Demystifying Reinforcement Learning in Production Scheduling via Explainable AI

Daniel Fischer[1][*], Hannah M. Hüsener[1], Felix Grumbach[1], Lukas Vollenkemper[1], Arthur Müller[2] and Pascal Reusch[1]

[1] Center for Applied Data Science (CfADS), Hochschule Bielefeld Gütersloh, Germany.

[2] Department of Machine Intelligence, Fraunhofer IOSB-INA, Lemgo, Germany.

*Corresponding author(s). E-mail(s): daniel.fischer@hsbi.de;

Contributing authors: hannah_maria.huesener@hsbi.de; felix.grumbach@hsbi.de; lukas.vollenkemper@hsbi.de; arthur.mueller@iosb-ina.fraunhofer.de; pascal.reusch@hsbi.de;

Abstract

Deep Reinforcement Learning (DRL) is a frequently employed technique to solve scheduling problems. Although DRL agents ace at delivering viable results in short computing times, their reasoning remains opaque. We conduct a case study where we systematically apply two explainable AI (xAI) frameworks, namely SHAP (DeepSHAP) and Captum (Input x Gradient), to describe the reasoning behind scheduling decisions of a specialized DRL agent in a flow production. We find that methods in the xAI literature lack falsifiability and consistent terminology, do not adequately consider domain-knowledge, the target audience or real-world scenarios, and typically provide simple input-output explanations rather than causal interpretations. To resolve this issue, we introduce a hypotheses-based workflow. This approach enables us to inspect whether explanations align with domain knowledge and match the reward hypotheses of the agent. We furthermore tackle the challenge of communicating these insights to third parties by tailoring hypotheses to the target audience, which can

serve as interpretations of the agent's behavior after verification. Our proposed workflow emphasizes the repeated verification of explanations and may be applicable to various DRL-based scheduling use cases.

Keywords: Deep Reinforcement Learning, Explainable AI (xAI), Production Scheduling, Captum, SHAP, Hypotheses-based workflow

1 Introduction

1.1 Background and key concepts

While the application of Artificial Intelligence (AI) is gaining more attention in many disciplines (Y. K. Dwivedi et al., 2021; Fast & Horvitz, 2017; Heuillet et al., 2021; R. Dwivedi et al., 2023), such as medicine or education (T.-C. T. Chen, 2023b), the public remains divided on its benefits and risks, and generally lacks a deep understanding of AI and machine learning (ML) in general (Bao et al., 2022; Gillespie et al., 2021; Liehner et al., 2023). Concerns exist regarding ethics, transparency and job displacement (Commission, 2024; Liehner et al., 2023), but these do not not stop AI researchers from pushing the boundaries of the field (Peters & Jandrić, 2019).

One area where AI is becoming increasingly relevant is manufacturing and scheduling. Scheduling is defined as a decision-making process that "deals with the allocation of resources to tasks over given time periods" (Pinedo, 2012, p. 1). The allocation process can be divided into rules, such as a sequence in which machines process production jobs. Here, one way to tackle scheduling problems is Reinforcement Learning (RL). RL is a branch of ML where an agent interacts with an environment and is trained to behave optimally. Typically, RL problems are framed as Markov Decision Processes (MDPs). An MDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$. The set of states $\mathcal{S}$ represents all possible situations in which the agent can find itself, where each state s_t encodes information about the environment at a given time step t. The set of actions $\mathcal{A}$ includes all possible actions the agent can take in any given state. The transition probability function P defines the probability of moving from one state to another given a specific action, encapsulating the dynamics of the environment. The reward function R assigns a numerical value r_{t+1} to each action taken in a particular state, providing immediate feedback to the agent on the desirability of its actions. Lastly, the discount factor γ (ranging between 0 and 1) determines the importance of future rewards, balancing immediate and long-term gains in the agent's decision-making process (Sutton & Barto, 2018a). RL agents can function as a decision support tool to find near-optimal job sequences in short computing times, even for highly customized configurations. This capability is a significant advantage for automated planning systems, enabling near real-time (re-)scheduling (Grumbach et al., 2023).

Recently, deep reinforcement learning (DRL) as a subcategory of RL has become a preferred method for tackling complex scheduling problems. (Kayhan & Yildiz, 2021; Grumbach et al., 2022). DRL leverages deep neural networks (DNNs) to approximate the functions that map states to actions and estimate the associated rewards. This combination allows DRL agents to handle high-dimensional state spaces and complex environments, making them particularly effective in scenarios where traditional RL methods may struggle. By integrating deep learning techniques, DRL enhances the scalability and flexibility of RL, enabling it to solve more intricate scheduling tasks (Kayhan & Yildiz, 2021; Grumbach et al., 2022).

Although DRL performs well on a wide variety of tasks, the underlying motives and reasoning of the agent typically remain opaque. This so called black-box nature poses a significant hurdle for companies and users in terms of trust as well as adoption (Arrieta et al., 2020) and therefore requires explanation. To gain insight into the black-box of models such as DNN and DRL, the field of explainable AI (xAI) is thus aimed at making these models understandable and more transparent to increase trust and acceptance towards it. At the same time, xAI can shed light on possible bias, robustness of predictions as well logical validation and satisfy other objectives depending on the stakeholder (Heuillet et al., 2021; R. Dwivedi et al., 2023; Mohseni et al., 2021; Bekkemoen, 2023). Especially in manufacturing and scheduling, xAI is seldom used and existing approaches rarely take into account domain knowledge of target users (T.-C. T. Chen, 2023b, 2023a).

Within the field of xAI, explainable Reinforcement Learning (xRL) (Puiutta & Veith, 2020) refers to explaining a (D)RL agent's actions during decision-making. xRL is fundamentally tied to interpretable ML and due to this overlap there is less research on xRL in specific. (Dazeley et al., 2023)

For manufacturing and scheduling, xRL on the one hand has to convince the production planners and decision-makers that the agent is trust-worthy and makes reasonable decisions (Heuillet et al., 2021). On the other hand, companies need to ensure that the model is robust (Zhang et al., 2019). In both ways, it might be helpful to generate explanations that take into consideration the domain, audience (e.g., non-experts in AI), and application of the used AI model, instead of creating predominantly mathematical or data science jargon heavy explanations that are difficult to communicate to various stakeholders (Heuillet et al., 2021; R. Dwivedi et al., 2023; Bekkemoen, 2023; T.-C. T. Chen, 2023b).

We schematize this problem by analyzing workflows for xAI in practice. Existing workflows heavily focus the development process and deployment of the explanation model and do not take into account concept drift. Transparent workflows for manufacturing which make intuitive sense to management, engineers, production planners and other stakeholders are rare (Clement et al.,

2023; Tchuente et al., 2024). While approaches of xAI are receiving more scientific interest, a universal solution to making AI – or more narrowly DRL – interpretable has yet to be found (Heuillet et al., 2021). Additionally, many xAI methods lack falsifiability (Leavitt & Morcos, 2020).

1.2 Contribution and Research Questions

We combine two xAI methods and demonstrate how to create falsifiable explanations tailored to AI laymen in a real-world scheduling scenario. We adapt the holistic approach of Tchuente et al. (2024) to guide empirical investigations with xAI in business applications. Our xAI approach involves formulating hypotheses in natural language based on domain knowledge, analyzing the reward function of the agent, and descriptive statistics of use case data. This is to analyze the plausibility of the xAI results and to ensure that the generated explanations can be effectively communicated within the specific domain.

The scope of the paper is to answer the following questions:

- **(RQ1.1)** How can xAI methods, specifically DeepSHAP and Input x Gradient, be systematically implemented, applied, and validated to describe the decisions of a DRL agent in a real-world flow production context?
- **(RQ1.2)** How suitable are the chosen xAI methods for the specified use case and what are their advantages and disadvantages?
- **(RQ2.1)** How can a workflow be developed to integrate hypotheses derived from domain knowledge with xAI methods for scheduling applications, ensuring falsifiability?
- **(RQ2.2)** How can these xAI-explanations be processed and communicated to stakeholders using domain knowledge?

2 Literature Review

We start by laying a common ground on RL in production and scheduling. Afterwards, we introduce xAI methods and workflows before we stress the research gap for falsifiable and systematic xAI approaches in production planning. In the following, we differentiate between frameworks (software tools which *contain* xAI methods) and the methods themselves that can be mathematically formulated. In accompanying literature, proposed xAI workflows are often found under the synonyms "framework", "process", "process model" or "procedure".

2.1 Reinforcement Learning Applications in Production and Scheduling

Referring to the definition by Pinedo (2012), scheduling as is has to be optimized to meet the expectations of stakeholders. There are several possible dependent variables, such as lead times, resource utilization, or simply costs. One way to approach this problem is to solve it heuristically or analytically (e.g., with operations research methods). Although these methods are widely studied and

applied, heuristics tend to find only local optima, while mathematical models are only suitable for smaller problems with a simple search space.

RL provides an alternative to heuristic methods, trying to utilize ML to solve problems more efficiently. Because scheduling offers constant feedback for the agent through its sensors, the agent can navigate well in these environments (Khadivi et al., 2023; Sutton & Barto, 2018b). Over the years, several novel approaches emerged which are highly suitable for scheduling problems: Q-Learning (Watkins & Dayan, 1992) and DRL (see Introduction) (Mnih et al., 2015) to name a few. The last is especially useful when the state space size is not displayable in a table format as former RL algorithms did. DRL outperforms classical RL approaches and Q-Learning on a variety of scheduling tasks, see Khadivi et al. (2023) for an overview. For example, Serrano-Ruiz, Mula, and Poler (2024) use OpenAI Gym to model a quasi-realistic job shop scheduling environment using a digital twin based on a MDP and DRL with the proximal policy optimization algorithm, incorporating an observation space with more than a dozen job features and an action space with three heuristic priority rules, which demonstrated superior multi-objective performance compared to traditional heuristic rules.

An advantage of these DNN lies in their ability to model complex structures of the state space. The agent is then able to use this information for its decisions. A major downside is the black-box structure. Even if the reward function is chosen carefully, the resulting actions and the reached solutions seem promising, no conclusions can be drawn about the *reasons* behind the agent's decisions. It is expected that there is a huge information potential hidden in networks used for DRL (Kuhnle et al., 2022). In order to reveal these hidden information and enhance the explainability of DRL models, several approaches have been proposed.

2.2 State of the art xAI frameworks and methods

Before diving into the xAI approaches, a common terminology has to be laid out as Palacio et al. (2021) propose in their framework for unifying xAI.

Following this framework, the *explanation* is the description of information made to create understanding by humans. *How* explanations are created is defined by the xAI *methods*. Here, the methods need to specify what is needed as input and in return what output is produced. On the other hand, the *interpretation* is the meaning that is associated with the explanation. This also relates to *why* this meaning is assigned (e.g., a causal link or the context). (Palacio et al., 2021)

Other researchers argue that in the field of AI, interpretability ("interpretable AI") refers to a lower complexity of the model so that its components can be analyzed and understood (e.g., small decision trees are interpretable) intrinsically. These types of algorithms are sometimes labelled 'transparent'. While xAI approaches can enhance model transparency already during development,

for black-box models that are not intrinsically interpretable or transparent, explainers are oftentimes used. These additional algorithms can explain the outcomes after the original model was executed. This is also referred to as *post-hoc* explainability. So, while interpretability may also refer to the complexity, explanations specify the reasoning that lead to a behavior. Following this definition, an AI-model could be interpretable without being explainable and vice versa. A fully explainable model or system is supposed to create explanations for it's own actions. The overarching concept of this is sometimes termed 'Transparent AI'. (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; Milani et al., 2024)

In the following, we refer to explanations as the output of the xAI methods and interpretations as the meaning as well as conclusions that can be drawn from them. We will further this later on.

Within xAI approaches, four important differentiations have to be made. There are model specific and model agnostic methods, as well as global and local explanations. Methods can either be applicable to only a specific model and algorithm or be used for all types of ML models. How easily xAI methods can be used for different underlying ML models is sometimes called portability. Explanations can be created for the entire model, only specific predictions (e.g., local or instance explanations for single actions) or both (Heuillet et al., 2021; R. Dwivedi et al., 2023; Mohseni et al., 2021; T.-C. T. Chen, 2023b; Milani et al., 2024).

2.2.1 The choice of explanation type

Various xAI methods can produce different explanation types for different use cases or needs (for an overview see Mohseni et al. (2021) or T.-C. T. Chen (2023b)). The choice what type to use is highly dependent on the target audience, which means that one has to consider the background knowledge of the explainee. An AI expert could be interested in detailed explanations for debugging, while laymen might only want to understand enough to decide whether or not they can rely on the AI system (e.g., via explanations or more transparency on the algorithm) (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; R. Dwivedi et al., 2023).

Researchers can use xAI to describe how the entire model works, for example by using global explanations (Mohseni et al., 2021; T.-C. T. Chen, 2023b). In xRL, there are approaches aimed at helping developers debugging their model by better understanding the states, actions and their outcomes (Dazeley et al., 2023). Quality control might also play a role. For example, Klar et al. (2024) use policy summarization (Wells & Bednarz, 2021), state value evaluation and perturbation methods to filter out random explanations. This way they ensure that random solutions are not influencing the outcome of the planning of a factory layout.

Example-based explanations concentrate on explaining selected instances in the data, e.g., by using local explanations for describing a single action of the agent (T.-C. T. Chen, 2023b). However, single explanations are criticized for being insufficient to explain the workings of complex models (Leavitt & Morcos, 2020).

Other types of xAI include what-if explanations (to show how changes influence the output), contrastive explanations, which are aimed at highlighting why certain outcomes did not happen (e.g., "Why not another action?"), and counterfactual explanations to demonstrate which changes in the input or model would lead to a different outcome (e.g., "What has to be observed for a different action?"). These explanations are considered to be easily understandable for humans, but they can be challenging to use in large state spaces or in some cases need specifically generated features. (Dazeley et al., 2023; Mohseni et al., 2021; M. Y. Kim et al., 2021; T.-C. T. Chen, 2023b)

Explaining the reasons for a specific outcome based on an input is termed a *why* explanation (Mohseni et al., 2021). Within why-explanations, feature-based techniques show how the input features change the model output by assigning importance values (R. Dwivedi et al., 2023).

In the next sections, we will explain some xAI methods in detail. This is not intended to be a complete overview, which is not the scope of this paper.

2.2.2 xAI Methods

One of the ways to categorize xAI methods for deep learning is into intrinsic, perturbation- and gradient- or backpropagation-based (T.-C. T. Chen, 2023b; Kamath & Liu, 2021). We first introduce decision trees, a popular and intrinsically interpretable algorithm that is broadly used. Then, we outline SHAP as well as several perturbation- and gradient-based xAI methods.

Decision Trees

Decision trees are widely used and offer explainability out of the box (Kotsiantis, 2013). Starting with the work of Breiman, Friedman, Olshen, and Stone (1984) the foundation of modern decision trees has been laid. With his CART (classification and regression tree) the rules discovered by the algorithm are displayable in a transparent form. Decision trees are highly versatile, e.g. for regression and classification. In the original CART algorithm, on each split the information gain has to be maximized which relies on information entropy (Quinlan, 1986) and the concept of information content (Shannon, 1948).

Decision trees have been used in a wide range of domains, including stock market (Wu et al., 2006), marketing (J. W. Kim et al., 2001), image classification (C.-C. Yang et al., 2003) and scheduling (Portoleau et al., 2024).

SHAP (SHapley Additive exPlanation)

SHAP is a framework introduced by Lundberg and Lee (2017). The authors show that multiple xAI methods such as Lime (Ribeiro et al., 2016) and DeepLIFT (Shrikumar et al., 2017) are additive feature attribution methods. Because of the complex nature of blackbox models, (local) explanation models g approximate the original model f to make a prediction $f(x)$ based on an input x interpretable. Oftentimes inputs are simplified x' and can be mapped back into the original input space using a mapping function $x = hx(x')$. In the class of additive feature attribution methods, the explanation model can be written as a linear function: Starting from ϕ_0, the model output where all simplified inputs are missing, the simplified input features (binary variables) are being attributed an effect ϕ_i if they are present. Then, all feature attributions are summed, giving an approximation of the original model prediction. (Lundberg & Lee, 2017)

When the values ϕ_i used are Shapley values from cooperative game theory (Shapley, 1953) three properties (local accuracy, missingness, and consistency) for meaningful explanations are satisfied (Lundberg & Lee, 2017). Note that Shapley values are permutation-based (Shapley, 1953). SHAP values are "Shapley values of a conditional expectation function of the original model" (Lundberg & Lee, 2017, p. 4), where every feature is assigned an importance value that shows how the expected model prediction changes when conditioning on that specific feature. The starting point of this additive attribution is the so called base value $E[f(z)]$, which represents the expected prediction if all feature values were unknown. (Lundberg & Lee, 2017)

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z_i',$$

where $z' \in \{0,1\}^M$, M is the number of simplified input features, and $\phi_i \in \mathbb{R}$.

When input features are correlated or when dealing with non-linearity, the order of adding features to the expectation is relevant; however, to simplify computation independence between features is usually assumed in SHAP (Lundberg & Lee, 2017). Due to permutation and especially if features are dependent, this can result in individual SHAP values for unrealistic data points (Aas et al., 2021; Kamath & Liu, 2021), which imposes a limitation of the SHAP methods.

While the exact computation of SHAP values for real-world data is difficult, the authors propose model-agnostic and model-type-specific methods for approximation, whereas using model-specific information can improve computational performance. Using SHAP, both local and global explanations can be created (R. Dwivedi et al., 2023). SHAP has previously been used for xAI in manufacturing (T.-C. T. Chen, 2023b).

DeepLIFT

DeepLIFT (Shrikumar et al., 2017) is an additive feature attribution method (Lundberg & Lee, 2017). It recursively explains a DNN's predictions by comparing the original activation of each neuron to a reference (background) value for every input and assigning an importance score to individual parts of the input (Lundberg & Lee, 2017; Shrikumar et al., 2017). In DeepLIFT, the contribution of one neuron to another is defined to satisfy the properties of summation to delta and linear composition, which is achieved using the backpropagation rules proposed by the authors (see (Shrikumar et al., 2017) for more). A limitation of this method lies in the need to determine an appropriate reference.

In the prediction of weaning from mechanical ventilation, DeepLIFT has been used to ensure that a convolutional NN makes predictions using clinically important features (Jia et al., 2021).

DeepSHAP

The DeepSHAP explainer is specific for deep learning models, utilizing their compositional structure (Lundberg & Lee, 2017). It builds on DeepLIFT (Shrikumar et al., 2017). If the input features are independent, the neural network is linearized, shapley values are chosen as attribution values and the reference value is taken to be E[x], DeepLIFT approximates SHAP values. Making use of DeepLIFT's form of back-propagation (multipliers m), DeepSHAP combines the SHAP values for a feature i and the prediction y for the network components (e.g., f_3) – that can be solved analytically if linear – into SHAP values for the entire network. (Lundberg & Lee, 2017)

$$\phi_i(f_3, y) \approx m_{y_{i f_3}}(y_i - E[y_i])$$

A limitation of DeepSHAP lies in its dependence on the background input that must be chosen to compute the mean prediction (Fernando et al., 2019). Stability improves with larger background sample sizes (Yuan et al., 2022) .

Recently, DeepSHAP has been used to explain the results of a DNN in the field of condition monitoring for hydraulic systems (Keleko et al., 2023).

Input X Gradient

The method Input X Gradient (or *gradient*input*) works by taking the partial derivatives of the output with respect to the input and multiplying them with the input itself (Kindermans et al., 2016):

$$R_i^c(x) = x_i \cdot \frac{\partial S_c(x)}{\partial x_i}$$

where $R_i^c(x)$ is the contribution of each given input x_i, $S_c(x)$ is the output function and $\frac{\partial S_c(x)}{\partial x_i}$ is the partial derivative of the output function with respect to the input. While the partial derivative itself can provide information regarding

how an infinitesimal change in the input influences the output, Input X Gradient goes one step further (Adebayo et al., 2018). Also being able to detect gradient saturation is a huge benefit in contrast to the plain gradient consideration.

One upside of Input x Gradient lies in its monitorability. For attribution values, decision boundaries and critical thresholds can be defined. For example, if the attribution for a certain action contradicts the 68–95–99.7 rule (Wooditch et al., 2021), an alert may be sent out. Input X Gradient is sensitive to scaling (Sundararajan et al., 2016). Leavitt and Morcos (2020) mention that Input x Gradient can lead to unreliable explanations when considering single neuron activations. False claims could be made by making the false assumption that a single neuron may be *causally* responsible for a specific action of the agent. This disregards the complex interactions between the neurons in the whole network.

Chatterjee et al. (2024) use Input X Gradient to colour specific areas of the lung by patients eventually suffering from COVID-19. Ozer, Guler, Cansever, and Oksuz (2023) also use this technique for x-ray images.

Layer-wise relevance propagation

Layer-wise relevance propagation (LRP) uses heatmaps to show single pixel contributions to an output (Bach et al., 2015; Binder et al., 2016). It works by a special layer-to-layer backpropagation method which shows the contribution of every neuron to the prediction (Montavon et al., 2018). Each layer receives so called relevance scores from the succeeding layer and redistributes them proportionally to its inputs. There is not a method of choice in calculating these scores, so the modeller is free in determining what "relevance" means in the given context. The $\alpha\beta$ rule, among others, tries to balance relevance between positive and negative contributions (Kohlbrenner et al., 2020).

Y. Yang, Tresp, Wunderle, and Fasching (2018) use LRP in a medical domain to generate several therapy suggestions. They validate the explanations with experts to support their findings. Arras, Horn, Montavon, Müller, and Samek (2016) apply LRP to text classification and identify important words in documents.

Since we are explaining an RL-model, we researched general xAI methods as well as specific methods in xRL. Different xRL methods exist, but as Heuillet et al. (2021) point out many of those are not applicable to RL in real-world scenarios: One reason being that in RL, there are specific assumptions, algorithms and environments with different constrains to consider. The authors propose to focus research on global xAI approaches. In fact, we find approaches in literature where DRL results were explained using general xAI methods (such as SHAP or even decision trees) that are not specific to RL (Eikså et al., 2024). Nonetheless, the next section reviews state of the art xRL methods.

2.2.3 xRL

Xiong et al. (2024) state that in xRL there a multiple different approaches and respective methods: One can explain the model logic, the reward function (by decomposition or shaping), the states and the tasks. Specifically for explaining the states, post-hoc methods such as SHAP may be used.

Milani et al. (2024) propose to categorize xRL into methods for feature importance (e.g., local state–action relationships), training and MDP (f.e., what objective is prioritized), as well as policy-explanations (global long-term behavior explanations), so that they align closer with RL-logic. For feature-importance, there are different explanation formats and various approaches. Methods can range from using surrogate models to make the policy interpretable, to using inherently interpretable models that can structure policy information (such as decision trees), up to simply explaining what states were important for an action using visualization (e.g., saliency maps) or natural language. Regarding training and MDP, the authors split xRL methods into three categories. Some methods use the transitions and explain the agent's behavior with causal models. Another method is reward decomposition, where – before training – the reward function is constructed in a way that comprises different reward types with a meaning (e.g., to explain that a specific reward component is more important than another for certain actions). However, the authors point out that this technique only works for specific Q-learning algorithms. Another method in xRL regarding the training and MDP consists of identifying and inspecting data points that were most influential in the training of the agent. Lastly, policy-explanation methods either inspect long-term agent behavior for important states in training or for clustered similar states. Another approach is converting the NN that represents the policies into an understandable representation.

The meta review by Bekkemoen (2023) sheds light on xRL explanations and stakeholder needs. They develop a new taxonomy which differentiates between three types of xRL agents: Interpretable Agents, Intrinsic Explainability and Post Hoc Explainability. Interpretable agents are intepretable out of the box with a single function approximator. Intrinsic explainability describes preparation of the RL system before training to make it explainable. Post hoc explainability aims at describing the agent *after* training. Because intrinsic and post hoc explainability overlap, they developed additional criteria for differentiating these agents. After categorizing agents, they map several question categories (by "How (to)?", "What?", "Why (not)?", "What if?) onto the discovered agents and their capabilities. They also stress the importance of the validity period and method used.

Regarding intrinsic explainable DRL, in the field of network slicing, Rezazadeh, Chergui, and Mangues-Bafalluy (2023) developed a method that uses the reward hypothesis to create an xAI reward for the agent by combining SHAP and entropy values (Shannon, 1948) to increase decision certainty. This xAI reward

is paired with the task-related reward in the training phase for a resource allocation optimization problem. The authors show superior performance of the explainable DRL agent in comparison to standard DRL.

Asides from explainability, Xiong et al. (2024) highlight that there is a lack of standardized evaluation metrics to compare xRL methods. They categorize evaluations into subjective (e.g., user's understanding and trust) and objective. In their xRL-Bench, they focus on the latter, specifically on fidelity (e.g., faithfulness: Does the explanation match agent's logic?) and stability (consistency of explanations). This is assessed by either masking or perturbing important and unimportant states and inspecting deviations in the model outputs before and after. Milani et al. (2024) also discuss metrics for xRL methods and review which metrics are used in literature. Interestingly, they find that fidelity and understandability of xRL are rarely assessed, while visualizations are most popular. Though not quantitative, visualizations are often utilized to illustrate interpretability.

2.2.4 Frameworks for xAI workflow and design decisions

Asides from the methods themselves, we must also consider the procedure of generating xAI in a complex business setting. This includes the workflow, design choices as well as other aspects to look out for. We draw on frameworks from both the xAI community as well as other domains.

Leavitt and Morcos (2020) point out that approaches of xAI, especially for deep learning, often rely on visualization or single examples and lack falsification via hypotheses, quantifiability and human as well as general verification of validity. T.-C. T. Chen (2023b) also highlight that xAI results need to be validated. This is something that should be kept in mind in the workflow and design of explanations, in order to ensure a scientific standard.

Mohseni et al. (2021) propose a framework for xAI design and evaluation. In their nested model, the outer layer focuses on the outcomes. When designing an xAI system, general considerations have to be made first: What is the goal, the target group and what is supposed to be explained? This also involves determining how to evaluate the xAI system to measure if the expectations were met. In the next layer, the process of explaining has to be decided on with the user in mind. This includes aspects like the explanation formats and the amount of details to include, while also evaluating the usefulness of the chosen explanations. The inner layer is the type of xAI method itself. When black-box models are explained with ad-hoc algorithms, fidelity to the original model plays a role. The trustworthiness of the underlying model should also be evaluated.

In their framework for design and analysis of xAI, M. Y. Kim et al. (2021) outline the historical development of explanations and point out that explanations should to be scientific or at least causal. Dazeley et al. (2023) also highlight the importance of causality and introduce the Causal xRL Framework to create causal explanations in RL. They draw on two existing theories. First, Dazeley

et al. (2021) suggest that xRL can take place at different orders and that higher level explanations should be incorporated for acceptance of AI systems. Zero-order explanations only consider how an input resulted in an output (base case in xAI). First-order explanations that consider an agent's objective to maximise a reward signal are also relevant. Explaining the agent's intentionality (e.g., the objectives behind the behavior) might improve understanding of the agent, in comparison to only giving zero-order explanations (Dazeley et al., 2021, 2023). Secondly, due to the temporal sequence of transitions in RL, Dazeley et al. (2023) propose to provide causal explanations. They draw on the Casual Explanation Network by Böhm and Pfister (2015) , which aims at explaining intentionality, cause and reason of behavior as well as implications for the future. In our setting, the Causal xRL Framework can be understood as following: The agent is expected to make production decisions based on two objectives (e.g., goals). For this, the agent observes the state (perception, zero-order) and depending on the attributes of the features one objective might be prioritized (disposition, first-order). To fulfill the objective (first-order), the agent will then make an action that will in return cause an outcome (zero-order). If an agent has more than one objective the explanation of disposition plays a role, which Dazeley et al. (2023) point out as an outlook for future research. Goal-driven xAI approaches are currently an emerging research field (Dazeley et al., 2023). However, it is mainly a focus in autonomous agents and robots, while research on xRL and reactive agents in particular is limited to policy retention without including domain knowledge (Sado et al., 2023).

Shi et al. (2023) widen the circle of people involved by incorporating domain experts and researchers together. Their approach is suited for manufacturing and consists of a prototype of a task guiding system. With a graphical user interface they summarize information about different models and agent behaviour. This makes it easier for third parties to interact with the explanations. Langer et al. (2021) build upon that and create a model to focus on the different *stakeholder desiderata*. Their model classifies different stakeholders and their demands regarding xAI explanations. Different stakeholder classes like deployers, regulators, users and "affected" are presented. One key insight are the highly versatile expectations of the different stakeholders. These expectations are already affected by the different domain knowledge of the parties. The intersection of these interests has to be found to provide a reasonable xAI model.

Asides from the xAI literature, we also researched workflows from other domains. From a broad perspective, using a unified, *general* framework which is well-founded in the literature makes sense. The **CR**oss **I**ndustry **S**tandard **P**rocess for **D**ata **M**ining framework (CRISP-DM) by Wirth and Hipp (2000) provides a holistic view on all internal departments and decision processes involved for deploying a data mining model. However, it does not focus on xAI models specifically and does not contain two layers for the explained model and the explaining model. The model is put to production in the Deployment stage.

After that, the framework stops. CRISP-DM is thus not feasible to guide xAI projects after deployment and to verify their explanations. It is also not suitable for environments where a lot of the constraints change and the model has to be adapted. Several researchers tried to improve on that.

Tchuente et al. (2024) specifically reviewed xAI in business settings and proposed a workflow for xAI in business applications, which is discussed in detail in our approach 3.4. As limitations of their workflow, the authors highlight that it has to be adapted to specific contexts and further work is need regarding robustness and validation of explanations. Later on, we directly take up these limitations with our approach.

2.3 Research Gap

We did not find many established frameworks for xAI in scheduling. The amount of xRL methods suitable in real-world scenarios is limited (Bekkemoen, 2023; Heuillet et al., 2021). Also, many xAI methods we reviewed were suitable for debugging, but not for non-AI experts. Here, the questions arises which xAI methods may be applied to describe the decisions of a DRL agent in a real-world flow production context (RQ 1.1)? How can they systematically be implemented, applied, and validated (Chi & Liao, 2022)? Every use case poses different challenges for the developer to consider. Explanations may vary according to the questions which arise. Identifying the most suitable xAI method as well as weighing its (dis-)advantages is crucial (RQ 1.2).

Many efforts in xRL only provide zero-order explanations, which are not comprehensive enough to create trust and acceptance in the AI system and only few approaches include utilizing an agent's objectives or dispositions to create casual explanations (Dazeley et al., 2021, 2023). Adding domain knowledge to xRL is important (Milani et al., 2024), but considering it in xAI approaches for manufacturing is rare T.-C. T. Chen (2023b). Here, the questions arise how domain knowledge as well as context can be included into an explanation. How can xAI-results be processed and presented to stakeholders using this knowledge (RQ 2.2)?

In the xAI community, researchers have criticized the lack of unified terminology (Palacio et al., 2021) as well as falsification and missing validity checks (Leavitt & Morcos, 2020; T.-C. T. Chen, 2023b). How can falsifiablity be ensured (RQ 1.1, 2.1)?

We fill this gap by developing a workflow based on hypotheses (RQ 2.1), utilizing the domain knowledge (RQ 2.2) of the users and knowledge of the agent's workings. The framework builds upon existing xAI methods and -frameworks and focuses on real world scheduling applications (RQ 1.1, 1.2).

3 Approach

In the following section, we illustrate our real-world use case and the scheduling problem formulation, as well as the data set were are working with. Then, we state our xAI workflow with the aim to fill the research gap.

3.1 Preliminaries

To gain a deeper understanding of our use case, we now formulate the scheduling problem and the MDP to address it.

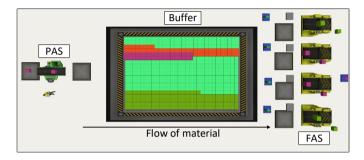### 3.1.1 Real-world case and scheduling problem formulation

Fig. 1 Model of the considered two-stage flow production system by Müller, Grumbach, and Kattenstroth (2024).

The systematic application of xAI techniques is examined in the context of a real-world use case at a large German manufacturer of household appliances. In previous publications, an extensive scheduling model, along with a specialized DRL agent, has been investigated, implemented and tuned (Müller, Grumbach, & Kattenstroth, 2024; Müller, Grumbach, & Sabatelli, 2024). The considered manufacturing process involves a two-stage flow production system. As visualized in Figure 1, the shopfloor consists of a pre-assembly stage (PAS) and a final assembly stage (FAS). In the PAS, a single station produces eight types of semi-finished products (hereinafter referred to as products), which are then finished at one of four possible FAS stations. Between these two stages, there is a limited intermediate buffer where the products are temporarily stored. According to Müller, Grumbach, and Sabatelli (2024), the key model components of the extended permutation flow shop can be defined semi-formally as outlined below. For a comprehensive mathematical description, refer to Müller, Grumbach, and Kattenstroth (2024).

- Given a set of specific products to be finalized in the FAS, each based on a standardized product of eight different types produced in the PAS.
- All PAS and FAS stations can process only one product at a time and all processing times are deterministic.

- Each product must be finalized at exactly one of four FAS stations. In this context, each FAS has a predefined schedule containing the sequence of single products to be manufactured.
- A FAS station can only start finishing the product when it is available in the central buffer. If the next required product is not available in the central buffer, the FAS station will incur idle times until it becomes available.
- In the PAS, all products required by the FAS that are not initially available in the central buffer must be produced.
- Switching from one product type to another may cause sequence-dependent setup efforts in the PAS.

Given these constraints, two conflicting objectives are optimized in lexicographical order: first, minimizing idle times in the FAS, and second, minimizing setup efforts in the PAS. This problem is addressed by determining a central decision variable: the sequence in which products are loaded into the production system in the PAS. This decision is managed by the existing agent, which will be described in detail in the following section.

3.1.2 Existing RL Approach

The real world setting stated above needs to be formalized. Several approaches are feasible here. For example, the environment could be modelled using operations research methods. Because of RL-agent's abilities to efficiently find local optima, we chose DRL. The scheduling problem is modelled as an MDP as presented below. For a detailed explanation, refer to (Müller, Grumbach, & Kattenstroth, 2024).

State Space

The state space is encoded as a vector and comprises following elements:

- **next_24h_demand_prod** for all products: Specifies the demand of the related product for the next 24 hours, taking into account the amount left in the buffer.
- **end_of_planning_period_demand** for all products: Specifies the demand of the related product for the planning period left, taking into account the amount left in the buffer.
- **buffer_content_duration_prod** for all products: Specifies how long the amount of an product type in the buffer will suffice to meet the demands of the FAS if this type is no longer produced in the PAS.
- **buffer_fill_level**: Specifies the fill level of the buffer, i.e., the amount of all products in relation to its total capacity.
- **last_prod_type_is** for all products: Specifies the last type of products manufactured, represented in one-hot encoding.

Action Space

We define the action space to be discrete. It consists of 8 actions representing the 8 different proudct types with 0 centered start (e.g., action 7 = agent recommends to build product 8). Each time the agent selects an action, a lot of 50 units of the corresponding product type is produced.

Reward Function

The primary objective of the agent is to avoid idle times. Rather than penalizing idle times directly, we penalize *criticality*, which we define to be the ratio of `next_24h_demand_prod` to `buffer_content_duration_prod`. This has proven to be significantly more effective for training, as criticality provides a richer learning signal compared to idle times (Müller, Grumbach, & Kattenstroth, 2024). Furthermore, we penalize the agent if `buffer_content_duration_prod` for any product type falls under a threshold of 30 minutes, which encourages the agent to maintain a certain margin for each required product type. The other objective is to minimize setup efforts. Therefore, we add setup efforts weighted by a factor directly to the reward function.

Domain Randomization

To make the agent decisions more robust, domain randomization (DR) has been used (Müller & Sabatelli, 2023). DR consists of the idea to present not one but many environment distributions to the agent at each episode. The agent is forced to adapt to a wider spectrum of scenarios, making its policy more robust. In the concrete use case, six different weeks (environments) were presented to the agent at random. Each week had distinct characteristics regarding demand and buffer sizes.

Dataset

The agent made scheduling decisions based on representative synthetic data that were carefully designed to reflect realistic scenarios. These data were obfuscated to preserve the integrity of the study and protect the interests of the project partner, ensuring no sensitive or proprietary information was compromised. After giving the input values, which consist of checkpoints for the weeks, the agent and environment was initialized and while the agent acted in the environment, we saved the state and action pairs. The generated data frame consists of 103 rows and 35 columns, corresponding to the actions of the agent and the state space.

We started our investigations with an already trained NN; therefore, we consider the whole dataset as a test set. Thus, we achieve the same attribution values for each run through the data in Input X Gradient. Settings where the network is trained again on new data may lead to the necessity to use cross validation and to fit confidence intervals for the attribution values to achieve robustness. We want to note that we had to rebuild the NN, because the original was not available to use in the xAI methods due to it being a custom class.

Now, how can we systematically explain the data of the agent from our use case using xAI?

3.2 Domain-knowledge hypotheses to combine falsification, interpretation and communication of xAI

As we have pointed out, many efforts in xAI, and xRL specifically, provide zero-order explanations and only few include utilizing an agent's objectives or dispositions to create casual explanations (Dazeley et al., 2021, 2023; M. Y. Kim et al., 2021). Also, domain knowledge is not considered adequately in the context of manufacturing (RQ 2.2) (T.-C. T. Chen, 2023b). Thus, we want to utilize the reward function and pair it with domain knowledge (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; R. Dwivedi et al., 2023; Milani et al., 2024) to create 'higher-order' explanations (Dazeley et al., 2023). Note that following Palacio et al. (2021) the (causal) context and the 'why' behind an AI system's behavior would be part of the interpretation and not the explanation itself. Also, we want to take into account the importance of falsification in xAI (Leavitt & Morcos, 2020).

We postulate that all of these aspects can be combined in a cohesive way. In order to make explanations interpretable and to ensure user's trust, we need to add context as well as knowledge of the agent's workings (Dazeley et al., 2023; Mohseni et al., 2021), while keeping the target audience in mind (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; R. Dwivedi et al., 2023; T.-C. T. Chen, 2023b). Importantly, these are all information we can put together before the explanation is created: If we have a (causal) understanding of what a trained agent should do in a specific use case and situation based on domain knowledge and the agent's objectives (decomposing the reward into meaningful statements), we can formulate a hypothesis (Mohseni et al., 2021; Dazeley et al., 2023; Leavitt & Morcos, 2020; Milani et al., 2024). For example, we could say: *In the production line, it is important to keep products in buffer. In production scheduling, an agent is punished if the buffer content is too low. Therefore, in a situation where buffer level is low, the agent should produce more products to fill the buffer.* This hypothesis can then be applied to an explanation created with an xAI method (e.g., a method showing that a specific feature was important for the behavior of the agent). If the hypothesis matches the explanation, we conclude that the AI system operates how we expected and we have already created a humanly comprehensible interpretation that involves context and causality (e.g., the relevant feature indicated that buffer content was low, which is why it was relevant in the decision of the agent to fill the buffer). If the hypothesis does not match the explanation created by the xAI method, we have detected an error and need to revise the AI system or integrate further context knowledge. By doing this, we combine falsification using hypotheses (Leavitt & Morcos, 2020) with (causal) interpretation (Palacio et al., 2021; M. Y. Kim et al., 2021) and adding higher-levels for trust and acceptance (Dazeley et al., 2023). At the same time, we include the background knowledge of the users who

are domain experts (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; R. Dwivedi et al., 2023; Milani et al., 2024). This specific combination is a novel approach for xRL in a real-world business setting to the best of our knowledge. Thereby, we pursue to develop an approach for xAI in business and logistics, specifically using production scheduling as an example, where experts want to quickly understand whether the output of the AI system aligns with their domain knowledge or not.

3.3 The choice of xAI methods for our use case

Regarding the choice of xAI methods, one has to consider the goal and the target audience (Mohseni et al., 2021; M. Y. Kim et al., 2021; Heuillet et al., 2021; R. Dwivedi et al., 2023). In our use case, we are neither interested in interpreting static components of the model, nor are we trying to create a self-explaining model as is a general goal in some xAI research (Mohseni et al., 2021; M. Y. Kim et al., 2021). We are using an existing DRL-model to explain (post-hoc) why the agent made it's decisions.

It is important to highlight that debugging or explaining the technical details of the entire model is not suitable for our target audience (Mohseni et al., 2021; Dazeley et al., 2023). Our approach is aimed at domain experts in production scheduling with user trust and acceptance in mind. The goal is to provide a simple explanation to understand why the DRL agent created a specific production plan. This is why we do not use contrastive or counterfactual explanations, but instead focus on why the specific production decisions were made – leaving what Mohseni et al. (2021) term 'why'-explanations best suited.

In our use case, features – such as the demand or buffer content – are key components in scheduling; therefore, it seems intuitive to use them to create domain-based explanations. Which is why we chose feature-based xAI techniques to show how the input features influence the model output by assigning importance values (R. Dwivedi et al., 2023).

For RL in real-world scenarios, Heuillet et al. (2021) propose to use broad xAI approaches. We follow this suggestion by utilizing model-agnostic xAI frameworks, namely SHAP and Captum.

To sum up, 'why'-explanations concentrating on the features in the data and model-agnostic xAI frameworks are most suitable for our use case.

In order to provide alternative explanation-styles, which can be helpful for the explainee (M. Y. Kim et al., 2021), we chose one method – DeepSHAP – that primarily relies on visualization, and another method – Input X Gradient – that is more quantifiable and can be presented in a table-format. We create explanations for every class of action, instead at looking at single instances.

For the interpretation of SHAP values, we follow the available SHAP documentation and S. Chen (2021). Attribution values returned by Input X Gradient can be interpreted as standard attribution values. Positive attribution values

increase the chance of a specific action being chosen, while negative attributions decrease the probability (Das & Rad, 2020).

3.4 The choice of a systematic workflow

Our approach uses the workflow by Tchuente et al. (2024), who provide a robust structure for guiding empirical investigations with xAI in business applications in general (holistic approach).

The workflow by Tchuente et al. (2024) is specifically adapted to xAI models and anticipates changes in the assumptions of the base model. They include a wide range of business parties in their process to mitigate reservations of sceptics.

Fig. 2 Framework for xAI in business by Tchuente et al. (2024). Idea, data and context are presented as three clusters.

In the following we describe the framework more thoroughly as a basis for our approach to the research questions.

3.4.1 Business question identification and importance scoring

The framework by Tchuente et al. (2024) starts with the identification of the business question and its importance (i.e. to the stakeholders). Important (target-)variables may be identified. In order to achieve this, several techniques may be used. It is important to verify that the results stems from a collaboration of many stakeholders to ensure a wide range of opinions and expertise is included in the questions to be answered. The process may be led by engineers or data scientists to ensure that the questions are in fact answerable by algorithms

at hand and that the desired objectives are realistic. Lastly, the objectives can be enriched with the sources of the data to help speeding up the next phase. A proper example in this context could be the business question, how the lead time of an exemplary product can be reduced and how these results can be communicated to the stakeholders. Here, the target variable is the lead time. Independent variables could be production times on several machines, dummy variables indicating special product configurations etc. It is important to evaluate what persons may be involved and to include their expertise.

3.4.2 Data collection

After clarifying the desired objective and variables, the data collection starts. Incorporating Data Engineering here might help with the collection. It makes sense to store batch data at disk and to create a pipeline for data streams to guarantee that it is available straight away when needed. The raw data might originate from databases, data warehouses or even pdf-documents. We refer to the widely known ETL process for further reading (Vassiliadis et al., 2002).

3.4.3 Data preprocessing and feature engineering

Before we can make the data available to the model, we have to handle missing values (Tchuente et al., 2024). The pipeline used here must be transparent and documented to ensure that third parties can follow which data related decisions had been made (Chakraborty et al., 2017). These can highly influence the results of the model and explainer, i.e. if important structures in the data are deleted.

3.4.4 Fitting and validation of the model

More important than training the model and validating it, the process of choosing the right model and its communication is key (Nyawa et al., 2023). Tchuente et al. (2024) emphasize the process of evaluating which model is the most precise. This highly technical decisions may not be interesting to stakeholders at first sight, but may offer problem misconceptions or valuable insights for others. Communication is key even in this phase. At the end of this thought process, the adequate model or model ensemble is found. Explaining its limitations and interpretability is also crucial.

Another possibility is to explain a model which is already live. In this context, one may directly assess which explainers may be suitable.

3.4.5 Testing the model

This step includes testing the model on the test set. When splitting the dataset into training-, validation- and test set, stakeholders should ensure that the data parts are still representative.

3.4.6 Explanation of model outputs

In this step, the modeller is choosing the explanation method to be used. If only a whitebox model has been chosen in the latter step, this step can be skipped. There are several methods available for explaining the model results derived (see chapter xAI methods). The approach chosen has to reflect the objectives the stakeholders want to achieve and the data formats which are at hand. Several explainers only support tabular data or images. Orchestrating several explainers to provide more robust explanations may be beneficial. Doshi-Velez and Kim (2017) point out that the opinion of domain experts and practitioners might be useful here, because they can verify at first glance if results should be carried in the next phase. The formatting of the results should be comprehensible for every party involved and may be embellished for other applications.

3.4.7 Robustness checking

If the explanations hold at first glance, one must check them more thoroughly. Robustness in this context means that the explanations provide consistent results. Unfortunately, inconsistent results are possible (Slack et al., 2019) depending on the model explained. Consistency can be verified by double-checking the explanations of different approaches to see if they reach the same conclusions (Senoner et al., 2021). Additional strategies relying on measuring the ability of the explainer to emulate the behavior of the original model (fidelity, (Chi & Liao, 2022)) or using classic metrics like accuracy are possible.

3.4.8 Validation of explanations

If the results are robust, they have to be validated by domain experts. Validity can include the degree of applicability in practice. In the end result, the results should "make sense" to every party involved in the whole process. Because the assumptions made at the deployment of the model forfeit eventually (concept drift), it is important to iterate over the last two steps in a regular interval.

4 Results and Implications

We begin by examining the use case to develop hypotheses from a domain-knowledge perspective. These hypotheses are then tested using selected xAI methods. Additionally, we discuss the advantages and disadvantages of the chosen methods. Based on our findings, we propose an adapted workflow and outline the limitations of our process, offering directions for future research. The workflow stems from the approach by Tchuente et al. (2024) and is inspired by their phases Idea, Data and Context.

4.1 Initial exploration of the use case

To develop a basis for comprehensible and expertise-based hypotheses, an analysis must be carried out in the specific domain. For this, we analyze the

data using an exploratory data analysis (Tukey, 1977). The chosen action of the agent can be considered the dependent variable in our dataset, while we treat the other variables as features.

4.1.1 Most manufactured products

Product 5 was produced the most, followed by product 8, then product 1. Product 7 was produced the least (Fig 3). The most produced product 5 appeared more than five times as often as the second most produced product 8. The dataset is thus imbalanced regarding product 5 (the dependent variable).

Fig. 3 Bar plot of the total amount of products that were produced. Product five has been produced the most. Some products have not been produced at all in the given dataset.

A potential thesis to draw here is that the variables concerning the products that were produced should play a more important role in the xAI methods than the ones concerning those products that were not produced. Variables relating to products that were not produced may still be part of the reasoning for the agent to make a production decision. However, the reason why certain products were produced should be reflected in the importance of the variables concerning these products, due to the domain-specific objectives (e.g., products are produced when there is demand). If an product was produced but none of the variables relating to its demand and buffer seem to play a role, there might be an underlying logical issue.

4.1.2 Last product type

The last product type feature is a dummy encoded variable (0 or 1) indicating if the predecessor is the same product as produced before. For instance, if `last_prod_type_is_prod5` equals 1, the product produced one step before was product 5. It thus encodes the ordering of the products (Fig. 4). Product 5 has

been produced first followed by short periods of products 8 and 7. Product 1 also breaks the production cycle of product 5 at index 50. Product 8 is being produced again a second time at the end.
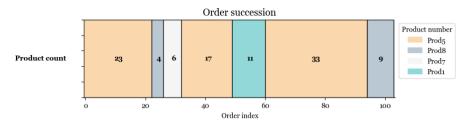

Fig. 4 Order of produced product lots. The x-axis shows the order of the products. Product 5's production has been interrupted three times. 33 lots of product 5 have been produced uninterruptedly. This is the longest unbroken sequence.

Besides avoiding idle times in the FAS, the second objective of the agent is to minimize setup efforts in the PAS. Both have to be balanced out, even though idle times are slightly prioritized. As a consequence, we should see that the agent does not constantly change between production of products (unless that is necessary due to an empty buffer or wrong buffer content), which would lead to sequence depended setup efforts. We see that this is the case for product 1 and product 7; however, the agent produces product 5 and product 8 with breaks in between. Here, the other main objective of the agent – to avoid high criticality and keep a safety margin – should be of relevance. In certain cases, the agent might need to switch between production of products (at the cost of increasing setup efforts), if criticality of another product is too high.

The insight derived from this is that a) because of the objective of minimizing setup efforts `last_prod_type_is_prodX` $= 1$ increases the probability that the same product will be produced again. However, b) if the criticality of another product is high, the agent switches to producing this product and `last_prod_type_is_prodX` $= 1$ of the current product `X` will lower the probability of it being produced again (see Table 2). The objective of avoiding criticality is then of bigger concern than minimizing setup efforts. If both are relevant at the same time, the effects can be contradictory. For example, minimizing setup efforts will lead to production, even when there is low criticality. High criticality in contrast will interfere with minimizing setup efforts.

4.1.3 Buffer fill level

Since the PAS and the FAS are connected via a limited buffer, the buffer acts as a bottleneck that constrains the material flow. The buffer fill level increases over time (Fig. 5). This is intuitive, since the agent causes products to be produced and the PAS has a higher throughput than the FAS. This effect becomes particularly significant later in the planning period.
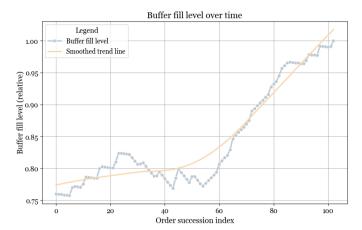
Fig. 5 The buffer fill level over 113 decisions of the agent. After index 50 there is a clear upward trend. The trend line has been fitted with a locally weighted liner regression model (`frac = 0.66`). Note that the y-axis starts at 0.75. Reason: The throughput of the FAS decreases from index 60, so that the PAS fills the buffer with future relevant products. The critical phase in which utilization and setup efforts are balanced are therefore up to around index 60.

4.1.4 Buffer fill level for the specific products

To gain knowledge about specific products, the role of the buffer has to be examined for these product types. The following plot shows the mean buffer content for all products over the whole trajectory (Fig. 6).
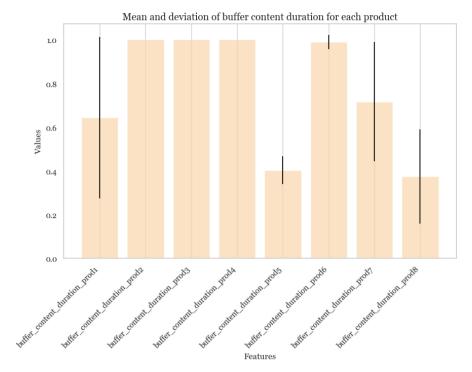
Fig. 6 Mean buffer content duration for products. The vertical lines show standard deviations.

Utilizing this feature, we can interpret the production of these products. The products that were *not* produced (2, 3, 4, 6) had a mainly full mean buffer. Since the buffer satisfies the demand, products with a full buffer were not produced. The product types that were produced had a lower mean buffer and more deviation overall. In fact, product 5 – produced the most – had the lowest mean buffer. Here, it is important to highlight that the agent tries to keep a sufficient level in buffer for all product types to maintain the safety margin and avoid criticality. Therefore, it is plausible to assume that those products with a low buffer fill level are produced more often. Vice versa, if the `buffer_content_duration` of a given product is high, it is more unlikely that this product will be produced. However, this only accounts for criticality. When considering setup efforts the effect can be reversed. The probability of producing an product can increase although the `buffer_content_duration` of this product is already high to minimize setup efforts. Thus, the criticality and setup efforts hypotheses may be of relevance simultaneously and their effects can overlap.

4.1.5 Demand

Demand is encoded in two features, `next_24h_demand` and `end_of_planning_period_demand`. The adjoining table summarizes these:

Table 1 The mean demand for the products. * = produced

Product	next_24h_demand	end_of_planning_period_demand
1*	0.002636	0.068180
2	0.000000	0.000250
3	0.000000	0.000000
4	0.000000	0.000000
5*	0.063485	1.000000
6	0.000000	0.000000
7*	0.013801	0.330981
8*	0.041675	0.857493

It is important to note that both demand types, `next_24h_demand` and `end_of_planning_period_demand`, show the demand left after considering the amount of demand the buffer can satisfy. Therefore, a logical conclusion is that if the buffer for an product increases the demand should decrease. Since the 24h demand is more urgent in a manufacturing environment, those products with a higher `next_24h_demand` should be more likely to be produced. In fact, we see in Table 1 that only those products with a 24h demand were produced at all. Product 5 with the highest mean demand (both types) was produced the most.

The conclusion to be derived from this is that net demand (after deducting the buffer) of an product is positively associated with its production. However, if no product is critical, the agent can continue to produce an product with low demand to minimize setup efforts. Additionally, demand of other products can also influence if an product with high demand is produced. If the demand of another product is higher, a high demand of the product being produced in the instance can make it more unlikely that it is being produced again, because the criticality of another product is higher and the agent has to switch to producing this product.

4.1.6 Criticality

Criticality C is defined as:

$$C = \frac{\texttt{next\_24h\_demand}}{\texttt{buffer\_content\_duration}},$$

where `buffer_content_duration` > 0. Increasing criticality is linked to increasing demand and decreasing buffer content of an product. Criticality is implicitly present in the data, but not explicitly as a feature. We assume that the NN was able to learn this non-linear relationship (Bengio et al., 2013).

4.2 Generating hypotheses

The hypotheses to be derived using domain knowledge (RQ 1.1), descriptive statistics and the objectives of the agent can be subsumed under the two parts of the agent's reward function.

Criticality: High criticality is characterized by high net demand and low buffer content, which is a state the agent tries to avoid, because it increases risk of idle times. Therefore, a high net demand and/or low `buffer_content_duration` of one of the produced products in the data should be positively associated with production in the xAI methods and vice versa. However, if the demand of the *successing* product is higher and/or the buffer is lower, the agent should switch to that product and `last_prod_type` = 1, a higher demand and/or lower buffer of the current product will speak *against* it being produced again. In Table 2, the criticality for product 5 is zero (the demand is zero and the buffer is halfway full (0.522), but the criticality for product 8 is greater than 1 (there is demand and the buffer content is low). Therefore, the agent switches from producing product 5 to product 8 in the second action.

Setup efforts: The agent tries to minimize setup efforts, which is indicated by the `last_prod_type` dummy variable. If an product was produced already (`last_prod_type` = 1), it should increase the probability that the same product will be produced again. This however can lead to a trend that speaks against the hypothesis implied by criticality: The probability of producing an product may increase, although the buffer of this product is high and the demand is low. This can happen, when no other product is critical; therefore, the agent sticks to the production of one product to minimize setup efforts. If we find the reversed trend and `last_prod_type` = 1 decreases the probability that the product is produced again, this should correspond with high criticality – the first objective of the agent – for a different product that is then produced in the next instance.

Table 2 Criticality Analysis

Product	Metric	Value
Product 5	Criticality	0
	24h_demand	0
	buffer_duration	0.522
	last_prod_is_prod_5	1.0
Product 8	**Criticality**	**1.098**
	24h_demand	**0.191**
	buffer_duration	**0.174**
	last_prod_is_prod_5	**0**
Product 8	Criticality	0.875
	24h_demand	0.175
	buffer_duration	0.200
	last_prod_is_prod_8	1.0
Product 8	Criticality	0.716
	24h_demand	0.159
	buffer_duration	0.222
	last_prod_is_prod_8	1.0

Table 3 Setup-efforts Analysis

Product	Metric	Value
Product 5	Criticality	0.887
	24h_demand	0.268
	buffer_duration	0.302
	last_prod_is_prod_5	**1.0**
Product 5	Criticality	0.784
	24h_demand	0.243
	buffer_duration	0.310
	last_prod_is_prod_5	**1.0**
Product 5	Criticality	0.735
	24h_demand	0.228
	buffer_duration	0.310
	last_prod_is_prod_5	**1.0**
Product 5	Criticality	0.732
	24h_demand	0.227
	buffer_duration	0.310
	last_prod_is_prod_5	**1.0**

These two tables illustrate both objectives of the agent with examples from production. In Table 2, we can see that product 8 is critical (in bold), because there is demand for this product, but the buffer content is low. The agent then switches from producing product 5 with zero criticality to production of this product. In Table 3, setup efforts are minimized by sticking to the production of the same product, which is indicated by the last product type always being the same (in bold).

As both tables suggest, the hypotheses need to be tested with the produced products, taking all features into account (RQ 1.1) and embedding the complete production scenario. We now look at the top ten most important variables and products for both xAI methods. Then, we compare both methods regarding their performance and interpretability. We also apply the hypotheses to the products not produced (see Chapter 6.2).

4.3 Applying hypotheses

4.3.1 Product 1

Product 1 was produced the third most (Fig. 4). In the order of production, it predecessor and successor product was product 5 which was the most produced product.

Regarding DeepSHAP, we can see that if the last product produced was already product 1, it is more likely that this product will be produced again (Fig. 18) and vice versa. This aligns with the setup efforts hypothesis. Moreover, a lower buffer content is positively associated with production of the product and vice versa, which matches the criticality hypothesis. If the last product type was product 5, product 1 is more likely to be produced. This aligns with product

1 only being produced following product 5 in the course of production. The other variables have SHAP values around zero.

Regarding Input X Gradient (Table 5), it can be seen that increasing `buffer_content_duration_prod1` mitigates the production of product 1. This can be taken one step further by observing that a higher buffer content for product 5 supports the production of product 1. This is in line with the criticality hypothesis, because product 5 is less critical based on a full buffer. In the result, product 1 is produced more. If the last product which was produced is product 1 as well, it has a positive impact on the renewed production of this product. Here the setup-efforts hypothesis is confirmed and matches the data that shows that product 1 was produced continuously without a machine retrofitting. If the `end_of_planning_period_demand` of product 5 or 8 were increasing, product 1 had slightly less attribution which matches the criticality hypothesis. The demand at the end of the planning period for both product 5 and 8 favoured less production of product 1, but this effect was really low. Both hypotheses categories are thus supported by Input X Gradient for product 1.

4.3.2 Product 5

Product 5 was produced the most (Fig. 4). Its production was interrupted by the products 8 and 1, so the machines had to be retrofitted.

Regarding DeepSHAP, we can see that if product 5 was produced last, it is more likely to be produced again (Fig. 7), which is in line with the setup efforts hypothesis. A full buffer content of product 1 is positively associated with production of product 5, while an empty buffer of product 1 speaks against production of product 5. This pattern supports the criticality hypothesis, because if product 1 becomes critical (e.g., due to a low buffer content), production of product 5 should be less likely and vice versa, which is what we find here. In the order of production, product 8 was produced twice following product 5, but product 5 was not produced after product 8. Therefore, it makes sense that the production of product 5 is more likely when product 8 was not previously produced. However, it should be noted that there are two outliers indicating the opposite direction of effect. Product 1 disrupted production of product 5 once. The pattern of the variable `last_prod_type_is_prod1` is similar to the one of product 8. Most of the time, product 5 was not produced after product 1 and as we can see in the SHAP plot, production of product 5 is associated with the previous product not being product 1. However, there are two outliers again. Fuller buffer content of product 7 speaks for production of product 5 and vice versa. This is again in line with the criticality hypothesis. Next, we see that if the last product produced was product 7, this speaks against product 5 being produced next. Here, we have to take into account that product 5 was produced the most, product 7 was produced the least. There is only one case in the order of production, where product 5 was produced after product 7; however; in all other cases this was not true. Therefore, the pattern aligns with the course of production. The other findings in the SHAP plot, all

support the criticality hypothesis: A low buffer content of product 5 is positively associated with production of product 5. A higher buffer content of product 8 speaks for production of product 5, while a higher 24h-demand of product 8 is negatively associated with production of product 5. Higher end of planning period demand for product 1 also speak against production of product 5.
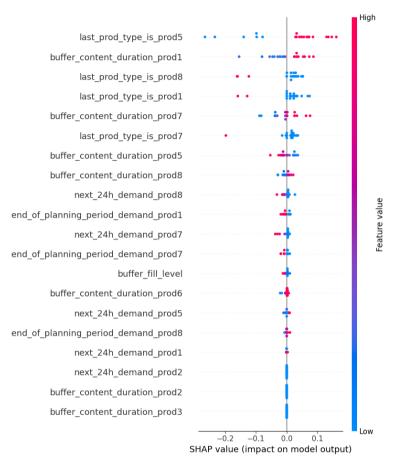

Fig. 7 SHAP summary plot for action 4. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

The following table shows the attribution values we received with the Input X Gradient method:

Table 4 Top 10 Variables for Input X Gradient and Product 5.

Rank	Variable	Value
1	buffer_content_duration_prod5	-0.17
2	buffer_content_duration_prod8	0.12
3	last_prod_type_is_prod5	0.10
4	end_of_planning_period_demand_prod5	0.08
5	buffer_content_duration_prod1	0.07
6	buffer_content_duration_prod6	-0.06
7	next_24h_demand_prod8	-0.05
8	end_of_planning_period_demand_prod8	-0.04
9	buffer_content_duration_prod3	0.03
10	buffer_content_duration_prod4	-0.02

If the buffer content was high for product 5, the probability that it was produced again sunk. In addition, if the buffer for product 8 was high, product 5 was produced more often. A plausible conclusion could be that the buffer content was sufficient to satisfy the demand of product 8, thus alleviating criticality. Unsurprisingly, if the last product was product 5 it was more likely that it was produced again (last_prod_type_is_prod5). The setup efforts hypothesis is thus supported here. The end_of_planning_period_demand_prod5 also had a positive influence. The machine had to be retrofitted two times for products 1 and 8, high buffer content for these products amongst other effects triggers product 5's production as well. A full buffer signals that product 5 can be produced again (lower criticality of products 1 and 8). Also slightly supporting the criticality hypothesis, a higher demand for product 8 in 24h had a negative attribution for product 5.

Both hypotheses are supported thus by Input x Gradient for product 5.

4.3.3 Product 7

Product 7 was produced the least. The predecessor of product 7 was product 8, the successor product 5.

Regarding SHAP (see Fig. 23), we can see that if product 7 was not produced last, production of this product is less likely, which is in line with the setup efforts hypothesis. There is only one individual SHAP point indicating the opposite direction of this hypothesis (i.e., that production of product 7 is positively associated with it being produced again). This makes sense, because descriptively this product was produced only a couple of times, too few cases to create a visual trend in the plot. Fuller buffer content of product 7 speaks against its production, which is in line with the criticality hypothesis; however, the SHAP values for the opposite direction of this hypothesis (i.e. less buffer content is associated with more production) are close to zero. Again, there is only one individual point indicating the latter, which might be tied to product 7 being rarely produced. If it was barely ever produced, it is intuitive that there will be only few cases indicating such a direction of effect. Product 7 was never produced directly after product 5, which aligns with the plot where we can see that if the last product produced was 5, production of product 7 is less likely.

Lower buffer contents of product 8 is negatively associated with production of product 7, in alignment with the criticality hypothesis. However, the SHAP values for the opposite direction of this hypothesis are close to zero. Fuller buffer contents of product 1 and 5 are positively associated with production of product 7, in line with the criticality hypothesis. However, again the SHAP values are close to zero.

Regarding Input x Gradient (Table 10), the attributions were generally low. A higher `buffer_content_duration_prod_8` compliments the production of product 7, whereas a higher product 7 buffer speaks against its production. This supports the criticality aspect (see above). Interestingly, the generic buffer fill level has a negative influence on the production of product 7. This may be because of its rare occurrence. In summary Input x Gradient supports the criticality hypothesis.

4.3.4 Product 8

Product 8 was produced the second most.

Regarding DeepSHAP, we can see that if product 8 was not produced last, it is less likely to be produced again (Fig. 24). This is generally in line with the logic of the setup efforts hypothesis, though it is the reversed pattern (e.g., the product *not* being produced last is associated with it *not* being produced again). Interestingly, for the other direction of effect (e.g., the product being produced last, is associated with it being produced again) there are only few individual SHAP points supporting this trend. This could be related to product 8 batches only being produced thirteen total times, while product 5 was produced more than five times more often. Therefore, it is plausible that the effect is smaller for product 8 than it is for product 5. Previous production of product 5 is negatively associated with production of product 8, which is in line with production order, because product 5 was produced in the biggest batches without interruption and there are only two cases in total where product 8 was produced after product 5. Less buffer content duration of product 8 is positively associated with production of product 8, which is in line with criticality hypothesis. Further, a fuller buffer of product 5 (e.g., criticality is low) speaks for production of product 8. Though the SHAP values are small, this matches the criticality hypothesis. The pattern for buffer content duration of product 1 is ambiguous, but here the SHAP values are close to zero as well A higher end of planning period demand of product 1 speaks for production of product 8, which might seem unintuitive. Here, it is important to differentiate between the two types of demand in the domain. For criticality, the 24h-demand plays the most important role. This means that the end of planning period demand of an product may be high, but if another product is *more* critical in the next 24h it is going to be produced instead. In this particular case, the variable of buffer content duration of product 8 was higher up in the order of variable importance than end of planning period demand of product 1. This might indicate that product 8 was more critical due to an empty buffer, which

is why it was produced; even though, end of planing period demand of product 1 was high. The other variables have SHAP values around zero.

Regarding Input x Gradient (Table 11), it can be derived that a higher buffer of product 8 was not supporting further production of product 8 (supporting the criticality hypothesis). If the buffer for product 5 increased in contrast, this supported the production of product 8. Here, the inverse relationship of the criticality hypothesis is visible: If the buffer for product 8 was high, the production of it sunk. If the buffer for product 5 was high, the production of product increased. This can also be seen in the light of the predecessor/successor relation: Product 8 had been uninterruptedly produced for a longer period of time (product 8 was produced after product 5 two times (four times and nine times respectively). The demand of the end of the planning period of product 8 slightly increased the chances of producing product 8. The criticality hypothesis can thus be confirmed, but not with total confidence. The attribution values were too low. Regarding the setup efforts hypothesis, Input x Gradient does not fully support this hypotheses for product 8.

4.4 Verifying hypotheses

Regarding DeepSHAP, we see that the hypotheses of avoiding criticality and minimizing setup efforts hold true both in the patterns for the variables directly related to the produced products and also for the variables concerning the other products. Input x Gradient supports both hypotheses as well for product 1 and 5. Product 8's attribution values were too low to draw a viable conclusion.

4.4.1 Variables related to the produced products

In DeepSHAP, we can see that if the last product produced was already product 1, product 5, or product 8, it is more likely that these products will be produced again and vice versa, which aligns with minimizing setup times. A lower buffer content of product 1, product 5, or product 8, is positively associated with production of these products respectively and vice versa, which is in line with the criticality hypothesis. Due to it rarely being produced, we can also verify the hypotheses for product 7, but only in one direction: If product 7 was not produced last, production of this product is less likely. Fuller buffer content of product 7 speaks against its production.

Input x Gradient also confirms the setup time hypothesis with the feature `last_prod_type_is_prodX` for product 1 and 5 but not for product 7 and 8. Increasing `buffer_content_duration_prodX` for all four products confirms the criticality aspect as well. A fuller buffer signals lower criticality. The buffer is full and thus it is not critical to produce the product.

4.4.2 Variables for other products

In DeepSHAP, a fuller buffer content of product 1, product 7 or product 8 speak for production of product 5 and vice versa, while a higher demand of

product 8 or product 1 speaks against production of product 5. This is in line with the criticality assumption. If the buffer of these products is low or the demand is high and they become critical, product 5 should be less likely to be produced to avoid idle times. However, if they are not critical, the agent should stick to production of product 5 to minimize setup efforts, which is what we see in the plots for product 5. This pattern holds true for the other classes as well, e.g., a fuller buffer content of product 5 speaks for production of product 8. Lower buffer contents of product 8 is negatively associated with production of product 7.

In the case of Input x Gradient, a full buffer for product 5 supported the production of product 1 (because the buffer of product 5 was high, yielding low criticality for this product). If the demand of product 5 or 8 were increasing, the influence of product 1 decreased, because the criticality of product 5 and 8 increased. The demand at the end of the planning period for both product 5 and 8 favoured less production of product 1, but this effect was low. For product 5, other features also played an important role. A high buffer content for product 8 supported the production of product 5, because the criticality of product 8 sunk. This holds true for the products 1 and 6 as well. Also slightly supporting criticality, a higher demand for product 8 in 24h had a negative influence on the production of product 5. For product 8, fuller buffer content for product 5 supported the production of product 8. Again, because the criticality of product 5 sunk. If the last product was product 5, the probability that product 8 was produced sunk slightly. Note that product 8 was not produced perpetually and was interrupted by product 7. It also was produced after 33 products of 5 were produced, so the machine was retrofitted at the end. For product 7 - the least produced one - we see the criticality hypothesis confirmed because of a higher buffer content of its predecessor product 8. Analogous, a higher buffer for the buffer of 7 itself lowers the probability that it is produced again.

In summary, the overall patterns in the DeepSHAP plots match our hypotheses. Considering Input x Gradient, in the case of product 1 the setup effort hypothesis can be confirmed because there it uninterrupted production. The criticality hypothesis is confirmed as well. For product 5, both hypotheses can be confirmed again. The reduction of retrofitting can be confirmed with `last_prod_type_is_prod5`. Product 8 only slightly favours the criticality hypothesis. There is a negative attribution for the `buffer_content_duration_prod8` which favors to the criticality aspect (compare above). Product 7 favours the criticality hypothesis. Its attribution values were low as well.

4.4.3 Robustness checking

In order to ensure transferability and robustness of our approach, we tested our hypotheses on a different week of production with a different data constellation (only products 4 and 5 were produced). Here, our hypotheses match the DeepSHAP results as well (see 6.1). For example, if product 5 (26) was produced

last, it is more likely to be produced again (setup efforts hypothesis) and if the buffer content of product 5 is lower or the demand is higher, production of product 5 is more likely (criticality hypothesis). Input x Gradient does not support these hypotheses, because the attributions yield contrary results because of only two actions. The buffer content duration for product 7 and the buffer content duration for product 2 give conflicting attribution values for products 4 and 5 (inverse correlations). On the later attributions we see the setup efforts hypothesis confirmed for product 4 (*last_pr_type_is_pr4*) is positive).

4.4.4 Falsification of explanations

We were able to validate the xAI explanations by comparing them to the hypotheses we generated in advance based on domain knowledge. Now, we can present the hypotheses as interpretations of the agent's behavior to the stakeholders, instead of having non-experts in AI inspect dozens of plots and attribution values. The production planners (or management) can then in return give feedback on the plausibility of these explanations.

4.5 Comparing DeepSHAP and Input x Gradient

DeepSHAP matched the hypotheses more often in our use case. Using SHAP values, different effects for higher and lower feature values can be inspected, which is not the case for Input x Gradient and very beneficial to gain a deeper understanding in the decision process of the agent. Also, SHAP offers a range of different visualizations that can facilitate understanding. However, in DeepSHAP there are single outliers and unreasonable data points as previously discussed in 2.2.2 and this method might lead to several hurdles along the way when confronted with variables that are highly dependent. Furthermore, a background dataset has to be chosen; therefore, DeepSHAP is only applicable for use cases that create larger datasets. We conclude that DeepSHAP performs better for classes with many cases (e.g., product 5) and worse for classes with few cases (e.g., product 7). It should also be considered that the concept of Shapley values may be harder to communicate.

Input x Gradient favours the hypotheses less often than DeepSHAP. The interpretation of the explanations is not visibly attractive as in the case of DeepSHAP. Attribution plots can be generated to compare the attribution values for different products and features. It can not be directly compared because the single observations are not visible in the plot like in the SHAP plot. Different gradient attribution approaches are more often used in the context of convolutional NNs. It is harder to use them extensively for tabular data. In the case of the most produced products, attribution values can be computed efficiently and support our findings. In case of the other less produced products like product 7, the attribution values are small (see 6) and can thus not be interpreted meaningfully.

Overall, SHAP was more suitable for the given data (RQ 1.2). Generally, it is desirable to be able to interpret all actions of the agent. For Input x Gradient, this was not possible. The dataset was limited to specific actions and several features showed slight attribution for non-produced products. In a practical setting, this may be hard to communicate. SHAP might be the way to go here, because it may provide better explanations. Another approach may be to gather more data, if possible. This would enhance both methods. SHAP's plotting functionality works out of the box and may assist better in finding interpretations.

4.6 Adapted xAI workflow

Using the results and insights we gained by generating explanations for the DRL agent in our use case based on the workflow of Tchuente et al. (2024), we now adapt said workflow to make it practically feasible for using xAI in the context of production planning (RQ 2.1). We could apply the greyed out phases unfetteredly in this special use case. To apply the whole workflow for explaining DRL agents in a production scheduling context, we made several adjustments.

Fig. 8 Our proposed framework is rooted in Tchuente et al. (2024), but introduces new phases.

4.6.1 Weight Extraction:

The first novel phase consists of extracting the weights of the network and gathering the available data. This is particularly important, because in RL a direct mapping between states and actions is oftentimes not possible and thus the function is approximated, for example by using NNs. By extracting the weights, we have access to the learned policy.

The approach must be adapted depending on the data available:

- If only the raw data of the agent's decision is available and there is no access to model weights or other information regarding the network, it is feasible to train a surrogate model on the data frame. For example, a random forest

classifier can be used to model the agent's behaviour. This classifier can then serve as an input for various xAI frameworks. The information gain here is limited, because the agent used a different model.

- Generating new weights is also a feasible approach. When only a data frame is available and the net dimensions are clear, one can mimic the network structure and create a new network with the same proportions (*fake network*). Then, use inference on the data frame and describe its new weights with the xAI frameworks. The information gain here is better than in the latter approach although still noisy. In the network, the structure is the same but the weights are different. In the context of post-hoc model agnostic explainers, using different weights is not optimal. This is because ideally, we want to explain the models "ideas" during the training phase.

- When the data, weights and network structure (not the network itself) are at hand, these can be used to generate the explanations. In our use case, this was achieved by the export_model function of the Ray framework, which was utilized to train the agent (Moritz et al., 2018). If the network has been implemented in PyTorch (Paszke et al., 2019), extracting the weights is not needed. For our use case, we mimicked the network structure in PyTorch and copied the weights from the Ray model. You may extract certain layers to explain specific structures of the net. After this, we loaded the network with the classical *torch.load* command.

- If the real network (e.g. not a Ray version), the weights and the raw data is available, you may directly apply the xAI methods on the net.

This step ends the data phase.

4.6.2 Hypotheses generation:

After we extracted the weights, we can start by formulating hypotheses about the agent's behaviour. For this, we consider causal presumptions based on domain-knowledge and the reward function, as well as inspecting descriptive statistics. This approach works for xRL as well as for other xAI; for cases outside of DRL, the knowledge of the reward function has to be exchanged with other model specific assumptions or be left out. Hypotheses can be formulated using positive or negative relationships between the variables. It is also possible to construct relationships between hypotheses. Note that if the reward function is already constructed in an interpretable way *before* training (e.g., by reward decomposition), this will simplify this approach. However, for post-hoc explainabilty, influencing stages before the model was employed is not possible.

4.6.3 xAI method selection:

Once the hypotheses stand, we can choose suitable xAI frameworks and methods identified in the information phase. Randomness may arise in the split of training and test data, the explanation method itself or in the training of the network. Setting seeds in this context is a safe way to achieve fixed explanations.

It is desirable to apply frameworks which cover a wide range of xAI methods. Here, one might apply a model-specific and a model-agnostic xAI method or use local and global explainers. For this step, a cooperation with data scientists might be beneficial. We give an overview of recommendations; however, these are not comprehensive and an analysis on the requirements, constrains, and characteristics of the use case must be performed to find the right explainer:

1. Model-specific explainers might be used when debugging is the main goal or when computational costs are high and can be lowered using model-specific information.
2. Model-agnostic methods can be utilized when model-specific methods are constructed too narrowly and cannot be applied to complex real-world scenarios.
3. Local explanations may be incorporated when only single decisions are to be explained.
4. Global explainers should be used when the behavior of the (entire) model is of interest.

To enhance robustness, it is possible to fit the explainer on different configurations of the network and to use cross validation in the same step (Browne, 2000). After capturing the explanation values, it is important to check for outliers in the generated list of attributions. Also, the attributions may be aggregated using the mean or median. A confidence interval may be fitted to further support the findings (Napolitano et al., 2023; Neuhof & Benjamini, 2024). Napolitano et al. (2023) for example develop Interval FastSHAP based on Coalition Interval Games and IntervalSHAP. They show that the prediciton quality can be increased by using more predictors.

4.6.4 Falsification:

Now, we apply the xAI methods and interpret the findings using our hypotheses to verify or falsify the explanations: Do the hypotheses serve as an interpretation of the xAI explanations? If so, the xAI results align with domain knowledge and the hypotheses can be communicated as interpretations of the agent's behavior (green light in Figure 9). If explanations and hypotheses do not match, visit the section validity check.

4.6.5 Validity Check:

The goal here is to inspect inconsistencies between the xAI methods, the hypotheses and the broader context (yellow and red light in Figure 9). Investigate where and to what degree there are deviations between the hypotheses and the xAI explanations, then compare both with domain-knowledge and descriptive statics (e.g., the broader context).

1. Do the xAI results match the broader context? In this case, you may have missed crucial elements in the hypotheses generation. Improve your hypotheses and retest them, preferably on a new data set.
2. Do the xAI results still speak against domain-knowledge? If so, you need to double-check the validity of the chosen xAI methods (assumptions, prerequisites, plausibility for the use case) and possibly apply the hypotheses to other xAI methods. If the mismatch between hypotheses and xAI results persists, you may have detected issues in the underlying AI-model and need to consult developers.

To fully cover for the non-deterministic nature of the explanations, we deliver an extension of the proposed framework which highlights how the Validity Check can be characterized:

Fig. 9 The validity check consists of three scenarios ranging from valid hypothesis to partly valid to not valid.

If the hypotheses align with xAI results, the explanations can be deployed and communicated (Scenario 1; RQ 2.2). The journey is not over here, though. Continuous observation and adaption are crucial. If the hypotheses are partly valid, consider the descriptive statistics and domain-knowledge gathered before. If these support the results of the xAI methods, you may need to reify the hypotheses. If the hypotheses do not align with xAI results and you checked them before as well, it is time to either question the validity of the used

xAI method or revise the AI-model itself. Here, you may need to consult domain-experts and developers.

The proposed framework is generic in the sense that it can be applied in various business scenarios involving xAI. Although, it suffers from several limitations.

4.7 Limitations and future research

As long as AI remains an ever advancing technology, xAI will play an important role. Our proposed framework shows how to approach xRL in a production scheduling setting. However, this approach still involves experts manually constructing and comparing the hypotheses to the explanations.

First of all, the way hypotheses generation is approached needs to be formalized. There is the need to define objective, transparent criteria for the determinants understandability, comparability, communicability, transferability, and validability of hypotheses in any scenario. The question of the "ideal" hypothesis still persists, because this is a highly complex question. Different factors from various scientific disciplines play a role here. Ideality is a subjective question.

Second of all, precise rules to identify significant deviations from the hypotheses are lacking, which might be constructed similarly to hypotheses testing in statistics. This would be needed so that ideally the agent may be able to self-explain.

Thirdly, we were not able to use the original custom class NN with the xAI methods and had to rebuild it, thereby introducing some noise. There were 5 cases, where our rebuild network did not match the predictions of the original one. While we are inspecting overall patterns, there were individual cases where effects were inconsistent with our hypotheses. Additionally, as discussed in the literature review, there were few unrealistic points in the SHAP plots, most likely due to feature dependence.

In the future, it could be promising to create a self-explaining AI using our hypotheses-based approach. An application that adjusts the type of explanation to the needs of the explainee in a conversational or interactive style might be beneficial to increase understanding. Here, it might be fruitful to employ large language models. Additionally, future research should focus on evaluating the explanations and putting the human – the explainee – in the foreground. Our paper focuses on creating explanations and interpretations, but these should also be tested and adapted in terms of effectiveness, user satisfaction, and trust, which requires studies involving humans and a collaboration with social and cognitive science. (M. Y. Kim et al., 2021; Mohseni et al., 2021; Milani et al., 2024)

Also, quantitative metrics to evaluate xRL methods, f.e. regarding their fidelity, are lacking in literature (Milani et al., 2024; Xiong et al., 2024) and were also not assessed in this paper. In the future, these types of metrics should be included.

The framework does not cover details of approaching stakeholders. There may be several blockades to overcome when communicating xAI. These blockades may be gridlocked opinions in the company, a lack of authority or a lack of productive capital to implement suggested changes (social proof, authority, scarcity; c.f. (Cialdini, 2001)). This should also anticipate human biases and heuristics of thought (Tversky & Kahneman, 1974). Humans prefer explanations which intuitively make sense to them and which align with their world view (Gentner & Stevens, 2014). Workflows should reflect on that to create explanations that are transparent and ready to communicate to third parties (Riveiro & Thill, 2021).

Our workflow has not yet been tested in other production and manufacturing settings (e.g. the chemical industry), which could be an opportunity for future research. Also, the framework is data-driven and will be hard to implement when available data is scarce.

Lastly, using our hypotheses approach, we consider the causal understanding in the production planning context. However, the xAI methods themselves are primarily of correlative nature and thus cannot provide evidence for causal effects.

5 Conclusion

In this case study, we investigated the application of state-of-the-art xAI techniques for a DRL-based scheduling model. We built upon an existing agent in a real-world flow production setting, focusing on enhancing the interpretability of the agent's decisions for domain experts. Our comprehensive investigation addressed both method-specific questions (RQ1.1, RQ1.2) as well as organizational aspects (RQ2.1, RQ2.2). On the method side, we utilized two prominent xAI frameworks, SHAP (DeepSHAP) and Captum (Input x Gradient), to analyze the reasoning behind the scheduling decisions. On the organizational side, our proposed xAI approach is based on the workflow of Tchuente et al. (2024), serving as a general procedural model for applying xAI methods in business use cases.

In summary, our findings highlight several critical issues in the current xAI literature, including a lack of falsifiability and consistent terminology, insufficient consideration of domain knowledge, inadequate attention to the target audience or real-world scenarios, and a tendency to offer simple input-output explanations rather than causal interpretations. Moreover, we observed that existing workflows often lack sufficient detail on how xAI methods and their results can be integrated with domain-specific aspects.

To address these challenges, we introduced a hypotheses-based workflow with feedback loops. This workflow allows for the inspection of explanations to ensure they are consistent with domain knowledge and the reward hypotheses of the agent. Our results show that both DeepSHAP and Input x Gradient

are well-suited to explain the behavior of the agent, provided the methods are systematically embedded in the proposed workflow. However, DeepSHAP proved to be slightly more effective in our use case, as it was able to differentiate all the agent's actions more clearly. We hypothesize that this xAI workflow may also be applicable to other DRL-based scheduling models and should be tested and further developed in future studies.

Funding

Author contributions (CRediT)

Daniel Fischer: *Methodology, Formal analysis, Investigation, Software, Writing - original draft*; Hannah M. Hüsener: *Methodology, Formal analysis, Investigation, Software, Writing - original draft*; Felix Grumbach: *Conceptualization, Supervision, Writing – review & editing*; Lukas Vollenkemper: *Conceptualization, Supervision, Writing – review & editing*; Arthur Müller: *Data curation, Software, Writing – review & editing*; Pascal Reusch: *Funding acquisition, Resources*;

Declarations

Conflict of interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.

Open Access

References

Aas, K., Jullum, M., & Løland, A. (2021). Explaining individual predictions when features are dependent: More accurate approximations to shapley values. *Artificial Intelligence*, *298*, 103502.

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018). Sanity checks for saliency maps. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 31). Curran Associates, Inc. Retrieved from https://proceedings.neurips.cc/paper_files/paper/2018/file/294a8ed24b1ad22ec2e7efea049b8737-Paper.pdf

Arras, L., Horn, F., Montavon, G., Müller, K.-R., & Samek, W. (2016). *Explaining predictions of non-linear classifiers in nlp.* Retrieved from https://arxiv.org/abs/1606.07298

Arrieta, A. B., Díaz-Rodríguez, N., Ser, J. D., Bennetot, A., Tabik, S., Barbado, A., ... Herrera, F. (2020, 6). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, *58*, 82–115. doi: 10.1016/j.inffus.2019.12.012

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, *10*(7), e0130140.

Bao, L., Krause, N. M., Calice, M. N., Scheufele, D. A., Wirz, C. D., Brossard, D., ... Xenos, M. A. (2022). Whose ai? how different publics think about ai and its social impacts. *Computers in Human Behavior*, *130*, 107182.

Bekkemoen, Y. (2023, 11). Explainable reinforcement learning (xrl): a systematic literature review and taxonomy. *Machine Learning 2023 113:1*, *113*, 355–441. Retrieved from https://link.springer.com/article/10.1007/s10994-023-06479-7 doi: 10.1007/s10994-023-06479-7

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798-1828. doi: 10.1109/TPAMI.2013.50

Binder, A., Bach, S., Montavon, G., Müller, K.-R., & Samek, W. (2016). Layer-wise relevance propagation for deep neural network architectures. In *Information science and applications (icisa) 2016* (pp. 913–922).

Böhm, G., & Pfister, H.-R. (2015). How people explain their own and others' behavior: a theory of lay causal explanations. *Frontiers in psychology*, *6*, 109763.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Cart. *Classification and regression trees*.

Browne, M. W. (2000). Cross-validation methods. *Journal of Mathematical Psychology*, *44*(1), 108-132. Retrieved from https://www.sciencedirect.com/science/article/pii/S0022249699912798 doi: https://doi.org/10.1006/jmps.1999.1279

Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., ... Gurram, P. (2017). Interpretability of deep learning

models: A survey of results. In *2017 ieee smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, internet of people and smart city innovation (smartworld/scalcom/uic/atc/cbdcom/iop/sci)* (pp. 1–6). doi: 10.1109/uic-atc.2017.8397411

Chatterjee, S., Saad, F., Sarasaen, C., Ghosh, S., Krug, V., Khatun, R., . . . Rose, G. e. a. (2024). Exploration of interpretability techniques for deep covid-19 classification using chest x-ray images. *Journal of imaging*, *10*(2).

Chen, S. (2021). Interpretation of multi-label classification models using shapley values. *CoRR*, *abs/2104.10505*. Retrieved from https://arxiv.org/abs/2104.10505

Chen, T.-C. T. (2023a). Applications of xai to job sequencing and scheduling in manufacturing. In *Explainable artificial intelligence (xai) in manufacturing: Methodology, tools, and applications* (pp. 83–105). Springer.

Chen, T.-C. T. (2023b). Explainable artificial intelligence (xai) in manufacturing. In *Explainable artificial intelligence (xai) in manufacturing: Methodology, tools, and applications* (pp. 1–11). Springer.

Chi, H., & Liao, B. (2022, 4). A quantitative argumentation-based automated explainable decision system for fake news detection on social media. *Knowledge-Based Systems*, *242*, 108378. doi: 10.1016/j.knosys .2022.108378

Cialdini, R. B. (2001). The science of persuasion. *Scientific American*, *284*(2), 76–81.

Clement, T., Kemmerzell, N., Abdelaal, M., & Amberg, M. (2023, 1). Xair: A systematic metareview of explainable ai (xai) aligned to the software development process. *Machine Learning and Knowledge Extraction 2023, Vol. 5, Pages 78-108*, *5*, 78-108. Retrieved from https://www.mdpi.com/2504-4990/5/1/6/htmhttps://www.mdpi.com/2504-4990/5/1/6 doi: 10 .3390/MAKE5010006

Commission, E. (2024). *Ai for public good: Eu-u.s. research alliance in ai for the public good.* Directorate-General for Communications Networks, Content and Technology. Retrieved from https://digital-strategy.ec.europa.eu/en/library/ai-public-good-eu-us-research-alliance-ai-public-good

Das, A., & Rad, P. (2020). *Opportunities and challenges in explainable artificial intelligence (xai): A survey.* Retrieved from https://arxiv.org/abs/2006.11371

Dazeley, R., Vamplew, P., & Cruz, F. (2023, 8). Explainable reinforcement learning for broad-xai: a conceptual framework and survey. *Neural Computing and Applications*, *35*, 16893–16916. (Zeigt nur auf die XAI für RL aussehen soll) doi: 10.1007/s00521-023-08423-1

Dazeley, R., Vamplew, P., Foale, C., Young, C., Aryal, S., & Cruz, F. (2021, 10). Levels of explainable artificial intelligence for human-aligned conversational explanations. *Artificial Intelligence*, *299*, 103525. (MEta review

for XAI) doi: 10.1016/j.artint.2021.103525

Doshi-Velez, F., & Kim, B. (2017, 2). *Towards a rigorous science of interpretable machine learning.* Retrieved from https://arxiv.org/abs/1702.08608v2

Dwivedi, R., Dave, D., Naik, H., Singhal, S., Omer, R., Patel, P., ... Morgan, G. e. a. (2023). Explainable ai (xai): Core ideas, techniques, and solutions. *ACM Computing Surveys*, *55*(9), 1–33.

Dwivedi, Y. K., Hughes, L., Ismagilova, E., Aarts, G., Coombs, C., Crick, T., ... Eirug, A. e. a. (2021). Artificial intelligence (ai): Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy. *International Journal of Information Management*, *57*, 101994.

Eikså, K., Vatne, J. E., & Lekkas, A. M. (2024). Explaining deep reinforcement learning policies with shap, decision trees, and prototypes. In *2024 32nd mediterranean conference on control and automation (med)* (p. 700-705). doi: 10.1109/MED61351.2024.10566218

Fast, E., & Horvitz, E. (2017). Long-term trends in the public perception of artificial intelligence. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 31).

Fernando, Z. T., Singh, J., & Anand, A. (2019). A study on the interpretability of neural retrieval models using deepshap. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval* (pp. 1005–1008).

Gentner, D., & Stevens, A. L. (2014). *Mental models.* Psychology Press.

Gillespie, N., Lockey, S., & Curtis, C. (2021). *Trust in artificial intelligence: A five country study* (Tech. Rep.). The University of Queensland and KPMG Australia. Retrieved from https://doi.org/10.14264/e34bfa3 doi: 10.14264/e34bfa3

Grumbach, F., Badr, N. E. A., Reusch, P., & Trojahn, S. (2023). A memetic algorithm with reinforcement learning for sociotechnical production scheduling. *IEEE Access*, *11*, 68760–68775. Retrieved from http://dx.doi.org/10.1109/ACCESS.2023.3292548 doi: 10.1109/access.2023.3292548

Grumbach, F., Müller, A., Reusch, P., & Trojahn, S. (2022, December). Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning. *Journal of Intelligent Manufacturing*, *35*(2), 667–686. Retrieved from http://dx.doi.org/10.1007/s10845-022-02069-x doi: 10.1007/s10845-022-02069-x

Heuillet, A., Couthouis, F., & Díaz-Rodríguez, N. (2021). Explainability in deep reinforcement learning. *Knowledge-Based Systems*, *214*, 106685.

Jia, Y., Kaul, C., Lawton, T., Murray-Smith, R., & Habli, I. (2021). Prediction of weaning from mechanical ventilation using convolutional neural networks. *Artificial intelligence in medicine*, *117*, 102087.

Kamath, U., & Liu, J. (2021). *Explainable artificial intelligence: an introduction to interpretable machine learning* (Vol. 2). Springer.

Kayhan, B. M., & Yildiz, G. (2021, October). Reinforcement learning

applications to machine scheduling problems: a comprehensive literature review. *Journal of Intelligent Manufacturing*, *34*(3), 905–929. Retrieved from http://dx.doi.org/10.1007/s10845-021-01847-3   doi: 10.1007/s10845-021-01847-3

Keleko, A. T., Kamsu-Foguem, B., Ngouna, R. H., & Tongne, A. (2023). Health condition monitoring of a complex hydraulic system using deep neural network and deepshap explainable xai. *Advances in Engineering Software*, *175*, 103339.

Khadivi, M., Charter, T., Yaghoubi, M., Jalayer, M., Ahang, M., Shojaeinasab, A., & Najjaran, H. (2023). *Deep reinforcement learning for machine scheduling: Methodology, the state-of-the-art, and future directions.*

Kim, J. W., Lee, B. H., Shaw, M. J., Chang, H.-L., & Nelson, M. (2001). Application of decision-tree induction techniques to personalized advertisements on internet storefronts. *International Journal of Electronic Commerce*, *5*(3), 45–62.

Kim, M. Y., Atakishiyev, S., Babiker, H. K. B., Farruque, N., Goebel, R., Zaïane, O. R., ... Chun, P. (2021, 11). A multi-component framework for the analysis and design of explainable artificial intelligence. *Machine Learning and Knowledge Extraction 2021, Vol. 3, Pages 900-921*, *3*, 900–921. Retrieved from https://www.mdpi.com/2504-4990/3/4/45/htmhttps://www.mdpi.com/2504-4990/3/4/45   doi: 10.3390/make3040045

Kindermans, P.-J., Schütt, K., Müller, K.-R., & Dähne, S. (2016). Investigating the influence of noise and distractors on the interpretation of neural networks. *arXiv preprint arXiv:1611.07270*.

Klar, M., Ruediger, P., Schuermann, M., Gören, G. T., Glatt, M., Ravani, B., & Aurich, J. C. (2024, 2). Explainable generative design in manufacturing for reinforcement learning based factory layout planning. *Journal of Manufacturing Systems*, *72*, 74–92. doi: 10.1016/j.jmsy.2023.11.012

Kohlbrenner, M., Bauer, A., Nakajima, S., Binder, A., Samek, W., & Lapuschkin, S. (2020). Towards best practice in explaining neural network decisions with lrp. In *2020 international joint conference on neural networks (ijcnn)* (p. 1-7). doi: 10.1109/IJCNN48605.2020.9206975

Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review*, *39*, 261–283.

Kuhnle, A., May, M. C., Schäfer, L., & Lanza, G. (2022, 10). Explainable reinforcement learning in production control of job shop manufacturing system. *International Journal of Production Research*, *60*, 5812–5834. Retrieved from https://www.tandfonline.com/doi/abs/10.1080/00207543.2021.1972179   doi: 10.1080/00207543.2021.1972179

Langer, M., Oster, D., Speith, T., Hermanns, H., Kästner, L., Schmidt, E., ... Baum, K. (2021, 7). What do we want from explainable artificial intelligence (xai)? – a stakeholder perspective on xai and a conceptual model guiding interdisciplinary xai research. *Artificial Intelligence*, *296*, 103473. doi: 10.1016/j.artint.2021.103473

Leavitt, M. L., & Morcos, A. (2020). Towards falsifiable interpretability

research. *arXiv preprint arXiv:2010.12016*.

Liehner, G. L., Biermann, H., Hick, A., Brauner, P., & Ziefle, M. (2023). Perceptions, attitudes and trust towards artificial intelligence—an assessment of the public opinion. *Artificial Intelligence and Social Computing*, *72*(72).

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in neural information processing systems*, *30*.

Milani, S., Topin, N., Veloso, M., & Fang, F. (2024). Explainable reinforcement learning: A survey and comparative review. *ACM Computing Surveys*, *56*(7), 1–36.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., . . . Ostrovski, G. e. a. (2015). Human-level control through deep reinforcement learning. *nature*, *518*(7540), 529–533.

Mohseni, S., Zarei, N., Ragan, E. D., & Ragan, . E. D. (2021). 24 a multidisciplinary survey and framework for design and evaluation of explainable ai systems. *ACM Transactions on Interactive Intelligent Systems*, *11*. Retrieved from https://doi.org/10.1145/3387166 doi: 10.1145/3387166

Montavon, G., Samek, W., & Müller, K.-R. (2018). Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, *73*, 1-15. Retrieved from https://www.sciencedirect.com/science/article/pii/S1051200417302385 doi: https://doi.org/10.1016/j.dsp.2017.10.011

Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., . . . Stoica, I. (2018). *Ray: A distributed framework for emerging ai applications*.

Müller, A., Grumbach, F., & Kattenstroth, F. (2024). Reinforcement learning for two-stage permutation flow shop scheduling–a real-world application in household appliance production. *IEEE Access*, *12*, 11388–11399. doi: 10.1109/access.2024.3355269

Müller, A., Grumbach, F., & Sabatelli, M. (2024). *Smaller batches, bigger gains? investigating the impact of batch sizes on reinforcement learning based real-world production scheduling.* (Test)

Müller, A., & Sabatelli, M. (2023). *Bridging the reality gap of reinforcement learning based traffic signal control using domain randomization and meta learning*.

Napolitano, D., Vaiani, L., & Cagliero, L. e. a. (2023). Learning confidence intervals for feature importance: A fast shapley-based approach. In *Edbt/icdt workshops*.

Neuhof, B., & Benjamini, Y. (2024, 02–04 May). Confident feature ranking. In S. Dasgupta, S. Mandt, & Y. Li (Eds.), *Proceedings of the 27th international conference on artificial intelligence and statistics* (Vol. 238, pp. 1468–1476). PMLR. Retrieved from https://proceedings.mlr.press/v238/neuhof24a.html

Nyawa, S., Gnekpe, C., & Tchuente, D. (2023, 2). Transparent machine learning models for predicting decisions to undertake energy retrofits in residential buildings. *Annals of Operations Research*, 1–29. Retrieved from https://link.springer.com/article/10.1007/s10479-023-05217-5 doi: 10.1007/s10479-023-05217-5/tables/5

Ozer, C., Guler, A., Cansever, A. T., & Oksuz, I. (2023). Explainable image quality assessment for medical imaging. *arXiv preprint arXiv:2303.14479*.

Palacio, S., Lucieri, A., Munir, M., Ahmed, S., Hees, J., & Dengel, A. (2021). Xai handbook: towards a unified framework for explainable ai. In *Proceedings of the ieee/cvf international conference on computer vision* (pp. 3766–3775).

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . Chintala, S. (2019). *Pytorch: An imperative style, high-performance deep learning library.* Retrieved from https://arxiv.org/abs/1912.01703

Peters, M. A., & Jandrić, P. (2019). Artificial intelligence, human evolution, and the speed of learning. *Artificial Intelligence and Inclusive Education: speculative futures and emerging practices*, 195–206.

Pinedo, M. L. (2012). *Scheduling* (Vol. 29). Springer.

Portoleau, T., Artigues, C., & Guillaume, R. (2024, 1). Robust decision trees for the multi-mode project scheduling problem with a resource investment objective and uncertain activity duration. *European Journal of Operational Research*, *312*, 525-540. doi: 10.1016/J.EJOR.2023.07.035

Puiutta, E., & Veith, E. M. (2020). Explainable reinforcement learning: A survey. In *International cross-domain conference for machine learning and knowledge extraction* (pp. 77–95).

Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, *1*, 81–106.

Rezazadeh, F., Chergui, H., & Mangues-Bafalluy, J. (2023). Explanation-guided deep reinforcement learning for trustworthy 6g ran slicing. In *2023 ieee international conference on communications workshops (icc workshops)* (p. 1026-1031). doi: 10.1109/ICCWorkshops57953.2023.10283684

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 1135–1144).

Riveiro, M., & Thill, S. (2021). "that's (not) the output i expected!" on the role of end user expectations in creating explanations of ai systems. *Artificial Intelligence*, *298*, 103507.

Sado, F., Loo, C. K., Liew, W. S., Kerzel, M., & Wermter, S. (2023). Explainable goal-driven agents and robots-a comprehensive review. *ACM Computing Surveys*, *55*(10), 1–41.

Senoner, J., Netland, T., & Feuerriegel, S. (2021, 12). Using explainable artificial intelligence to improve process quality: Evidence from semiconductor manufacturing. *https://doi.org/10.1287/mnsc.2021.4190*, *68*, 5704–5723. Retrieved from https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2021.4190 doi: 10.1287/mnsc.2021.4190

Serrano-Ruiz, J. C., Mula, J., & Poler, R. (2024). Job shop smart manufacturing scheduling by deep reinforcement learning. *Journal of Industrial Information Integration*, *38*, 100582. Retrieved from https://www.sciencedirect.com/science/article/pii/S2452414X24000268 doi: https://doi.org/10.1016/j.jii.2024.100582

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, *27*(3), 379–423.

Shapley, L. S. (1953). A value for n-person games. In H. W. Kuhn & A. W. Tucker (Eds.), *Contributions to the theory of games, volume ii* (pp. 307–317). Princeton: Princeton University Press.

Shi, M., Savur, C., Watkins, E., Manuvinakurike, R., Mejia, G. G., Beckwith, R., & Raffa, G. (2023). An explainable ai user interface for facilitating collaboration between domain experts and ai researchers. In *xai (late-breaking work, demos, doctoral consortium)* (pp. 112–116).

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning important features through propagating activation differences. In *International conference on machine learning* (pp. 3145–3153).

Slack, D., Hilgard, S., Jia, E., Singh, S., & Lakkaraju, H. (2019, 11). Fooling lime and shap: Adversarial attacks on post hoc explanation methods. *AIES 2020 - Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 180–186. Retrieved from https://arxiv.org/abs/1911.02508v2 doi: 10.1145/3375627.3375830

Sundararajan, M., Taly, A., & Yan, Q. (2016). *Gradients of counterfactuals.*

Sutton, R. S., & Barto, A. G. (2018a). *Reinforcement Learning: An Introduction.* The MIT Press.

Sutton, R. S., & Barto, A. G. (2018b). *Reinforcement learning: An introduction.* MIT press.

Tchuente, D., Lonlac, J., & Kamsu-Foguem, B. (2024, 2). A methodological and theoretical framework for implementing explainable artificial intelligence (xai) in business applications. *Computers in Industry*, *155*, 104044. (Startpaper) doi: 10.1016/j.compind.2023.104044

Tukey, J. W. e. a. (1977). *Exploratory data analysis* (Vol. 2). Springer.

Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases: Biases in judgments reveal some heuristics of thinking under uncertainty. *science*, *185*(4157), 1124–1131.

Vassiliadis, P., Simitsis, A., & Skiadopoulos, S. (2002). Conceptual modeling for etl processes. *ACM International Workshop on Data Warehousing and OLAP (DOLAP)*, 14–21. Retrieved from https://dl.acm.org/doi/10.1145/583890.583893   doi: 10.1145/583890.583893

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*, 279–292.

Wells, L., & Bednarz, T. (2021). Explainable ai and reinforcement learning—a systematic review of current approaches and trends. *Frontiers in artificial intelligence*, *4*, 550030.

Wirth, R., & Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining* (Vol. 1, pp. 29–39).

Wooditch, A., Johnson, N. J., Solymosi, R., Medina Ariza, J., & Langton, S. (2021). The normal distribution and single-sample significance tests. *A Beginner's Guide to Statistics for Criminology and Criminal Justice*

Using R, 155–168.

Wu, M.-C., Lin, S.-Y., & Lin, C.-H. (2006). An effective application of decision tree to stock trading. *Expert Systems with applications*, *31*(2), 270–274.

Xiong, Y., Hu, Z., Huang, Y., Wu, R., Guan, K., Fang, X., ... others (2024). Xrl-bench: A benchmark for evaluating and comparing explainable reinforcement learning techniques. *arXiv preprint arXiv:2402.12685*.

Yang, C.-C., Prasher, S. O., Enright, P., Madramootoo, C., Burgess, M., Goel, P. K., & Callum, I. (2003). Application of decision tree technology for image classification using remote sensing data. *Agricultural Systems*, *76*(3), 1101–1117.

Yang, Y., Tresp, V., Wunderle, M., & Fasching, P. A. (2018). Explaining therapy predictions with layer-wise relevance propagation in neural networks. In *2018 ieee international conference on healthcare informatics (ichi)* (pp. 152–162).

Yuan, H., Liu, M., Kang, L., Miao, C., & Wu, Y. (2022). An empirical study of the effect of background data size on the stability of shapley additive explanations (shap) for deep learning models. *arXiv preprint arXiv:2204.11351*.

Zhang, J. J., Liu, K., Khalid, F., Hanif, M. A., Rehman, S., Theocharides, T., ... Garg, S. (2019). Building robust machine learning systems: Current progress, research challenges, and opportunities. In *Proceedings of the 56th annual design automation conference 2019* (pp. 1–4).
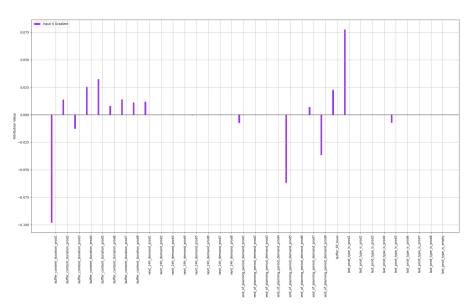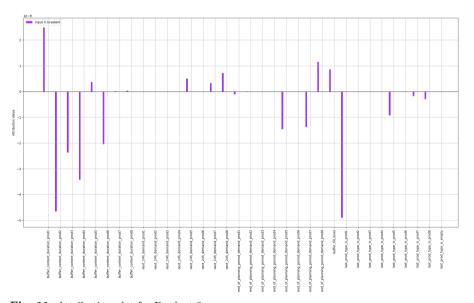
6 Appendix

6.1 Input x Gradient Attribution Plots

Fig. 10 Attribution plot for Product 1
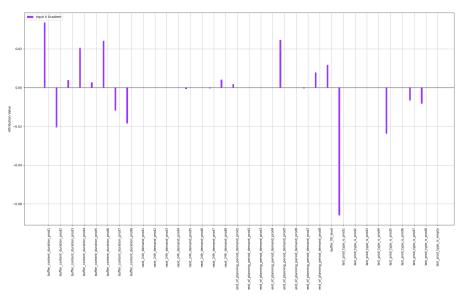

Fig. 11 Attribution plot for Product 2

Fig. 12 Attribution plot for Product 3

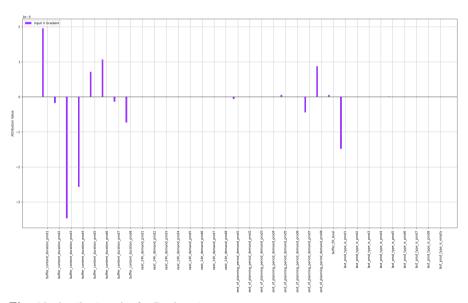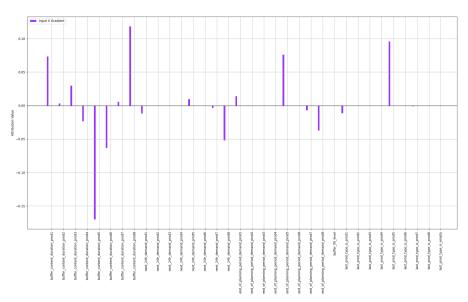**Fig. 13** Attribution plot for Product 4

Fig. 14 Attribution plot for Product 5
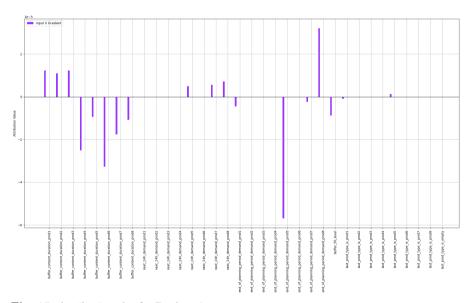

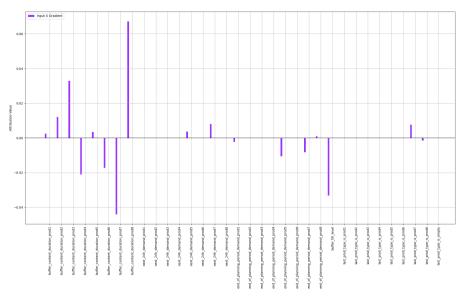**Fig. 15** Attribution plot for Product 6

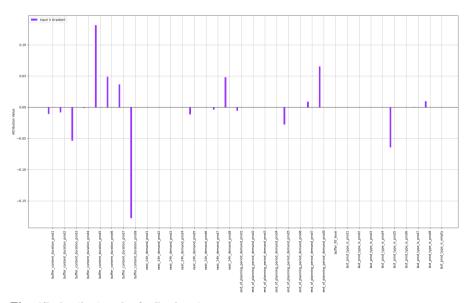**Fig. 16** Attribution plot for Product 7

Fig. 17 Attribution plot for Product 8
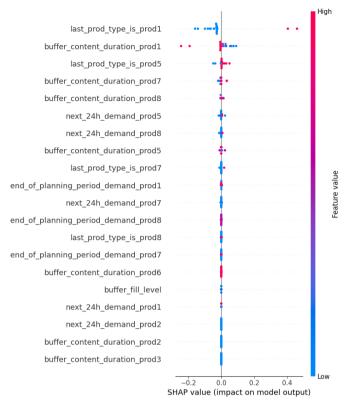
6.2 Product 1

Fig. 18 SHAP summary plot for action 0. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

Table 5 Top 10 variables for Input X Gradient and Product 1.

Ranking	Variable	Value
1	buffer_content_duration_prod1	-0.09835365414619446
2	last_prod_type_is_prod1	0.07755021005868912
3	end_of_planning_period_demand_prod5	-0.06204185262322426
4	end_of_planning_period_demand_prod8	-0.03671699017286301
5	buffer_content_duration_prod5	0.032430436462163925
6	buffer_content_duration_prod4	0.025369716808199883
7	buffer_fill_level	0.022456379607319832
8	buffer_content_duration_prod7	0.013850999064743519
9	buffer_content_duration_prod2	0.01368733774870634
10	buffer_content_duration_prod3	-0.012776759453117847

6.3 Product 6

Product 6 was never produced.

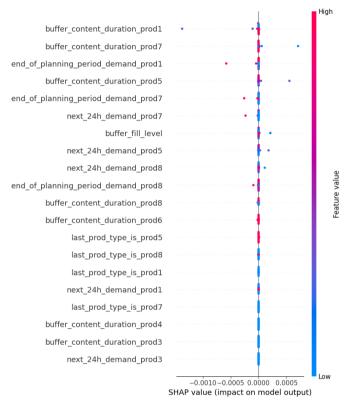Regarding SHAP, we can see that all variables have SHAP values around zero (19).

Fig. 19 SHAP summary plot for action 5. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

These are the attributions for Input x Gradient. As again, the values are too small to interpret them:

Table 6 Top 10 Variables for Input X Gradient and Product 6

Rank	Variable	Value
1	end_of_planning_period_demand_prod5	-5.6928e-05
2	buffer_content_duration_prod6	-3.2762e-05
3	end_of_planning_period_demand_prod8	3.2095e-05
4	buffer_content_duration_prod4	-2.5057e-05
5	buffer_content_duration_prod7	-1.7585e-05
6	buffer_content_duration_prod3	1.2410e-05
7	buffer_content_duration_prod1	1.2367e-05
8	buffer_content_duration_prod2	1.0998e-05
9	buffer_content_duration_prod8	-1.0736e-05
10	buffer_content_duration_prod5	-9.3647e-06

The attribution values are too low to be interpreted.

In summary, for both methods we can support the hypothesis that products that were not produced should play a less important role in xAI methods.

6.4 Product 2

Product 2 was never produced.

Regarding SHAP, SHAP values for most features are close to zero (20). The trend for the most important variable, buffer content duration of prod8 is ambiguous: Higher buffer content of prod8 is plotted with negative, null, and positive SHAP values simultaneously, which cannot be interpreted.

However, there are few individual points indicating that a higher next 24h demand of prod8 is negatively associated with production of prod1, which might be one of the reasons why this prod was not produced. According to our hypothesis, this would mean that production of prod8 was critical; therefore, it was produced instead of prod2.

A fuller buffer content of prod7 is slightly associated with production of prod2. Even though, prod2 was not produced, this is still in line with our hypothesis. If prod7 is not critical (e.g., the buffer is full) that makes it more likely that another product, for example prod2 can be produced. However, the SHAP values are close to zero, which makes sense, because the product was not produced.

Last product type is prod5 is fourth important in this plot. We can see that if the last product produced was not prod5, it speaks against production of prod2. Prod5 was produced the most often; therefore, it makes sense that if prod2 had been produced, the chances would be higher for this to happen following prod5.

We can also see that buffer content of prod6 and production of prod1 is slightly negatively associated with production of prod2. The other SHAP values for the rest of features are close to zero.

In line with our hypothesis that the produced products should play are more important role in the xAI methods than the ones that were not produced, using visual inspection we can tell that asides from few outliers most features have

SHAP values around zero and - besides buffer content duration for prod8 with an ambiguous pattern - no feature is strongly associated with production of prod2. We can see that - again asides from the ambiguous pattern in buffer content duration for prod8 - the few features that show greater absolute SHAP values speak against production of prod2, which makes intuitive sense, because the product was not produced.
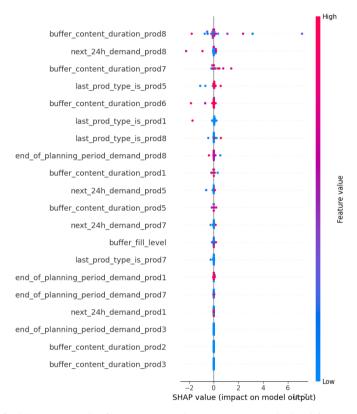

Fig. 20 SHAP summary plot for action 1. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

The following table shows the Input X Gradient attribution values:

Table 7 Top 10 Variables for Input X Gradient and Product 2

Rank	Variable	Value
1	last_prod_type_is_prod1	-4.8995e-08
2	buffer_content_duration_prod2	-4.6587e-08
3	buffer_content_duration_prod4	-3.4221e-08
4	buffer_content_duration_prod1	2.4892e-08
5	buffer_content_duration_prod3	-2.3662e-08
6	buffer_content_duration_prod6	-2.0466e-08
7	end_of_planning_period_demand_prod5	-1.4639e-08
8	end_of_planning_period_demand_prod7	-1.3842e-08
9	end_of_planning_period_demand_prod8	1.1487e-08
10	last_prod_type_is_prod5	-9.3096e-09

The attribution values are too small to interpret.

In summary, for both methods the hypothesis that produced products should play a more important role can be supported.

6.5 Product 3

Product 3 was also never produced.

Regarding SHAP, we can see that asides from few individual points most variables have SHAP values around zero and do not strongly speak for production of prod3 (21).

Fig. 21 SHAP summary plot for action 2. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

Regarding Input x Gradient, we derived the following attributions:

Table 8 Top 10 Variables for Input X Gradient and Product 3

Rank	Variable	Value
1	last_prod_type_is_prod1	-0.06599830090999603
2	buffer_content_duration_prod1	0.03362700715661049
3	end_of_planning_period_demand_prod5	0.02455740049481392
4	buffer_content_duration_prod6	0.02413715235888958
5	last_prod_type_is_prod5	-0.023769624531269073
6	buffer_content_duration_prod2	-0.0206350926309824
7	buffer_content_duration_prod4	0.020487982779741287
8	buffer_content_duration_prod8	-0.018504632636904716
9	buffer_content_duration_prod7	-0.01190112717449665
10	buffer_fill_level	0.011671244166791439

If the product before was product 1, the chances that product 3 will be produced again are decreased.

In summary, for SHAP we can support that hypothesis that products that were not produced should play a less important role in xAI methods.

6.6 Product 4

Product 4 was never produced.

Regarding SHAP, we can see that asides from few individual points most variables have SHAP values around zero and do not strongly speak for production of prod4 (22).
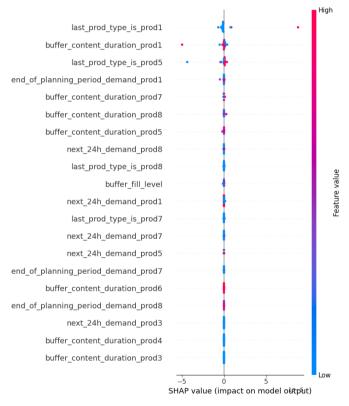

Fig. 22 SHAP summary plot for action 3. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

Regarding Input x Gradient, we derive

Table 9 Top 10 Variables for Input X Gradient and Product 4

Rank	Variable	Value
1	buffer_content_duration_prod3	-3.4675e-05
2	buffer_content_duration_prod4	-2.5701e-05
3	buffer_content_duration_prod1	1.9520e-05
4	last_prod_type_is_prod1	-1.4845e-05
5	buffer_content_duration_prod6	1.0646e-05
6	end_of_planning_period_demand_prod8	8.7112e-06
7	buffer_content_duration_prod8	-7.3491e-06
8	buffer_content_duration_prod5	7.1326e-06
9	end_of_planning_period_demand_prod7	-4.4811e-06
10	buffer_content_duration_prod2	-1.7606e-06

The attribution values are too small to interpret.

In summary, for both methods we can support the that hypothesis products that were not produced should play a less important role in xAI methods.

6.7 Product 7

Fig. 23 SHAP summary plot for action 6. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

Regarding Input x Gradient, these are the top ten attributions:

Table 10 Top 10 Variables for Input X Gradient and Product 7

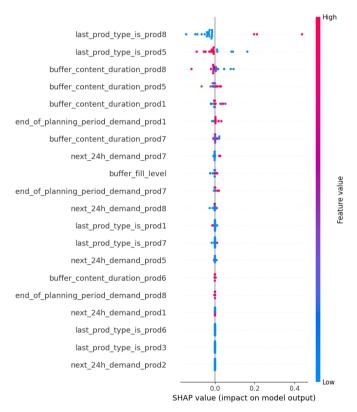| Rank	Variable	Value
1	buffer_content_duration_prod8	0.06717950850725174
2	buffer_content_duration_prod7	-0.0441112294793129
3	buffer_fill_level	-0.033247679471969604
4	buffer_content_duration_prod3	0.032851435244083405
5	buffer_content_duration_prod4	-0.02102786675095558
6	buffer_content_duration_prod6	-0.01731930486857891
7	buffer_content_duration_prod2	0.012098127976059914
8	end_of_planning_period_demand_prod5	-0.010609464719891548
9	end_of_planning_period_demand_prod7	-0.008218199014663696
10	next_24h_demand_prod7	0.008046381175518036

6.8 Product 8

Fig. 24 SHAP summary plot for action 7. Each point represents the local feature attribution value (Shapley value for feature and instance). Blue color indicates a low feature value, for binary variables this is 0, red indicates high feature values, for binary variables this is 1. A positive SHAP value is positively associated with the action, a negative SHAP value is negatively associated with the action. The features are displayed based on importance on average with decreasing importance from top to bottom.

Regarding Input x Gradient, these were the attributions:

Table 11 Top 10 Variables for Input X Gradient and Product 8

Rank	Variable	Value
1	buffer_content_duration_prod8	-0.17791211605072021
2	buffer_content_duration_prod5	0.13158348202705383
3	end_of_planning_period_demand_prod8	0.0651685819029808
4	last_prod_type_is_prod5	-0.06460312008857727
5	buffer_content_duration_prod3	-0.0536046028137207
6	buffer_content_duration_prod6	0.04870598390698433
7	next_24h_demand_prod8	0.04792417585849762
8	buffer_content_duration_prod7	0.0367254838347435
9	end_of_planning_period_demand_prod5	-0.027777016162872314
10	next_24h_demand_prod5	-0.011823729611933231

6.9 Robustness check

Using week 42 from real-world production, we generated new plots using Input x Gradient and SHAP for this new data. Then, we analyzed if our hypotheses still hold true, in order to ensure robustness of our approach.

In week 42, product 4 batches were produced 37 times and product 5 batches were produced 30 times. There were only two cases where our rebuild-network did not match the original one.

Table 12 Top 10 Variables for Input X Gradient prod4

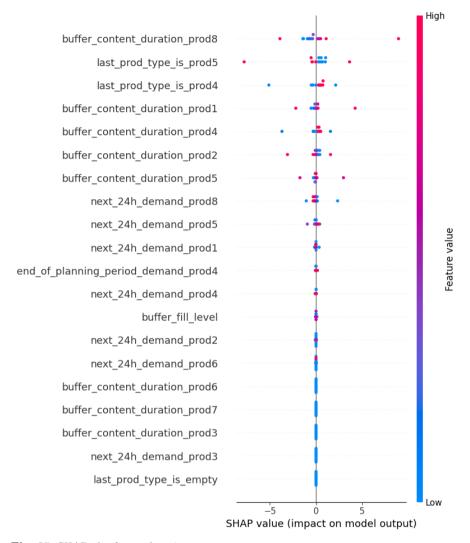| Rank	Variable	Value
1	end_of_planning_period_demand_prod8	-4.196784253451824e-09
2	buffer_content_duration_prod1	-2.7213378217538775e-09
3	buffer_fill_level	2.443531599283233e-09
4	buffer_content_duration_prod3	-1.8710877291994166e-09
5	end_of_planning_period_demand_prod5	-1.3332452919456728e-09
6	buffer_content_duration_prod4	-9.293499303453245e-10
7	buffer_content_duration_prod5	8.536242268597505e-10
8	buffer_content_duration_prod7	7.789259237611645e-10
9	end_of_planning_period_demand_prod7	-3.3234726082298494e-10
10	next_24h_demand_prod5	-2.8125202167217367e-10

Fig. 25 SHAP plot for product 4.

Except one outlier, more buffer content of product 8 makes production of product 4 more likely (criticality-hypothesis). Except one outlier, if product 4 was produced last, it is more likely to be produced again (setup-effort-hypothesis). For buffer content duration of product 4 the trend is ambiguous, but a fuller buffer seems to speak for production. While this goes against our criticality hypothesis, it is inline with minimizing setup efforts; product 4 was produced 37 times, 36 of these without interruption. If no other product was critical (most variables for demand are close to zero), it makes sense that the agent stuck to production of product 4, even though the buffer content was already fuller, in order to minimize setup times.
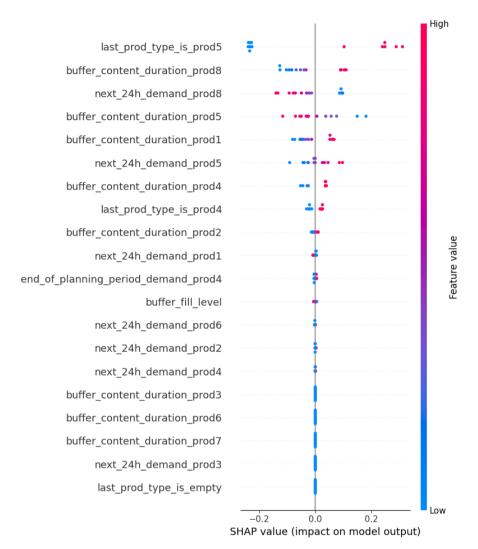
Fig. 26 SHAP plot for product 5

Table 13 Top 10 Variables for Input X Gradient prod5

Rank	Variable	Value
1	end_of_planning_period_demand_prod8	-0.031818896532058716
2	last_prod_type_is_prod4	-0.026379607617855072
3	buffer_fill_level	0.026135722175240517
4	buffer_content_duration_prod8	0.02558363787829876
5	buffer_content_duration_prod7	0.020351236686110497
6	buffer_content_duration_prod5	-0.0160413458943367
7	buffer_content_duration_prod2	0.015320430509746075
8	buffer_content_duration_prod1	-0.015021555125713348
9	next_24h_demand_prod8	-0.012931049801409245
10	end_of_planning_period_demand_prod5	0.011032014153897762

If product 5 was produced last, it is more likely to be produced again (setup-effort-hypothesis). Fuller buffer content of product 8 makes production of prod5 more likely, while high demand of product 8 speaks against production of product 5 (criticality-hypothesis). Similarly, less buffer content and more 24h demand of product 5 speak for its production (criticality-hypothesis).