIRP Journal of Manufacturing Science and Technology, 9*, 69–87. https://doi.org/10.1016/j.cirpj.2015.01.002

Michels, A. S., Lopes, T. C., Sikora, C. G. S., et al. (2018). The robotic assembly line design (rald) problem: Model and case studies with practical extensions. *Computers & Industrial Engineering, 120*(C), 320–333. https://doi.org/10.1016/j.cie.2018.04.010

Ming, X. G., & Mak, K. L. (2000). Intelligent setup planning in manufacturing by neural networks based approach. *Journal of Intelligent Manufacturing, 11*(3), 311–333. https://doi.org/10.1023/A:1008975426914

Moon, J., Park, G., & Jeong, J. (2021). Pop-on: Prediction of process using one-way language model based on nlp approach. *Applied Sciences*. https://doi.org/10.3390/app11020864

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12*, 2825–2830.

Schmidt, N., Lüder, A., Steininger, H., et al. (2014). Analyzing requirements on software tools according to the functional engineering phase in the technical systems engineering process. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation, Barcelona, Spain, ETFA* (pp. 1–8). https://doi.org/10.1109/ETFA.2014.7005144

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., et al. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation, 13*(7), 1443–1471. https://doi.org/10.1162/089976601750264965

Schuh, G., Prote, J. P., Luckert, M., et al. (2017). Knowledge discovery approach for automated process planning. *Procedia CIRP, 63*, 539–544. https://doi.org/10.1016/j.procir.2017.03.092

Sirois, S. (2004). Autoassociator networks: Insights into infant cognition. *Developmental Science, 7*(2), 133–140.

Spoor, J. M., Weber, J., Hagemann, S., et al. (2022). Concept of an inference procedure for fault detection in production planning. In *The Fourteenth International Conferences on Pervasive Patterns and Applications, Barcelona, Spain, PATTERNS '22* (pp. 10–17)

Stošić, B. D., & Fittipaldi, I. P. (1997). Pattern recognition via ising model with long range interactions. *Physica A: Statistical Mechanics and its Applications, 242*(3), 323–331. https://doi.org/10.1016/S0378-4371(97)00288-4

Tax, N., Verenich, I., La Rosa, M., et al. (2017). Predictive business process monitoring with lstm neural networks. In E. Dubois & K. Pohl (Eds.), *Advanced Information Systems Engineering* (pp. 477–492). Springer.

Zhao, Y., Nasrullah, Z., & Li, Z. (2019). Pyod: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research, 20*(96), 1–7.