Bus Inf Syst Eng 66(4):441–463 (2024)
https://doi.org/10.1007/s12599-023-00841-8

RESEARCH PAPER

Self-learning Agents for Recommerce Markets

Jan Groeneveld · Judith Herrmann · Nikkel Mollenhauer · Leonard Dreeßen · Nick Bessin · Johann Schulze Tast · Alexander Kastius · Johannes Huegle · Rainer Schlosser

Received: 9 November 2022 / Accepted: 15 September 2023 / Published online: 27 November 2023

Abstract Nowadays, customers as well as retailers look for increased sustainability. Recommerce markets – which offer the opportunity to trade-in and resell used products – are constantly growing and help to use resources more efficiently. To manage the additional prices for the trade-in and the resale of used product versions challenges retailers as substitution and cannibalization effects have to be taken into account. An unknown customer behavior as well as competition with other merchants regarding both sales and buying back resources further increases the problem's complexity. Reinforcement learning (RL) algorithms offer the potential to deal with such tasks. However, before being applied in practice, self-learning algorithms need to be tested synthetically to examine whether they and which work in different market scenarios. In the paper, the authors evaluate and compare different state-of-the-art RL algorithms within a recommerce market simulation framework. They find that RL agents outperform rule-based benchmark strategies in duopoly and oligopoly scenarios. Further, the authors investigate the competition between RL agents via self-play and study how performance results are affected if more or less information is observable (cf. state components). Using an ablation study, they test the influence of various model parameters and infer managerial insights. Finally, to be able to apply self-learning agents in practice, the authors show how to calibrate synthetic test environments from observable data to be used for effective pre-training.

Accepted after 1 revision by Natalia Kliewer.

J. Groeneveld · J. Herrmann · N. Mollenhauer · L. Dreeßen · N. Bessin · J. S. Tast · A. Kastius · J. Huegle · R. Schlosser (✉)
Hasso Plattner Institute, University of Potsdam, August-Bebel-Str. 88, 14482 Potsdam, Germany
e-mail: rainer.schlosser@hpi.de

J. Groeneveld
e-mail: jan.groeneveld@hpi.de

J. Herrmann
e-mail: judith.herrmann@hpi.de

N. Mollenhauer
e-mail: nikkel.mollenhauer@hpi.de

L. Dreeßen
e-mail: leonard.dreesen@hpi.de

N. Bessin
e-mail: nick.bessin@hpi.de

J. S. Tast
e-mail: johannschulze.tast@hpi.de

A. Kastius
e-mail: alexander.kastius@hpi.de

J. Huegle
e-mail: johannes.huegle@hpi.de

Keywords Recommerce · Dynamic pricing · Competition · Reinforcement learning · Market simulation · Sustainability

1 Introduction

Nowadays, shoppers and retailers alike are becoming more environmentally conscious. A study conducted in 2020 found that over two-thirds of shoppers planned on buying sustainable clothing in the future, and over half already did so regularly, see Statista (2020). At the same time, another study reveals that retailers favor online channels over offline channels when selling used goods, with 78% preferring the former and only 6% the latter, cf. Rabe (2020). This demand is causing more and more businesses, especially those selling their products through e-commerce channels, to adopt more sustainable strategies and enter the circular economy, see, e.g., Kirchherr et al. (2017).

In this context, the need for more sustainable business strategies xed, see Hawlitschek (2021), Weinhardt et al. (2021), as well as the proper use of the potential of artificial intelligence in modern information systems facing challenges of a transforming world, see Thomas et al. (2020).

In the domain of e-commerce, circular markets are also referred to as recommerce markets, a phrase coined in 2005, cf. Colony (2005). Recommerce markets, in which *used* products are sold, are constantly growing. Such markets follow the concept of a circular economy and help to save resources by giving products a longer lifetime. Recommerce firms buy returned articles (such as smartphones, clothes, etc.) from customers or other firms and – after optionally repairing or refurbishing them – resell them again to consumers as used products. As the costs for grading, storing, repairing, or refurbishing products are comparably low and consumers' interest in sustainability is increasing, recommerce is a beneficial business model. However, recommerce firms also face challenges, which can be described as follows:

1. To successfully manage trade-ins and sales a jointly optimized pricing is essential.
2. Further, when also new product versions are sold, substitution effects between new and used products have to be taken into account.
3. Many recommerce markets are characterized by duopoly or oligopoly competition.
4. The interaction of own and competitors' prices on demand are not easy to anticipate.
5. Usually fully manual pricing decisions are not feasible and automation is required, but effective rule-based pricing strategies are hard to derive.
6. Self-learning data-driven algorithms typically require a great amount of data to be trained in practice.

To tackle these challenges, simulated market environments are key to developing, testing, and evaluating the strategic interplay of automated pricing strategies applied in recommerce markets. In addition, the potential performance of self-learning strategies can be compared to rule-based baseline strategies.

In this paper, we propose a conceptual framework for such a recommerce market simulation, including an adjustable customer behavior model, which is capable to apply rule-based and self-learning agents based on reinforcement learning (RL). Monitoring tools shall allow to analyze each agent's policy and their effects on the overall market and the associated resource flows. With the help of such simulations, we seek to study the competitiveness of self-adapting pricing tools and their long-term impact on market competitors and customers.

Our key contributions can be summarized as follows:

- We use a synthetic simulation framework to study whether different state-of-the-art RL algorithms allow computing effective dynamic pricing strategies for recommerce markets in duopoly and oligopoly competition.
- We analyze how different RL algorithms perform compared with rule-based benchmark strategies in different market scenarios and evaluate associated steady states.
- We use self-play to identify strategies that achieve competitive results compared to strategies not seen in training.
- We study the impact of different model parameters and information structures on the performance of RL algorithms and the associated average prices, sales, stock levels, and resource flows.
- We demonstrate how to calibrate synthetic environments from data which allow to pre-train RL agents before applying them to incompletely known environments.
- We provide a rich open-source simulation and evaluation framework, see code repository.[1]

The remainder of this paper is organized as follows. In Sect. 2, we discuss related work. In Sect. 3, we describe our conceptual framework to simulate recommerce markets and to test self-learning algorithms. In Sect. 4, we present several experimental evaluations to study the performance of different state-of-the-art RL algorithms applied within different problem scenarios, including duopoly and oligopoly setups. Further, we perform an ablation study regarding the impact of several model parameters. To illustrate how to potentially apply self-learning agents in practice without being forced to train them extensively on real-life markets, we show in Sect. 5 how to calibrate synthetic test environments from observable data to be used for pre-training before releasing the agent to the target market. In Sect. 6, we summarize the main results obtained and discuss limitations and potential extensions of the model. Concluding remarks and ideas for future work are given in the final Sect. 7.

2 Related Work

We shortly discuss related work along the following different streams associated to the topic of the paper: circular economy (Sect. 2.1), dynamic pricing and market simulations (Sect. 2.2), and an overview of existing RL techniques (Sect. 2.3).

[1] https://github.com/hpi-epic/BP2021.

2.1 Circular Economy

A market is most commonly referred to as a circular economy if it includes the three activities of reduce, reuse, and recycle, e.g., see Kirchherr et al. (2017). This means that while in a classical linear economy market each product is sold once at its new price and after use thrown away, in a circular economy a focus is put on recycling and thereby waste reduction. An entry point to study the concepts of circular economy and sustainable markets is given in, e.g., Stahel (2016) and Bocken et al. (2016). The overall idea is to save resources, reduce the use of resources and to avoid waste, which is also closely related to closed-loop supply chains, see, e.g., Savaskan et al. (2004), Gönsch (2014). Further, related aspects are environmental costs (Commoner 1972) or recycling investments (Schlosser et al. 2021).

2.2 Dynamic Pricing and Market Simulations

Selling products on online marketplaces is a classical revenue management application, see, e.g., Talluri and Van Ryzin (2006). A comprehensive overview of the literature in dynamic pricing research is provided by the surveys by Chen and Chen (2015), den Boer (2015), Strauss et al. (2018), and Klein et al. (2020). The recent work of Gerpott and Berends (2022) particularly discusses dynamic pricing models under competition on online marketplaces.

The combined problem of (i) updating prices, (ii) learning demand behaviors, and (iii) identifying strategy equilibria in competitive markets is challenging as information is incomplete and merchants may constantly adapt their strategies. For analyzing and evaluating the complex interplay and long-term behavior of mutual self-adaptive pricing strategies, market simulations for classical e-commerce applications have been used, see Kephart et al. (2000), DiMicco et al. (2003), van de Geer et al. (2019), and Schlosser and Richly (2019). The latter put a focus on the dynamic interaction of two market participants and their global effects, focusing on each agent's profit as the main performance indicator.

While the previously mentioned research projects focus mostly on online retail markets, many other use cases of dynamic pricing are known. A magnitude of publications considers pricing under competition in very different scenarios, potentially including other constraints, optimization goals, or mechanics which have to be taken into account, ranging from inventory management to fairness goals. Some of those publications rely on reinforcement learning (RL), for example, Maestre et al. (2018). This is one of the examples relying on learning from a simulation, a requirement that is demanded by all model-free RL methods.

RL is usually considered in those use cases in which the state representation becomes too complex to apply otherwise optimal dynamic programming-based solution methods. This can occur in online markets with many competitors or relevant demand features, but in other specialized use cases as well. For example, if the underlying optimization goal does not only aim for revenue optimization, but also for optimal usage of a limited resource, as described in Turan et al. (2020). In this use case, an RL-based policy is charged with the task of managing not only the price of rides but also the supply of electric vehicles in a dynamic market.

The previously mentioned publications vary widely in their choice of the solution method. The one presented here, relying on RL and neural networks, is only rarely chosen. To learn specific market dynamics in dynamic pricing, neural networks are used, for example in Yang et al. (2022). In this example, a recurrent neural network is used to represent the market dynamics in a market with perishable products. The combination of RL and recurrent networks has been shown in the past, but is not applied here. Another project considering dynamic pricing using RL and perishable products is provided by Shihab and Wei (2022). Other examples of specialized use cases incorporating competition and other relevant features include cloud resource pricing, see Chen et al. (2022). Markets in which customers also provide products to the market and thus influence the market prices by adjusting their own sales, are, e.g., small-scale energy markets and smart grid systems. Pricing in such markets has been studied by Tsao et al. (2022).

One example of research that considers retail products and which also incorporates selling used versions of those, is provided in Wen et al. (2022). In this use case, the pricing is not dynamic regarding competition, which distinguishes it from the scenario described here.

However, the additional buyback option of recommerce markets and associated circular resource flows, all while keeping the demand dependent on competitors' choices, have not been studied in the mentioned frameworks.

2.3 Reinforcement Learning

In this section, we give a brief overview of existing RL methods. RL agents are trained through a process of trial-and-error. They iteratively interact with an unknown environment by means of an observable state and an action and observe feedback signals as well as the following state according to an underlying Markov process. Simulation-based RL algorithms enable the heuristically solving of large Markov decision problems with incomplete information.

The books by Sutton and Barto (2018) and Bertsekas (2019) summarize the state of the art in the field of RL. Examples of established RL algorithms are, e.g., Deep Q-learning Networks (DQN), cf. Mnih et al. (2015), Proximal Policy Optimization (PPO), cf. Schulman et al. (2017), or Soft Actor Critic (SAC), cf. Haarnoja et al. (2018).

Applications of DQN and SAC to dynamic pricing under competition can, e.g., be found in Kastius and Schlosser (2022). Unfortunately, RL algorithms typically require a lot of training data which makes them less attractive to use in practice. To overcome this issue, current approaches such as transfer learning, cf. Zhu et al. (2020), or multitask RL, cf. Teh et al. (2017), seek to use observable data more efficiently. Alternatively, synthetic market environments that mimic the use case under consideration can be used to pretrain RL agents.

In this context, we aim at closing a research gap by studying the applicability and effectiveness of different state-of-the-art RL methods to recommerce problems with additional buyback options using a rich open-source simulation and evaluation framework.

3 Model and Problem Description

In this section, we first briefly introduce the main components of our recommerce model (Sect. 3.1) and then describe the mathematical model and its problem formulation as Markov Decision Process from one firm's perspective (Sect. 3.2). In Sect. 3.3, we define a suitable environment to be able to apply RL methods in different market setups. Finally, we discuss different potential classes of RL methods for our recommerce use case and select suitable algorithms to be used in the evaluation and comparison.

3.1 Overview

To mimic real-life recommerce markets, in our market simulation framework, we consider the following main components: (i) supplier(s), (ii) firm(s) including a private data/event store, (iii) a (shared) marketplace, (iv) consumer, (v) resource in use, and (vi) waste (cf. garbage), see Fig. 1. The components are connected as follows. Firms set prices for new and used products on their (or a central) marketplace. Arriving consumers decide (whether and) which product to buy from which firm. Bought products are considered as a resource in use on the consumer side. They may be disposed of as garbage after a while or sold back to one of the firms which offer a corresponding buyback price. Firms can also order new resources from their (or a central) supplier at a certain cost (cf. virgin cost). To be able to easily integrate various RL libraries, we follow a standard stationary discrete-time setup with an infinite horizon.

Note that this basic model sketched above also allows to describe the following special cases: (i) monopoly settings, (ii) scenarios with just one product type, and (iii) classical e-commerce scenarios with no buyback option (cf. linear economy), as well as combinations of those cases (i)-(iii).

Active decisions are made by the firms and the consumers. The pricing decisions of a firm can be organized by a certain rule-based strategy as well as an RL agent exploiting a firm's gathered partially observable market data. Consumer behavior can be arbitrarily defined, e.g., by generating random numbers of interested customers with heterogeneous preferences and a diversified willingness-to-pay. Besides consumers of a myopic type, also certain shares of strategic or loyal customers can be defined and considered.

3.2 Model Description

In the following, we describe the setup, a firm's controls, the consumers' behavior, competitors' reactions, and a firm's objective.

3.2.1 Setup

We consider an infinite time horizon. The discount factor for one unit of time is denoted by δ. We consider K competing firms. Each firm sells a new version as well as used versions of a standardized product. If a customer buys a new product, the item becomes a used product and is added to the number of resources in use denoted by N_{use}. In this context, firms can buy back items from consumers. Each firm has an inventory of used products. Items repurchased from the consumers are added to a firm's inventory. We assume that the rebuy price includes average costs for cleaning, refurbishing, repairing, etc.). The inventory holding costs per item and unit of time are denoted by $c_{inv} \geq 0$. If a consumer buys a new product, the selling firm receives the item from a supplier and faces virgin costs $c_{virgin} \geq 0$ per item.

3.2.2 A Firm's Controls and Competitors' Reactions

Each firm k, $k = 1, ..., K$, sets a price $p_{new}^{(k)} \in A_{new}$, for new products, where A_{new} denotes the set of admissible prices. Further, each firm k, $k = 1, ..., K$, sets a price $p_{used}^{(k)} \in A_{used}$, for used products, where A_{used} denotes the admissible prices. Also, each firm k, $k = 1, ..., K$, sets a rebuy price

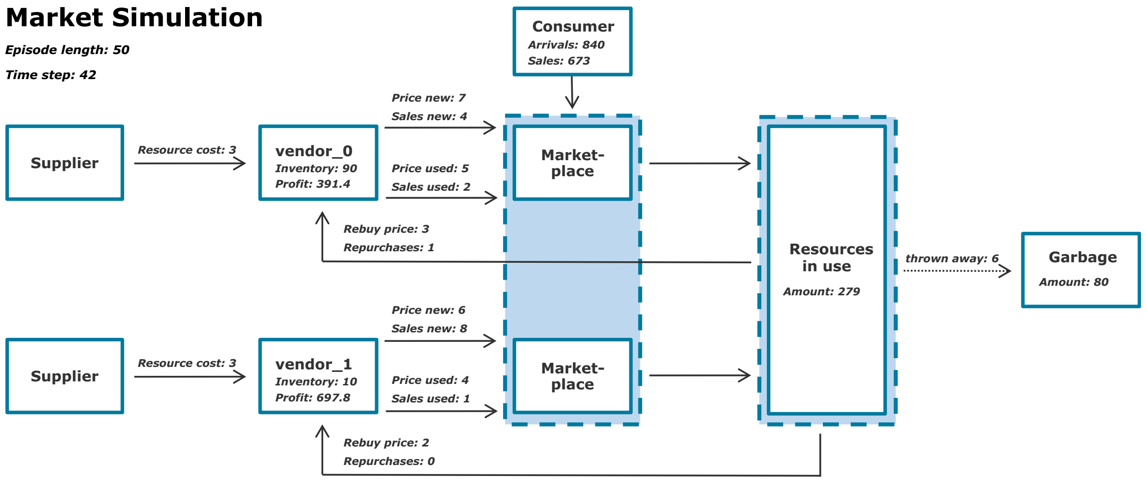


Fig. 1 Illustration of the main components of a recommerce market simulation with two competing merchants and prices for new items, used items, and buyback prices

$p^{(k)}_{rebuy} \in A_{rebuy}$, for rebuying used products, where A_{rebuy} denotes a set of admissible prices. The sets A_{new}, A_{used}, and A_{rebuy} can be chosen discrete or continuous.

A firm sets its three prices simultaneously for one period of time. The K firms set their prices subsequently in a certain order and with fixed delays. Taking the perspective of one specific firm, e.g., firm $k = 1$, this firm sets its prices at the beginning of a period of length one, e.g., from time t to $t+1$, taking into account the current prices $\mathbf{p}_{new}(t) \in A^K_{new}$, $\mathbf{p}_{used}(t) \in A^K_{used}$, $\mathbf{p}_{rebuy}(t) \in A^K_{rebuy}$ as well as the own inventory of used products, cf. $N^{(1)}_{used}(t) \geq 0$. Within the period $(t, t+1)$ each firm $k = 2, ..., K$ of the competing $K-1$ firms adjusts its price vector at its corresponding point in time $\tau^{(k)} \in (t, t+1)$ in a similar way by reacting to the current prices at time $\tau^{(k)}$, i.e., $\mathbf{p}_{new}(\tau^{(k)})$, $\mathbf{p}_{used}(\tau^{(k)})$, $\mathbf{p}_{rebuy}(\tau^{(k)})$ and its own inventory $N^{(k)}_{used}(\tau^{(k)}) \geq 0$. In this context, for each firm, we assume non-anticipating Markovian strategies, where competitors' prices are observable while competitors' inventory levels are not observable. For the firms' strategies, we allow for rule-based as well as self-learning AI-based strategies.

3.2.3 Consumer Behavior for Buying and Reselling

We consider a stream of arriving consumers whose number and timing can be defined in a steady deterministic and in a random fashion.

A single consumer arriving at a certain point in time t observes the current offer prices for new ($\mathbf{p}_{new} := (p^{(1)}_{new}, ..., p^{(K)}_{new})$) and for used ($\mathbf{p}_{used} := (p^{(1)}_{used}, ..., p^{(K)}_{used})$) products from all K firms. A customer's choice behavior can be defined arbitrarily and may include a no-buy option (cf. $k = 0$). In our model, we assume that a (myopic) customer buys at most one product and that the overall consumer buying behavior is expressed as a probability distribution for buying no item at all (cf. $P^{(0)}_{no\ buy}(\mathbf{p}_{new}, \mathbf{p}_{used}) \geq 0$) or buying either a new (cf. $P^{(k)}_{new}(\mathbf{p}_{new}, \mathbf{p}_{used}) \geq 0$) or a used (cf. $P^{(k)}_{used}(\mathbf{p}_{new}, \mathbf{p}_{used}) \geq 0$) product from one of the K firms given the current prices $\mathbf{p}_{new} \in A^K_{new}$ and $\mathbf{p}_{used} \in A^K_{used}$, such that

$$P^{(0)}_{no\ buy}(\mathbf{p}_{new}, \mathbf{p}_{used}) + \sum_{k=1,...,K} P^{(k)}_{new}(\mathbf{p}_{new}, \mathbf{p}_{used}) + \sum_{k=1,...,K} P^{(k)}_{used}(\mathbf{p}_{new}, \mathbf{p}_{used}) = 1. \tag{1}$$

Note, within this framework various choice models can be used.

Moreover, consumer behavior regarding selling back items is modeled as follows. We again consider a stream of consumers (interested in giving back an item), which may explicitly depend on the current number of resources in use N_{use}. The precise number of such customers arriving, e.g., within a period of time, and the timing can be defined in a steady deterministic way as well as in a random fashion. Further, a (myopic) consumer interested in reselling observes the current rebuy prices of all competitors ($\mathbf{p}_{rebuy} := (p^{(1)}_{rebuy}, ..., p^{(K)}_{rebuy})$) and sells back at most one product. We assume that the overall consumer reselling behavior is expressed as a probability distribution for selling no item at all (cf. $P^{(0)}_{no\ sell}(\mathbf{p}_{rebuy}) \geq 0$) or selling the used product to one of the K firms (cf. $P^{(k)}_{sell}(\mathbf{p}_{rebuy}) \geq 0$) given the current rebuy prices $\mathbf{p}_{rebuy} \in A^K_{rebuy}$, such that

$$P^{(0)}_{no\ sell}(\mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) + \sum_{k=1,...,K} P^{(k)}_{sell}(\mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) = 1. \tag{2}$$

Note that the resell probabilities may also depend on the competitors' current offer prices, which may serve as potential reference prices. Again this formulation is fairly general and compatible with established choice models.

3.2.4 *Problem Formulation from a Single Firm's Perspective*

A firm k's profit, $k = 1, ..., K$, is characterized by its sales, purchases, and its holding costs, which are connected to the inventory process $N^{(k)}(t)$. By $X_{new}^{(k)}(t)$ (and $X_{used}^{(k)}(t)$), we denote the number of new items (and used items) sold within period $(t, t+1)$. The number of used items repurchased from consumers within $(t, t+1)$ is denoted by $X_{rebuy}^{(k)}(t)$. Given a pricing strategy (policy) $a^{(k)} := (p_{new}^{(k)}, p_{used}^{(k)}, p_{rebuy}^{(k)})$, a firm k's random accumulated future profits from time t on (discounted on time t) amount to, $t \geq 0$, $k = 1, ..., K$,

$$G_t^{(k)} := \sum_{i=t}^{\infty} \delta^{i-t} \cdot \begin{pmatrix} X_{new}^{(k)}(i) \cdot \left(p_{new}^{(k)}(i) - c_{virgin}\right) + X_{used}^{(k)}(i) \cdot p_{used}^{(k)}(i) \\ -N_{used}^{(k)}(i) \cdot c_{inv} - X_{rebuy}^{(k)}(i) \cdot p_{rebuy}^{(k)}(i) \end{pmatrix}. \tag{3}$$

A firm k's goal, $k = 1, ..., K$, is to determine a non-anticipating (Markovian) 3-dimensional feedback pricing policy that for given a certain initial state

$$s_0^{(k)} := \left(N_{used}^{(k)}(0), \mathbf{p}_{new}(0), \mathbf{p}_{used}(0), \mathbf{p}_{rebuy}(0)\right), \tag{4}$$

which is characterized by a firm k's initial inventory level and the current market prices, optimizes a given objective, e.g., the maximization of expected total discounted profit, cf. (3), from time $t = 0$ on:

$$E\left(G_0^{(k)} | s_0^{(k)}\right). \tag{5}$$

Note, depending on the targeted use case, the number of resources in use might also be considered observable and added to the state. Further, the proposed model is general enough to deal with monopoly or duopoly scenarios as special cases and also subsume models with no differentiation between new and used products or without a rebuy channel (cf. linear economy).

Due to the size of the state space, standard dynamic programming-based solution techniques (assuming complete information about the dynamics of the underlying process) are, in general, not applicable and analytical closed-form solutions are very likely not exist. Hence, we seek to apply RL agents to the problem as an alternative approach.

3.3 Application and Selection of RL Agents

3.3.1 *Embedding of RL Environments*

Based on the model description and problem formulation given in Sect. 3.2, we describe how different RL Algorithms can be applied to our problem by mapping the recommerce model to a standard RL framework. Standard RL frameworks usually require a discrete-time (turn-based) setup and are characterized by a so-called environment, which includes states, actions, reward signals, and state transition dynamics. The RL agent plays against the environment by choosing actions from a certain action space and receiving (aggregated) reward signals and associated state transitions.

From a firm's, i.e., the agent's perspective, the *state* is characterized by its inventory level, the current prices of the competitors, cf. (4), and – if considered observable – the amount of resources in use. Further, a firm's *action* is a combination of prices for new products, used products, and the buyback price. Hence, for an RL agent, the action space is 3-dimensional and given by the price sets A_{new}, A_{used}, and A_{rebuy}.

The *reward signal* of a firm is the aggregated reward associated to realized sales, purchases, and holding costs (within one period); it is characterized by the underlying customer behavior, cf. demand probabilities (1) and the resell probabilities (2), including the defined arrival streams of interested consumers, see Sect. 4.1.2.

Finally, (realized) *state transitions* are organized via the MDP described in Sect. 3.2 and governed by the evolution of the own inventory level and, in particular, the subsequent price adjustments of all competing firms. This, in general, requires that certain, e.g., rule-based, policies are assigned to the competing firms, see Sect. 4.1.1.

The agent's objective is to find a state-dependent strategy that maximizes expected discounted long-term rewards, cf. (5). Note, the agent does not know internals of the environment, i.e., the defined consumer behavior and the defined competitors' strategies are not unknown to the agent. Finally, within the described environment, different standard RL algorithms can be applied by using common RL libraries.

3.3.2 *Selection of RL Algorithms*

Potential state-of-the-art RL algorithms for our problem are so-called Q-Learning-based techniques and policy gradient algorithms. Here, we will focus on RL algorithms using neural networks; note that tabular methods (as used in classical dynamic programming) cannot handle the problem because the size of the state space exceeds the computational limits by far.

Further, as also the action space of our problem will be typically large, we decided to consider algorithms that use a continuous action space. The main reason for this is that for a discrete formulation, neural networks require an output neuron for each individual action. But the size of this action space is $|A_{new}| \cdot |A_{used}| \cdot |A_{rebuy}|$, which does not scale with fine-grained price levels. Moreover, unlike the continuous ones, the actions in a discrete formulation have the disadvantage that they only represent categories and, hence, do not have a metric order. They also do not reflect that the prices are real vectors whose distances have any meaningful interpretation. In line with this, for example, Kastius and Schlosser (2022) found that Soft Actor Critic (SAC), cf. Haarnoja et al. (2018), performed better than Deep-Q-Learning (DQN), cf. Mnih et al. (2015), on their pricing benchmarks.

Finally, as RL methods with continuous action space, we selected the following state-of-the-art RL algorithms to be applied to our problem: A2C Mnih et al. (2016), SAC, and PPO Schulman et al. (2017). Alternative algorithms such as DDPG Silver et al. (2014) or TD3 Fujimoto et al. (2018) were tested but soon ruled out due to their clearly inferior performance. A3C, an asynchronous alternative to A2C, was also considered but deemed unnecessary due to the environment's ability to generate data faster than the algorithm itself could process it.

4 Evaluation

Our evaluation is organized as follows. In Sect. 4.1.1, we define the consumer behavior and rule-based benchmark strategies to be used in our experimental framework, including reproducible hyperparameters for the different RL agents are defined. In Sect. 4.2, we study the performance of RL agents against rule-based benchmark strategies in a duopoly. In Sect. 4.3, we consider an opportunistic version of the setup of Sect. 4.2. In Sect. 4.4, we use self-play to evaluate an RL agent playing against itself in a duopoly. In Sect. 4.5, we provide a study examining alternative versions of observable state spaces. In Sect. 4.6, we also analyze monopoly and oligopoly scenarios. Finally, in Sect. 4.7, we provide an ablation study with respect to various model parameters in order to verify the general applicability of the proposed model framework as well as the robustness of the market results.

4.1 Definitions and Model Specifications

4.1.1 *Competitors' Strategies*

In our experiments, we seek to model the timing of price updates in a realistic (i.e., not concurrently) and fair fashion (i.e., with uniform timely delays). For ease of simplicity, we let the K competing firms subsequently update their prices (in random order) with a fixed consistent delay of $1/K$ units of time. To be precise, firm k, $k = 1, ..., K$, adjusts its prices at points in time $t + (k-1)/K$, $t = 0, 1, ...$ Hence, in each period, we obtain K sub-intervals of length $1/K$ in which the market prices remain unchanged.

As a rule-based benchmark strategy (denoted by RBB), we define a common, representative competitive strategy, which seeks to undercut other competitors' prices but also balances its own inventory level. The strategy uses an incremental price unit h for undercutting competitors' prices and a given upper reference level M for the firm's own inventory level. If firm k plays this RBB strategy, the prices are (at any time t) adjusted as follows:

$$p_{new}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \max\left(\min_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{new}^{(i)}\right\} - h, c_{virgin} + h\right) \quad (6)$$

$$p_{used}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \begin{cases} \min_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{used}^{(i)}\right\} + h & , N_{used}^{(k)} < M/15 \\ \min_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{used}^{(i)}\right\} - h & , N_{used}^{(k)} < M/8 \\ \min_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{used}^{(i)}\right\} - 2h & , else \end{cases} \quad (7)$$

$$p_{rebuy}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \begin{cases} min(c_{virgin} - h, \max_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{rebuy}^{(i)}\right\} + h) & , N_{used}^{(k)} < M/15 \\ \max_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{rebuy}^{(i)}\right\} - h & , N_{used}^{(k)} < M/8 \\ \max_{i \in \{1,...,K\} \setminus \{k\}} \left\{p_{rebuy}^{(i)}\right\} - 2h & , else \end{cases} \quad (8)$$

Note, while the price for new items depends on the unit costs c_{virgin}, the used and repurchase prices depend on the own stock level. If the inventory level is small (high), used prices are chosen slightly higher (lower) compared to the competitors in order to increase (decrease) the inventory level. All prices are restricted to the action space (in case the calculation determines results smaller or larger prices than framed via the sets A_{new}, A_{used}, and A_{rebuy}).

4.1.2 Consumer Arrival and Behavior

For ease of simplicity, in our experiments, we use a deterministic arrival model. In each sub-period of length 1/K associated with the subsequent updates of all competitors, see Sect. 4.1.1, we consider B arrivals of customers with an independent buying behavior sampled from the probabilities $P^{(0)}_{no\ buy}$, $\mathbf{P}_{new}$, and $\mathbf{P}_{used}$, cf. (1).

In our experiments, we define the buying behavior for given prices $\mathbf{p}_{new}$ and $\mathbf{p}_{used}$ as follows. To compare prices of different competitors, we use the following two preference functions $u_{new}(p)$, with $p \in A_{new}$, and $u_{used}(p)$, with $p \in A_{used}$, $\theta_{new} \in [0,1]$, $\theta_{used} \in [0,1]$, $\kappa_{used} \in [0,1]$, defined by

$$u_{new}(p) := \frac{p^{(\max)}_{new}}{p} - e^{p-\theta_{new}\cdot p^{(\max)}_{new}}$$

and

$$u_{used}(p) := \frac{\kappa_{used}\cdot p^{(\max)}_{used}}{p} - e^{p-\theta_{used}\cdot p^{(\max)}_{used}}.$$

Based on the preference functions u_{new} and u_{used} and the fixed preference value of *one* associated to the *no buy* option, we use $\Sigma := e^1 + \sum_{i=1,\ldots,K} e^{u_{new}(p^{(i)}_{new})} + \sum_{j=1,\ldots,K} e^{u_{used}(p^{(j)}_{used})}$ and the softmax function to define $P^{(0)}_{no\ buy}$, $P^{(k)}_{new}$, and $P^{(k)}_{used}$, $k = 1,\ldots,K$, as:

$$P^{(0)}_{no\ buy}(\mathbf{p}_{new},\mathbf{p}_{used}) := e/\Sigma \tag{9}$$

$$P^{(k)}_{new}(\mathbf{p}_{new},\mathbf{p}_{used}) := e^{u_{new}(p^{(k)}_{new})}/\Sigma \tag{10}$$

$$P^{(k)}_{used}(\mathbf{p}_{new},\mathbf{p}_{used}) := e^{u_{used}(p^{(k)}_{used})}/\Sigma. \tag{11}$$

Next, we define the consumers reselling behavior. In each sub-period of length $1/K$ associated with the subsequent updates of all competitors, see Sect. 4.1.1, we consider $\lceil N_{use}\cdot w\rceil$ arrivals of customers interested in reselling, where we use the share $w \in [0,1]$. Each arriving consumer observes the current prices $\mathbf{p}_{rebuy}$ and the reselling behavior is independently sampled from $P^{(0)}_{no\ sell}$ and $P^{(k)}_{sell}$.

In our experiments, for given rebuy prices $\mathbf{p}_{rebuy}$, we define the buying behavior using the reference price $p_{min} := \min_{i\in\{1,\ldots,K\}}\left\{\min(p^{(i)}_{new}, p^{(i)}_{used})\right\}$ as well as the preference functions $u_{rebuy}(a) := 2\cdot e^{\frac{a-p_{\min}}{p_{\min}}}$, $a \in A_{rebuy}$, and $d_{rebuy}(\mathbf{p}_{rebuy}) := 2/\left(\max_{i\in\{1,\ldots,K\}}\{p^{(i)}_{rebuy}\}+1\right)$. Based on the preference functions d_{rebuy} and u_{rebuy} we use $\tilde{\Sigma} := e^{1+d_{rebuy}(\mathbf{p}_{rebuy})} + \sum_{i=1,\ldots,K} e^{u_{rebuy}(p^{(i)}_{rebuy})}$ and the softmax function to define $P^{(0)}_{no\ sell}$ and $P^{(k)}_{sell}$, $k = 1,\ldots,K$, as:

$$P^{(0)}_{no\ sell}(\mathbf{p}_{new},\mathbf{p}_{used},\mathbf{p}_{rebuy}) := e^{1+d_{rebuy}(\mathbf{p}_{rebuy})}/\tilde{\Sigma}. \tag{12}$$

$$P^{(k)}_{sell}(\mathbf{p}_{new},\mathbf{p}_{used},\mathbf{p}_{rebuy}) := e^{u_{rebuy}(p^{(k)}_{rebuy})}/\tilde{\Sigma}. \tag{13}$$

Note, the consumers reselling behavior depends on the different competitors' rebuy prices and the current prices for new and used items as a reference to discard rebuy offers.

4.1.3 Reproducible Example

In the following examples and experiments, if not chosen differently, we use the dynamics defined in Sects. 4.1.1 and 4.1.2 as well as the parameters summarized in Table 1.

4.1.4 Hyperparameters

The absence of theoretical knowledge about the determination of optimal hyperparameters requires exhaustive experimental efforts within various settings for our specific problem. This is not within the scope of this work. Instead, as we look for solutions avoiding tedious tuning, as a reasonable simple choice, we use the default hyperparameters of the original methodological papers and test their suitability for our recommerce problem. These can be found in the Appendix (available online via http://link.springer.com) in the Tables 4, 5, and 6. Nevertheless, it might be possible to identify better hyperparameter setups than the ones used for the following experiments. While no full parameter sweep was performed by us, in some cases the conclusions about the algorithms were validated over several hyperparameter combinations.

4.1.5 Implementation Setup

The comparison of the RL algorithms takes place on the market defined in Sect. 3.2. This market is simulated on the test platform developed by us. The agents are communicated via the Gym interface, see Brockman et al. (2016). For the algorithms to be compared, the implementations of the library *Stable Baselines*, cf. Hill et al. (2018), are used. Stable Baselines is an open-source RL library written in Python. It is built on PyTorch, see Paszke et al. (2019). In most cases, the algorithms implemented in Stable Baselines directly correspond to the proposed algorithms of the original papers and are characterized by high code readability. All hyperparameters are configurable.

4.2 Experiment A: An RL Agent Against the Rule-Based Strategy RBB in a Duopoly

In this experiment, we consider a duopoly against the rule-based benchmark competitor (RBB). We compare the

Table 1 Parameters with brief explanation and default values used for our experiments

Symbol	Explanation	Default value
δ	Discount factor per period	0.99
c_{virgin}	Purchase or production price for new products	3
c_{inv}	Price per stored used product per period (step)	0.1
A	Price sets $A_{new} = A_{used} = A_{rebuy} = A$	[0, 10]
$p^{(max)}$	Maximum price for all three price sets A_{new}, A_{used}, A_{rebuy}	10
B	Number of customers visiting the store per step	20
w	Proportion of owners considering resale per step	0.05
θ_{new}	Parameter for preference function (new items)	0.8
θ_{used}	Parameter for preference function (used items)	0.5
κ_{used}	Parameter for preference function (used items)	0.55
K	Number of competing firms	2
h	Price decrease for the RBB strategy	1
M	Upper reference value for used products in stock	100
E	Number of periods (steps) per episode	500

performance of the three RL algorithms: A2C, PPO, and SAC.

First, we discuss the results for the on-policy algorithms A2C and PPO. Figure 2 shows the learning curves of both algorithms. Each of these experiments was run four times independently for one million steps (two thousand episodes of 500 steps each), cf. Table 1. The learning curves depict the running averages of the episode returns (cf. profits) over a window of 100 episodes. The range between maximum and minimum episode returns of these four runs is colored. The bold line represents their average. In this graph, the loss region (i.e., values below zero) is hidden to allow a more accurate comparison in the upper-performance region. We observe that all agents reach the profit zone. However, with average scores of about 8000 per episode, we find that PPO clearly outperforms A2C. Further, we observe that PPO's learning stability is significantly higher compared to A2C. Figure 3 illustrates a detailed view of a typical PPO training run. The performance develops in a narrow band, and catastrophic forgetting does not occur. Its average performance increases to 8160 by the end and shows a consistently stable trend in return and price selection. Results for single runs of A2C are given in the Appendix, see Fig. 13.

The difference in learning speed and stability between PPO to A2C is not surprising as it exists *by design*. These experiments show that PPO is successful in its intention to increase training stability. This effect is achieved by limiting the stochastic policy change in each training step. On the other hand, this leads to the observed lower learning speed.

One observation in Fig. 3 deserves special attention because it seems paradoxical at first: Although with regard to profit the RL agent outperforms the rule-based agent after some time, it is later outperformed again. This gives the impression that the agent gets worse during training.

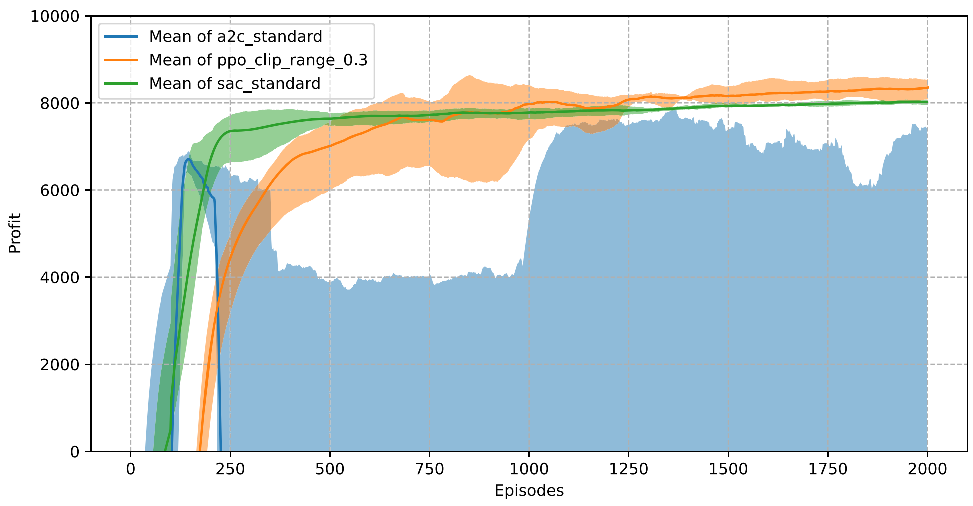


Fig. 2 Learning curves (mean rewards) of A2C, PPO, and SAC on the duopoly with the rule-based undercutting competitor RBB. The shaded areas show the minimum and maximum rewards of 4 runs

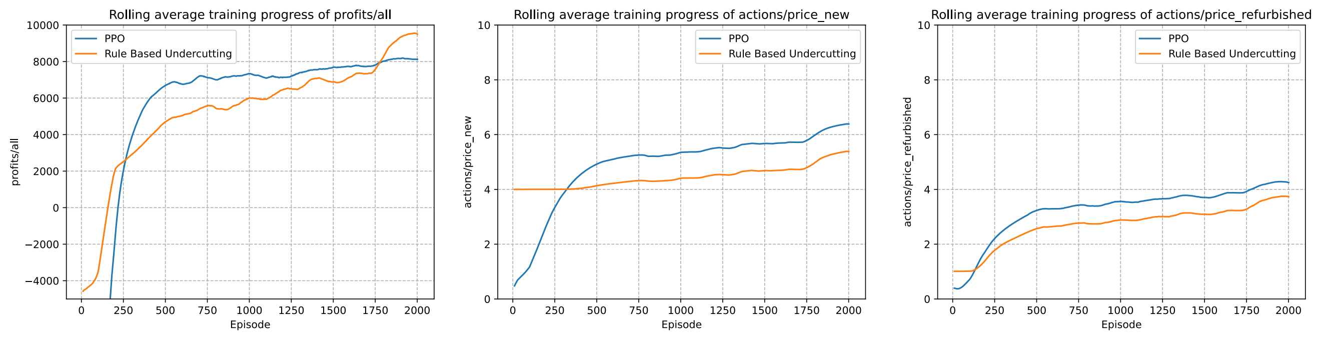


Fig. 3 Detailed view of a PPO training run to visualize stability: (left) learning curve; (center) average selection of new prices; (right) average selection of used prices

However, the opposite is true. The PPO agent first learns to price new and used goods higher. For new sale prices greater than four, it experiences that the rule-based competitor always underbids its price by 1. Through reciprocal undercutting, a downward price spiral leads to the price settling just above the purchase price. However, the extremely low rate of return allows only low profits. While the agent gains experience through exploration, it learns that it can increase its overall profit by moving the market to a higher price band. The competitor then continues to undercut the agent, but only by the value of one. In doing so, the agent accepts that more customers will buy from the competitor due to the low price and that the competitor will also earn more from each customer as prices increase. However, it can still increase its profits compared to the low-priced market. The effect is, therefore, that the RL agent, in reaction to the competitor who is always undercutting him, allows the latter an increase in profits in order to be able to achieve higher profits for himself as well. The existence of this effect is due to the fact that the only optimization criterion for the agent is its own profit. A reward formulation that includes outperforming the competitor as an objective is discussed in Sect. 4.3.

On the same duopoly market, we also evaluated SAC. Figure 2 shows the learning curve of four SAC agents in a training run of 2000 episodes as in the experiments for PPO and A2C. Within the 2000 episodes, SAC achieves slightly less mean rewards compared to PPO.

SAC was developed as an off-policy algorithm for two main goals besides achieving competitive results: First, it is intended to have high sample efficiency, meaning that it requires significantly fewer steps during training. Second, SAC aims at a high training stability. The learning curve confirms that SAC achieves these two main goals. At about 250 to 300 episodes, agents are already close to their peak performance. The rest of the training yields only small performance gains.

Similar to Figs. 3 and 4 shows more details during training. The average prices start – apparently due to a different parameter initialization – at higher levels than for PPO. They then drop and end at 6.4 and 5.4, similar to PPO.

With the implementation and hardware used for these experiments, training per step with SAC takes about 3.7 times as long as with PPO. Collecting the examples from the market takes about 25% of the training time with PPO, and only about 6% with SAC. The speed of training is very similar for A2C and PPO. If we calculate the actual training time, the higher sampling requirement for PPO is put into perspective. A2C is superior to the other algorithms in pure time requirements.

Note, because the SAC training is time-consuming and the learning progress is slow later on, the number of training steps is reduced in some of the subsequent experiments. The same is done with A2C since its maximum performance is achieved very early on.

4.3 Experiment B: An RL Agent against the Rule-Based Strategy RBB in a Duopoly (Opportunistic Version with Adapted Reward Function)

In Experiment A, the RL agents achieved overall good profits but were still outperformed by the rule-based competitor. In Sect. 4.2, we explained that this should not be interpreted as a weakness of the algorithms, but can be attributed to the reward function, which only evaluates its profit. Now, maximizing one's own profit is not an inappropriate metric, yet firms will be reluctant to leave more profit to their competitors than to themselves in a symmetric market. Therefore, it is obvious to evaluate not only one's own profits in the reward function but also whether the competitor is outperformed.

One way is to formulate the market as a *zero-sum game* in that the reward is precisely the difference between one's own profits and those of the competitor. This formulation puts a clear focus on outperforming the competitor and further potentially opens up the market scenario to insights from game theory for zero-sum games. However, this

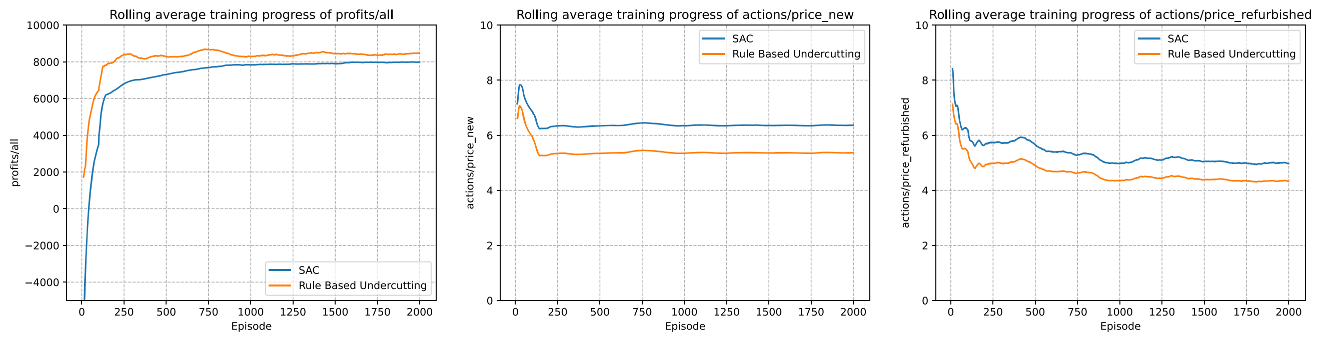


Fig. 4 Detailed view of a SAC training run analogous to Fig. 3

formulation is not practical because the goal of maximizing profit is not valued at all. Thus, optimizing solely the difference may result in significantly outperforming the competitor, but with overall low profits.

Therefore, a mixed reward function was used for the following series of experiments. It simply calculates the sum of (i) profit and (ii) the difference to the competitor's profit. For these experiments, both summands were weighted equally, but a hyperparameter could be inserted to balance the two goals of maximizing profit and outperforming the competitor.

The results, see Fig. 5, show that the competitor can be clearly outperformed, but naturally, the agents' profits are lower than compared to the original implementation, but this difference turns out to vary in its strength among the algorithms.

4.4 Experiment C: RL vs. RL Training via Self-Play

In Experiment A and B, the RL agents all trained against the undercutting rule-based competitor RBB. This satisfies the theoretical requirements of an MDP with fixed and known dynamics but poses a number of problems for practical applications. First, the competitor's policy must be known for this to work. However, because it can be assumed that the competitor will not reveal its pricing strategy, it would have to be estimated from historical data, accepting inaccuracies. Second, the strategy trained using RL is only reliable against that particular rule-based strategy. If the competitor suddenly changes its pricing strategy, this weakens the performance of the RL strategy and necessitates further expensive training.

Therefore, the user wants a policy that can hold up against various different competitor strategies rather than being overfitted to a specific one. DeepMind had a similar challenge in training go and chess strategies, which was solved by self-play, see, e.g., Silver et al. (2017).

For our market, we developed a self-play variant in which an RL agent continues to train on a duopoly market, but the competitor's policy is its own policy. In the programming implementation, a pointer to the RL agent is passed to the opponent, which eventually also implements the policy function. Naturally, as the agent is playing against itself and continuously updates its strategy, the Markov property is violated. Yet, the training still leads to relatively stable rewards in most cases. The motivation behind self-play is that the agent constantly develops strategies against its own, which are then, in turn, challenged. This is to prepare the agent for a variety of opponents' policies.

Our experiments with self-play were again conducted with the algorithms A2C, PPO, and SAC. In this context, the entire series of experiments was performed with the

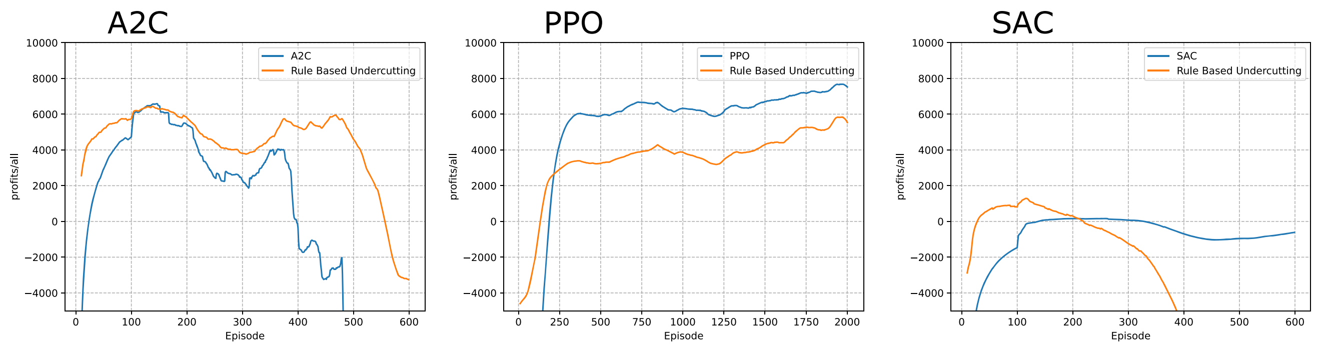


Fig. 5 Representative single runs of A2C, PPO, and SAC with opportunistic reward function

normal reward function and the mixed reward function from Experiment B, cf. Sect. 4.3.

Figure 6 shows the learning curves of the three algorithms when trained against themselves. The learning curves under mixed reward look similar and can be found in the Appendix, see Fig. 14. These curves initially show that learning success is achieved for all of these three agents. However, they alone cannot tell us whether the agents can actually compete against an opponent with an arbitrary pricing strategy. Therefore, the returns of these learning curves should not be directly compared to those from the previous sections. To establish comparability, a model was saved every 50 episodes during self-play and each of these models was subsequently tested for 25 episodes. This created accurate learning curves comparing the model trained during self-play to the rule-based agent RBB. A mean run was selected for the three algorithms and the two reward functions and illustrated in Fig. 7.

For both reward functions, the agents have successfully learned to cope with the market and show that they are also successful against the rule-based benchmark competitor. Note, the peak rewards of all agents in the evaluation are at 8000 or just below. A2C suffers from strong fluctuations, some of the A2C agents do not reach a level of 5000, see Fig. 6.

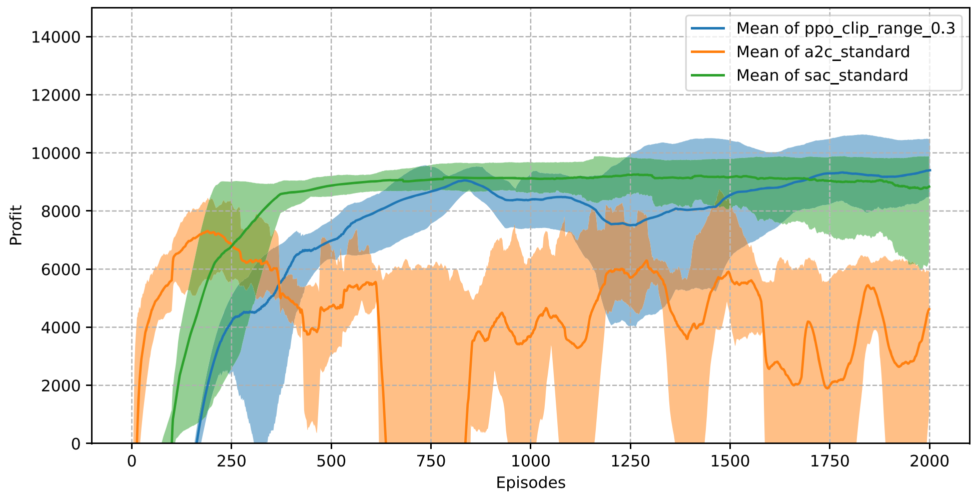
The effects of the mixed reward function can also be seen in the training runs. During training, the difference between the agent's and the opponent's profit is also tracked as an optimization criterion. When evaluated against RBB, the agent trained by self-play via PPO and SAC beats the competitor. While the benchmark results do not achieve the same results as by training directly against the rule-based agent, we observe that they are close. This is noteworthy because these successes were achieved without ever having observed the rule-based competitor before. Thus, these experiments can be considered successful, particularly for PPO and SAC.

4.5 Experiment D: Study for Different Observable State Spaces

In this section, we study cases in which different elements of the state are hidden from the agent. In our model, the fully observable state space contains: (i) current prices, (ii) the number of resources in use, and (iii) all competitors' inventory levels. However, observing some of these quantities is often *not* feasible in practice. For example, the competitor will not let competitors look into his or her warehouse and the number of products in circulation is not directly observable and not easy to estimate. Therefore, the question arises whether the algorithms work without these two pieces of information.

Figure 8a–c shows the learning curves of the three algorithms when trained against the undercutting rule-based strategy RBB under different observable state spaces. The result here is surprising. The expectation that less information would lead to worse results is not fulfilled. For PPO, except for a slight deterioration in stability, performance is similar for the three scenarios (cf. blue, orange, and green plots). This slight drop in stability can probably be explained directly by the lack of information. For A2C, performance improves without additional information, and also for SAC, mean rewards improve significantly. The SAC agents that are only deprived of the competitor's stock level perform quite similarly (slightly better) than those with complete information; the agents that additionally lack the information about the number of products in circulation perform significantly better. Not only does it lower the time to peak performance to below 100 episodes, but it actually improves the peak performance. This brings the maximum closer to that of PPO.

Fig. 6 RL vs. RL in a symmetric duopoly: Learning curves for A2C, PPO, and SAC at self-play; algorithms were trained for 2000 episodes (with normal reward function, 4 runs each)



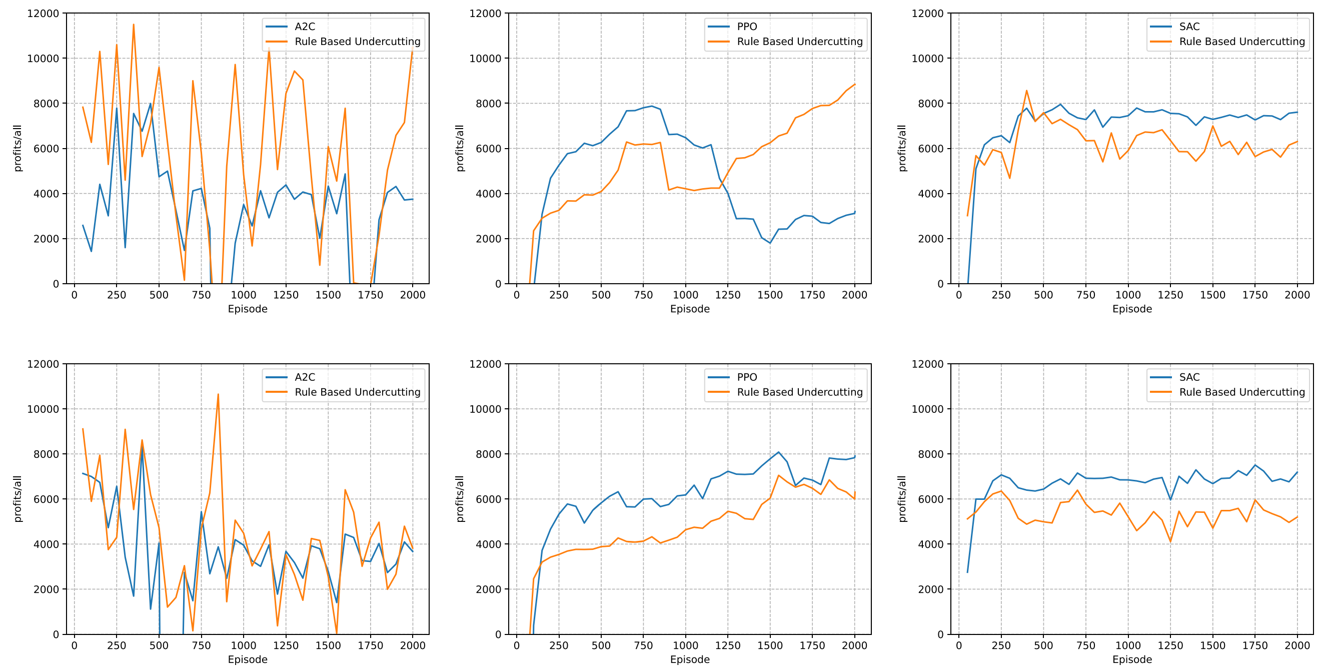


Fig. 7 Performance of representative runs for RL agents trained via self-play for x episodes and then evaluated against RBB ($x = 0, 50, \ldots, 2000$); column-wise by agent: (left) A2C, (center) PPO, (right) SAC; top row with normal reward, bottom row with mixed reward

This is an important finding (cf. practical applicability), and yet, it raises the question of how to explain unexpectedly good performance under incomplete information. The first explanation is that the omitted information plays only a subordinate role. They do explain to some extent the future action of the competitor and the number of owners willing to sell, but the effects are so indirect that they are difficult to exploit for improving a policy.[2] Further, a higher-dimensional observation space also poses a challenge in principle for machine learning methods. The need for samples increases and patterns in the data are harder to detect.

However, the higher dimension is not sufficient to explain the specific phenomenon in SAC. Another theory might serve as the underlying reason here. Figure 9 (middle) displays that an exemplary SAC agent has a strong sensitivity with regard to the two arguments, number of resources in circulation and the stock level of the competitor. Thus, the mean of the SAC policy for used prices differs between 2 and 10 for similar situations, a jump through the entire action space. That a near-optimal policy should behave this way can be ruled out given the low importance of the two arguments, confirming the policy of the clearly more successful PPO agent. As one would expect, the policy of the PPO agent hardly depends on these two arguments. The lower performance maxima of SAC can be explained by this weakness.

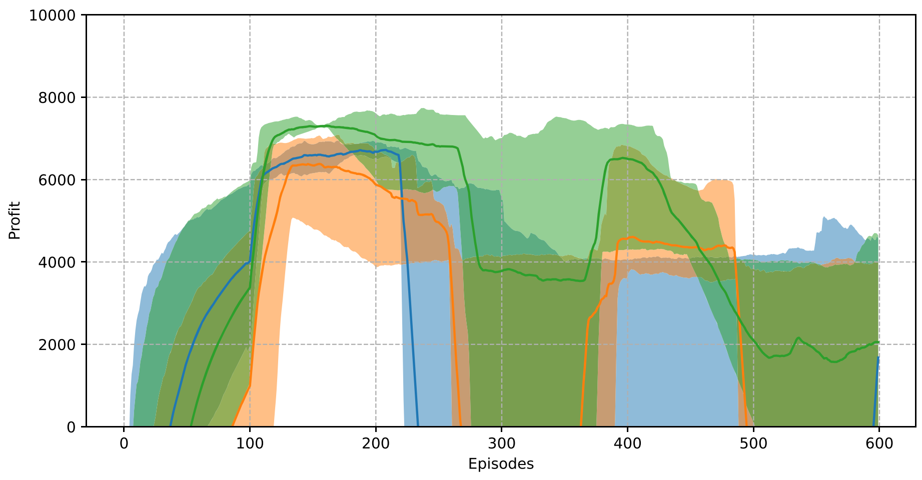
[2] The rule-based competitors do not use this information either.

An explanation can be found in the internal replay buffer Soft Actor-Critic relies on. It is based on the fact that more products are in circulation when policies are evolved in the market. This is shown in Fig. 9 (left) and is due to the fact that little-trained policies often buy back too much (and spend too much money in the process) in contrast to the more evolved ones. This means that, especially in early episodes, the experience buffer is filled only with state transitions with low *in-circulation* values. These samples remain in the experience-buffer, but are no longer useful for later training and can explain the anomalies in the policy. For example, a strong anomaly in the policy is in the range between 50 and 150 products in circulation, exactly the range from which the early samples come. Thus, omitting the *in-circulation* entry arguably improves the performance of SAC because it then no longer has the opportunity to overfit early to relationships that are not sampled from the market reliably in the long term. It is thus noted as a weakness of SAC compared to the on-policy method PPO that it is less able to ignore this bias in the early generated data.

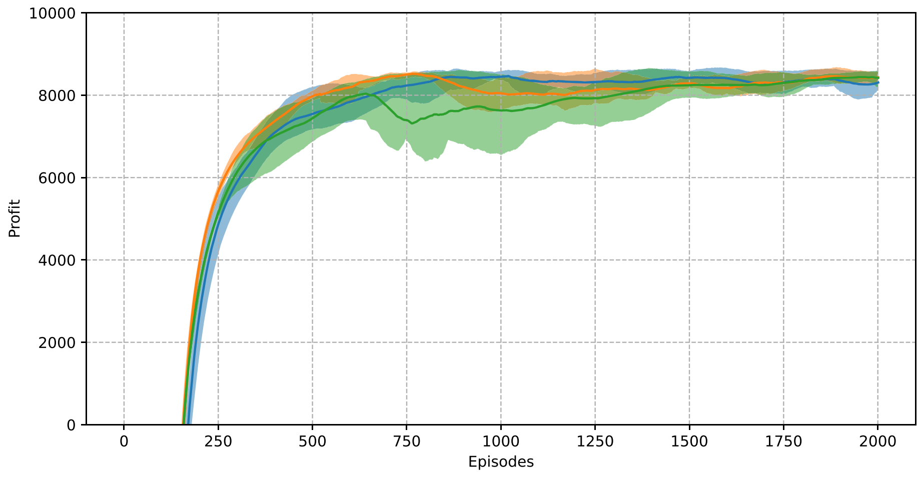
4.6 Experiment E: Monopoly and Oligopoly Scenarios

In the previous experiments A-D, the algorithms were tested in duopoly scenarios. Naturally, in real-world applications, there are also different scenarios to master. In Experiment E, we investigate how the RL algorithms perform on market scenarios such as monopoly setups (cf.

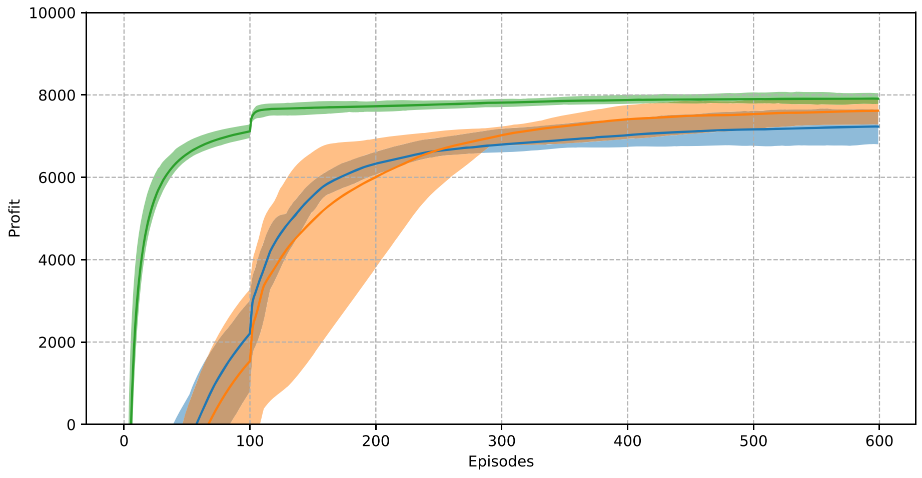
Fig. 8 Learning curves of A2C, PPO, and SAC with full versus partial observation; (blue) full observation, (orange) without competitor's stock level, and (green) without competitor's stock level and the number of products (respective minimum, maximum, and mean results are based on 4 runs) (color figure online)



(a) A2C



(b) PPO



(c) SAC

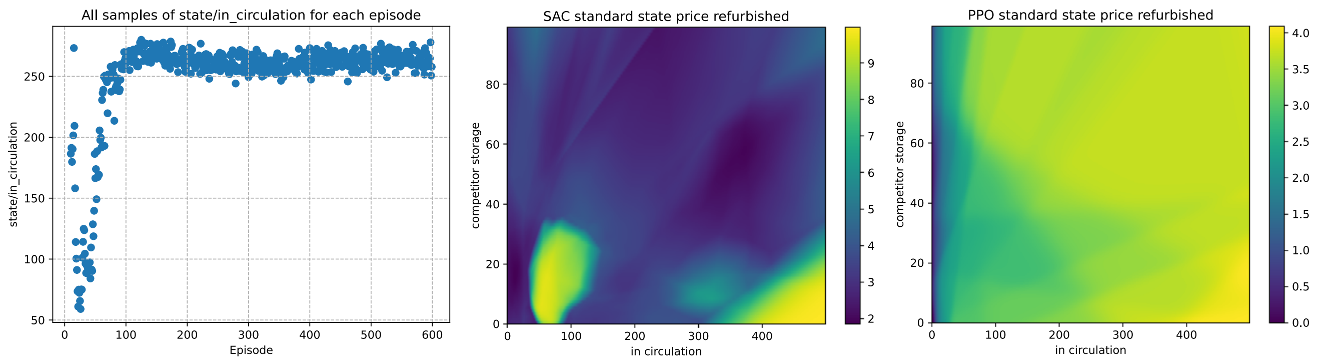


Fig. 9 View of agents with complete information: (left) progression of *in-circulation* counter in SAC training, (center and right) used price policy as a function of *in-circulation* and competitor's stock level for SAC and PPO; for the illustration, other state arguments were pinned to a representative value that is typical in a training run (own stock level: 25, competitor's new price: 6, competitor's used price: 4, competitor's repurchase price: 0)

Sect. 4.6.1) and, in particular, oligopoly scenarios (cf. Sect. 4.6.2).

4.6.1 *Monopoly Scenario*

Compared to duopolies, a monopoly scenario is likely to be less complex. Here, the RL agent is the only merchant, while the same stream of customers continues to visit the market. The other dynamics of the market are unchanged. The agent's reward function optimizes expected profits – comparisons with competitors are unnecessary.

Figure 10 shows the learning curves of the three agents in the described monopoly market with complete information. The parallels to the duopoly results are immediately striking:

1. The relative order of the agents in the learning curve is the same: PPO performs best, then SAC ahead of A2C.
2. A2C is the first to reach the profit zone, followed by SAC and finally PPO.
3. The training of PPO and SAC is stable, A2C's training is characterized by catastrophic forgetting.

We observe that, even though the monopoly market is simpler (cf. the size of the state space), the demand for samples does not decrease noticeably. Comparing the point at which the algorithms stop achieving significantly better results, A2C reaches this at around 150 episodes in both setups, SAC at around 300 episodes, and PPO at around 600 episodes, cp. Figure 2. In the monopoly, the algorithms achieve about twice as high profits as in the duopoly. This is plausible as they no longer have to share the market. In the Appendix, see Fig. 17, a PPO run in a monopoly is compared with one in a duopoly in terms of sales figures.

4.6.2 *Oligopoly Scenario*

Next, oligopoly scenarios are examined. In our experiment, we now consider $K = 5$ players. Here, the RL agent competes against the following set of four rule-based agents:

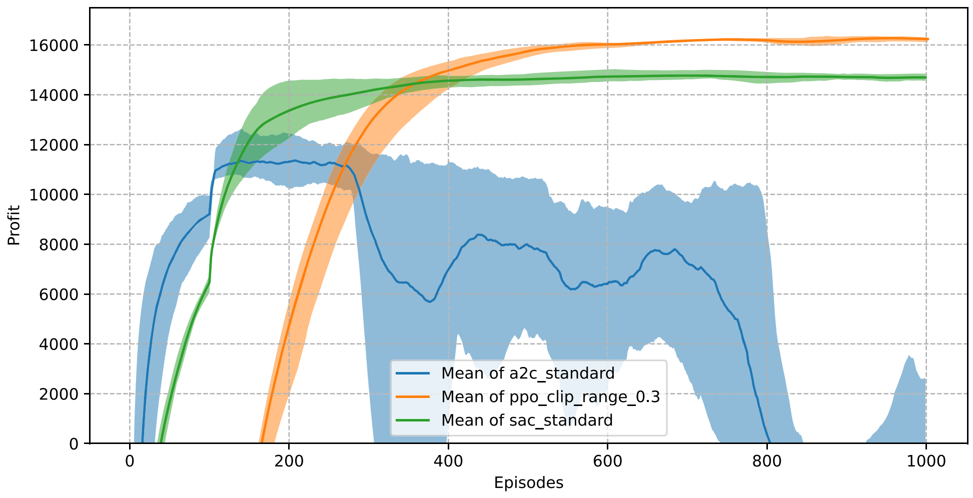


Fig. 10 Learning curves of A2C, PPO, and SAC in a monopoly scenario over 1000 episodes (4 runs)

1. a generalization of the known rule-based agent RBB (which for used and new products undercuts the minimum price of the other players by $h = 1$),
2. a rule-based agent that uses prices to regulate its stock but does not react to other agents' actions,
3. a passive agent that has fixed prices (6 as new price, 3 as used price, and 2 as buyback price), and
4. an agent created specifically for the oligopoly scenario: As a new price, it undercuts the median of the other sellers and regulates its stock to contain about 7 items.

Each step here is decomposed into fifths, cf. K, and the players set their prices in turn. In each of these fifths, $B/K = 4$ customers come to the marketplace ($B = 20$, cf. Table 1). The other parameters of the marketplace remain unchanged. The observation space is 18-dimensional (*in-circulation*, own stock level, and for each of the four competitors' three prices as well as the stock level). With regard to the state space, the scenario is more complex than the duopoly setup.

Figure 11 shows the learning curves on our oligopoly scenario (same scaling as in Fig. 2). The RL agents achieve learning success here as well, although the gains are naturally lower given the increased competition for the same demand. Consistent with our other experiments, PPO and SAC show stable learning curves. A2C shows familiar instabilities, but its performance at the peak actually lags only slightly behind that of the other algorithms. What is interesting in this experiment is that the SAC algorithm performs better than PPO, which was superior in peak performance in the other experiments. Because only 1000 episodes were trained due to the high training effort and the PPO curves still show a slight increase at the end, it cannot be ruled out that PPO catches up at a later time, but nevertheless, based on this learning curve SAC can be classified as superior.

The algorithm comparison on the oligopoly was also performed with the mixed reward function. Its results are similar, but the SAC runs there scatter much more. The learning curves for this experiment are given in the Appendix, see Fig. 18.

Figure 12 shows a typical training run for each of the three algorithms. The results here are positive. Each of the RL agents is able to significantly outperform all of its rule-based competitors during its training run. As expected, the agent with fixed prices performs the worst, as it is neither able to regulate its inventory nor to react to competitors' prices. The agent that only regulates its inventory ends up in second to last place, while the two rule-based agents that also react to competitor prices perform passably.

As in this oligopoly experiment the RL agents already outperform their competitors, cp. Experiment A, there is no need for the mixed (opportunistic) reward function, cp. Experiment B. Therefore, experiments on this setup were omitted.

4.7 Ablation Study for Steady State Results

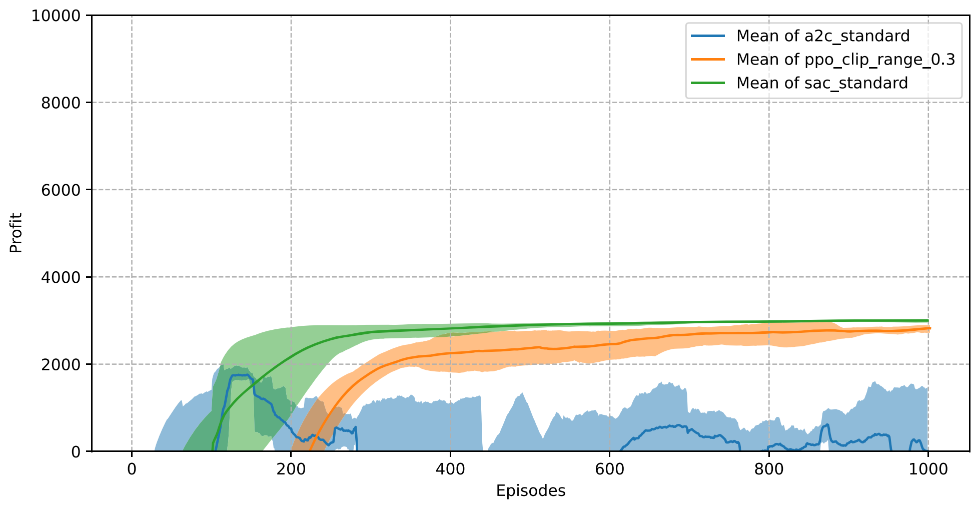
In this section, we provide an ablation study with respect to various model parameters in order to verify the general applicability of the proposed model framework as well as the robustness of the market results.

We summarize the results of the ablation experiments, cf. Table 2, in the following remark.

4.7.1 Remark 1

(i) The steady state of the *Base Case* is characterized as follows. In the presence of the aggressive and unyieldingly undercutting RBB competitor the RL agent sells less new items and less used items. However, the agent charges on average higher

Fig. 11 Learning curves of A2C, PPO, and SAC in an oligopoly scenario over 1000 episodes with normal reward and full observation (4 runs)



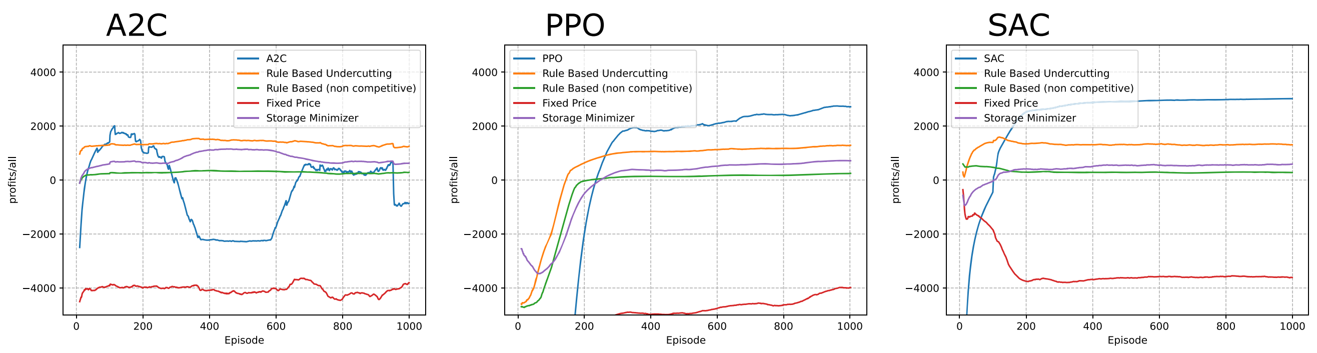


Fig. 12 Profits of typical training runs of A2C, PPO, and SAC compared to their rule-based peers in an oligopolistic setup

prices within both channels. Further, the RL agent repurchases less items at lower prices compared to the competitor. Overall, stock levels and rewards are similar for both firms.

(ii) An increasing customer *arrival intensity* B leads to overall plausible results: more sales, more repurchases, higher rewards for both firms, more virgin resources, more resources in use, and also more garbage. Both firms sell more new items at higher prices. Offer prices and sales prices for used items and repurchases are less affected. The RL agent meets demand using higher stocks.

(iii) Higher *production costs* c_{virgin} lead, as expected, to higher offer prices and higher sales prices for new items. We also observe higher rebuy prices as resources are more valuable. Overall, we have less in resources in use, less garbage, and less rewards for both firms. Compared to the Base Case, the RL agent sells more used items and repurchases more items; the competitor sells less new items. For higher c_{virgin} the RL agent beats the RBB strategy.

(iv) If the *customers' propensity to resell*, cf. w, increases rebuy prices drop and repurchases increase (for both firms). Offer and sales prices for used items are reduced and sales of used items increase for both firms. As expected, also the number of resources in use is smaller. Prices for new items are lower, however, while the RL agent sells more new items the competitor sells less new items compared to the Base Case. For larger w the competitor loses competition. While the RL agent effectively adapts his/her policy to the new conditions, the competitor's policy seems less well suited for the setup and should be re-tuned, which, however, is not straightforward.

(v) We varied the number of players K by considering different selections of competitors of the oligopoly experiment shown in Sect. 4.6.2. As expected, the RL agent's rewards are influenced by the competitiveness of the market, i.e. rewards decrease the more firms take part in the competition. In line with Fig. 12, we obtain that the RL agent (using PPO) again beats his/her competitors in the considered setups with 3, 4, and 5 players.

(vi) To test an alternative competitor policy, we consider an exemplary two-bound-like rule-based strategy (denoted by RSS) that avoids a race to the bottom and represents a less aggressive heuristic benchmark strategy:

$$p_{new}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \begin{cases} \min\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{new}^{(i)}\right\} - h & , \min\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{used}^{(i)}\right\} > c_{virgin} \\ p_{new}^{(\max)} & , else \end{cases} \tag{14}$$

$$p_{used}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \begin{cases} \min\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{used}^{(i)}\right\} - h & , \min\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{used}^{(i)}\right\} \geq 2 \\ 7 & , else \end{cases} \tag{15}$$

$$p_{rebuy}^{(k)}(N_{used}^{(k)}, \mathbf{p}_{new}, \mathbf{p}_{used}, \mathbf{p}_{rebuy}) := \begin{cases} \max\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{rebuy}^{(i)}\right\} + h & , \max\limits_{i\in\{1,\ldots,K\}\setminus\{k\}} \left\{p_{rebuy}^{(i)}\right\} < c_{virgin} - h \\ h & , else. \end{cases} \tag{16}$$

Overall, we obtain similar results, see Table 2. Note, since the RSS policy is less aggressive, the RL agent is able to achieve higher rewards. Further, the RSS policy suffers from holding costs as the inventory is not explicitly managed. This again shows that it is not easy to define a well balanced rule-based strategy in complex markets.

Ablation results for the choice of further model parameters characterizing the consumer behavior (θ_{new}, θ_{used}, κ_{used}), the threshold parameters of the RBB policy as well as the holding costs (c_{inv}) can be found at the end of the Appendix, see Table 7.

Table 2 Ablation Study: Steady state results, i.e., average offer and sales prices, sales, resource flows, stock levels, and rewards (per period) for the RL agent and the competitor, cf. "C"

		Base	B		c_{virgin}		w		K			RSS
		Case	10	30	2	4	0.025	0.075	3	4	5	
Offer prices	$\bar{p}^{RL}_{new,offer}$	6.12	5.84	6.58	5.94	6.63	6.25	5.26	6.02	5.64	5.60	6.01
	$\bar{p}^{C}_{new,offer}$	5.12	4.48	5.57	4.95	5.64	5.25	4.34	5.01	4.60	4.01	4.99
	$\bar{p}^{RL}_{used,offer}$	3.92	4.42	4.12	4.24	4.01	3.84	3.26	3.92	4.15	3.35	3.93
	$\bar{p}^{C}_{used,offer}$	3.34	3.79	3.56	3.60	3.64	3.36	2.68	2.96	2.82	2.60	2.92
	$\bar{p}^{RL}_{rebuy,offer}$	0.23	0.00	0.15	0.01	0.42	0.24	0.10	0.22	0.42	0.68	0.41
	$\bar{p}^{C}_{rebuy,offer}$	0.72	0.56	0.84	0.62	0.88	0.80	0.70	1.19	1.38	1.48	2.00
Sales prices	$\bar{p}^{RL}_{new,sold}$	6.01	5.84	6.50	5.88	6.56	6.16	5.21	6.00	5.68	5.56	5.85
	$\bar{p}^{C}_{new,sold}$	5.09	4.83	5.53	4.94	5.63	5.23	4.33	5.02	4.61	4.01	4.95
	$\bar{p}^{RL}_{used,sold}$	3.61	4.04	4.01	3.81	3.89	3.65	3.16	3.61	3.70	3.26	3.76
	$\bar{p}^{C}_{used,sold}$	3.07	3.61	3.37	3.33	3.44	3.11	2.51	3.19	2.84	2.63	2.88
	$\bar{p}^{RL}_{rebuy,sold}$	0.39	0.00	0.20	0.00	0.64	0.49	0.16	0.34	0.59	0.79	0.58
	$\bar{p}^{C}_{rebuy,sold}$	0.92	0.89	1.05	0.87	1.04	0.95	0.87	1.29	1.31	1.38	2.00
Sales	$\bar{X}^{RL}_{new}$	3.96	2.42	4.74	4.44	4.04	3.40	4.14	2.84	2.72	1.86	3.74
	$\bar{X}^{C}_{new}$	6.52	3.78	9.88	6.74	5.60	6.76	6.24	4.30	4.32	3.62	6.74
	$\bar{X}^{RL}_{used}$	1.72	0.62	2.50	1.46	1.98	1.88	2.42	1.24	0.88	1.12	2.06
	$\bar{X}^{C}_{used}$	3.63	0.98	5.14	3.00	3.42	3.82	4.20	3.34	2.90	2.06	3.56
	$\bar{X}^{RL}_{rebuy}$	1.74	0.68	2.54	1.44	2.08	1.78	2.34	1.18	0.84	1.14	2.06
	$\bar{X}^{C}_{rebuy}$	3.62	1.04	5.20	3.00	3.42	3.74	4.12	3.34	2.84	2.12	7.92
Resource flows, stocks & rewards	$\bar{N}_{in\,use}$	258	156	349	265	232	482	173	184	238	293	244
	$\bar{N}_{garbage}$	5.11	4.46	7.20	6.84	4.16	4.84	3.90	0.62	0.18	0.18	0.82
	$\bar{N}_{virgin}$	10.50	6.31	14.56	11.18	9.64	10.16	10.38	7.14	7.04	5.48	10.48
	$\bar{N}^{RL}_{stock}$	8.77	4.24	13.56	8.24	8.04	22.04	11.46	11.42	3.92	5.04	6.42
	$\bar{N}^{C}_{stock}$	8.35	8.06	9.28	8.78	7.98	8.74	8.64	6.02	5.60	5.24	29.88
	$\bar{G}^{RL}_{reward}$	15.60	8.94	24.72	21.96	15.89	14.54	15.30	11.45	9.66	5.81	15.74
	$\bar{G}^{C}_{reward}$	16.91	8.00	28.93	24.40	12.80	18.38	9.48	6.56	6.43	3.12	11.69

We vary different parameters with respect to our Base Case ($\gamma = 0.99$, $c_{virgin} = 3$, $c_{inv} = 0.1$, $B = 20$, $w = 0.05$, $\theta_{new} = 0.8$, $\theta_{used} = 0.5$, $\kappa_{used} = 0.55$, in a duopoly ($K = 2$) vs. the RBB policy with $h = 1$, $M = 100$, cf. Sect. 4.1.3). $K = 3, 4, 5$ subsequently extends the Base Case by a 3rd player "Rule Based (non competitive)", a 4th player "Fixed Price", and a 5th player "Storage Minimizer", as used in Sect. 4.6.2, see Fig. 12; the table contains results for the competitor RBB. RSS, cf. (14)–(16), exchanges the policy RBB

5 Calibrating Environments from Observable Data

5.1 Using Synthetic Test Environments for Pre-training

Naturally, the question arises of how to integrate the proposed model and solution framework into real-world information systems to solve companies' individual revenue management challenges. In this regard, it would be necessary to calibrate the environment of our model, cf. Section 3, to the use case such that it mimics the market under consideration. One way to do this is based on domain knowledge and corresponding experts. Alternatively, a suitable environment could also be defined by using historical data. First, besides known or easy-to-estimate market and cost parameters, one would have to estimate consumers' demand probabilities under competition, see, e.g., Schlosser and Boissier (2018a, b). For this purpose, established methods and sophisticated forecasting tools are available, see, e.g., Salinas et al. (2020) and references therein. Second, the underlying competitors' price reactions need to be imitated. Based on suitable data regarding own price adjustments and competitors' price reactions, standard methods can be used to compute and predict data-driven price anticipations, see, e.g., Schlosser and Richly (2019).

These two dynamics can then be used within an artificial model environment to generate price reactions, sales, and inventory levels for multiple parties, i.e., the agent and the competitors. Finally, the calibrated artificial environment can be used to train an RL agent without being forced to do that in practice. Finally, if obtained pricing strategies are plausible, they can be applied and trained further in the specific real-life application.

5.2 Test Example for the Base Case

To test the applicability of the sketched approach, we distinguish between an original test environment A (which is used to produce realized market data) and a fitted auxiliary environment B (which is used supposed to be used for training). We consider the following example.

Example 1 As environment A, we consider the setup of the Base Case, cf. Section 4.1, where the competing firm 2 plays again the RBB strategy. (Our) firm 1 initially plays the two-bound RSS strategy, see (14)–(16). To ensure a sufficiently diversified dataset, we add an exploration rate of $\varepsilon = 0.1$ (for i.i.d. uniform prices within set A) to firm 1's policy. For this setup, we simulate test data for $D = 20$ episodes (with $E = 500$ periods each) representing an available set of historic market data.

5.2.1 Fitting Sales Probabilities

Given the test data produced by environment A, cf. Example 1, we estimate the consumer behavior via expected sales and repurchases. We consider time intervals of length 1/2, i.e., half periods $(t, t+0.5)$, $t = 0, 0.5, ..., T$, where $T = E \times 20$. Next, we seek to explain the number of sales of new items for firm 1 within $(t, t+0.5)$, i.e., the dependent variable is $X_{new}^{(1)}(t, t+0.5)$. As explanatory variables we use the corresponding prices of new and used items for both firms, i.e., $p_{new}^{(1)}(t, t+0.5), p_{used}^{(1)}(t, t+0.5), p_{new}^{(2)}(t, t+0.5), p_{used}^{(2)}\ (t, t+0.5)$. Using, e.g., simple OLS regression providing corresponding β coefficients, allows to predict average new sales for firm 1 ($\hat{X}_{new}^{(1)}(t, t+0.5)$) for half periods for any given prices. Further, to simulate integer sales we sample the neighbors of $\hat{X}_{new}^{(1)}(t, t+0.5)$ with corresponding probabilities. Recall, to include the standard deviation more accurately also alternatives could be used.

Moreover, to simulate new sales for the competing firm 2, we exploit a symmetric setup, i.e., we use the same regression result and switch the perspectives of both firms regarding their prices. Note, this way, the competitor's sales do not need to be part of firm 1's observable dataset.

Further, sales for used items and repurchases can be predicted analogously for both firms. Note, for used sales, we also include the firm's own inventory level as an additional feature. For repurchases, we of course also include the current rebuy prices of both firms, cf. $p_{rebuy}^{(1)}(t, t+0.5)$ and $p_{rebuy}^{(2)}(t, t+0.5)$.

Overall, for Example 1, the described regressions for demand obtained results with an $R^2 = 0.52$.

5.2.2 Fitting Competitor's Price Reactions

Given the test data produced by environment A, cf. Example 1, we estimate the competitor's price reactions as follows. We consider time intervals of length 1, i.e., full periods $(t, t+1)$, $t = 0, 0.5, ..., T$, where $T = E \times 20$. First, we seek to explain the competitor's price for new items for period $(t+0.5, t+1.5)$, i.e., the dependent variable is $p_{new}^{(2)}(t+0.5, t+1.5)$. As explanatory variables we use the previous competitor price $p_{new}^{(2)}(t-0.5, t+0.5)$ as well as, e.g., a family of binary features $1_{\{p_{new}^{(1)}(t,t+1)<j\}}$, $j = h, 2h, ..., p_{new}^{(\max)}$, $h = 1$, $p_{new}^{(\max)} = 10$, which particularly allow to express nonlinear response functions to firm 1's price $p_{new}^{(1)}(t, t+1)$. Note, this approach provides deterministic predictions. Naturally, also extended models to predict mixed strategies are possible (but not in focus of this example). Further, firm 2's price reactions for used items as well as repurchases can be predicted analogously.

Finally, for the fitted price reactions (of the deterministic RBB policy, cf. Example 1), the described regressions obtained results with an $R^2 = 0.90$.

5.2.3 Training an Agent on the Fitted Environment B

The fitted sales probabilities (Sect. 5.2.1) and the fitted price reactions of the competitor (Sect. 5.2.2) are now used to define an auxiliary test environment B that mimics environment A without using explicit knowledge of the original dynamics. While model parameters such as c_{virgin}, γ, and c_{inv} can be assumed to be given, previous parameters describing the consumer behavior (B, θ_{new}, θ_{used}, κ_{used}) or the competitor's policy (M, h) are not required anymore.

The simulation of the environment B is straightforward. Price updates, sales realizations, evolutions of stock levels, and (accumulated) rewards are evaluated as before, cf. Sect. 3.2. Similarly, also the training of RL agents, cf. Sect. 3.3, remains unchanged in the fitted environment B. Recall, opposed to real-life applications the number of training runs in environment B is not limited and can be fully exploited. In this example, we used 2000 episodes to train a PPO agent on environment B.

5.2.4 *Evaluation of the Trained Agent on the Original Environment A*

Finally, the PPO agent trained on environment B, cf. Section 5.2.3, is evaluated in the original (hidden) environment A. Note, the Base Case solution against RBB, cf. Section 4.2, serves as a baseline and provides an upper bound for agents that were not allowed to interact with environment A, see Table 3. In our example, the pre-trained PPO agent received a performance of 94% compared to the Base Case solution, which shows that the overall approach works well as long as the fit of environment B is accurate.

In further examples using less data or abstain from exploration to obtain diversified data – as expected – results were significantly worse. Typically, an inaccurate or incomplete fit of the environment B leads to learned policies that are too optimistic in certain regions of the state space, which fires back in the original market. To actively resolve such issues and to improve the fit of environment B (besides exploration) one could iteratively enrich the dataset by shortly testing the current policy on environment A and subsequently continue to train on an updated fit of environment B.

Table 3 Performance Comparison: Steady state results for Agent B (i) after training on the auxiliary Environment B and (ii) evaluated in the original Environment A, cf. Example 1

		Base case	Trained agent B applied on Env. A		Agent B trained on Env. B	
Offer prices	$\bar{p}^{RL}_{new,offer}$	6.12	5.52	(0.90)	5.34	(0.87)
	$\bar{p}^{C}_{new,offer}$	5.12	4.57	(0.89)	4.57	(0.89)
	$\bar{p}^{RL}_{used,offer}$	3.92	3.94	(1.01)	4.03	(1.03)
	$\bar{p}^{C}_{used,offer}$	3.34	3.19	(0.95)	2.44	(0.73)
	$\bar{p}^{RL}_{rebuy,offer}$	0.23	0.02	(0.11)	0.03	(0.15)
	$\bar{p}^{C}_{rebuy,offer}$	0.72	0.61	(0.84)	0.25	(0.34)
Sales prices	$\bar{p}^{RL}_{new,sold}$	6.01	5.42	(0.90)	5.19	(0.86)
	$\bar{p}^{C}_{new,sold}$	5.09	4.57	(0.90)	4.55	(0.89)
	$\bar{p}^{RL}_{used,sold}$	3.61	3.79	(1.05)	3.94	(1.09)
	$\bar{p}^{C}_{used,sold}$	3.07	2.84	(0.93)	2.44	(0.79)
	$\bar{p}^{RL}_{rebuy,sold}$	0.39	0.04	(0.09)	0.05	(0.12)
	$\bar{p}^{C}_{rebuy,sold}$	0.92	0.82	(0.89)	0.25	(0.27)
Sales	$\bar{X}^{RL}_{new}$	3.96	4.50	(1.14)	3.26	(0.82)
	$\bar{X}^{C}_{new}$	6.52	6.98	(1.07)	5.00	(0.77)
	$\bar{X}^{RL}_{used}$	1.72	1.38	(0.80)	2.62	(1.52)
	$\bar{X}^{C}_{used}$	3.63	3.62	(1.00)	3.14	(0.87)
	$\bar{X}^{RL}_{rebuy}$	1.74	1.58	(0.91)	2.68	(1.54)
	$\bar{X}^{C}_{rebuy}$	3.62	3.62	(1.00)	3.14	(0.87)
Resource flows, stocks & rewards	$\bar{N}_{in\ use}$	258	277	(1.08)	219	(0.85)
	$\bar{N}_{garbage}$	5.11	6.58	(1.29)	2.54	(0.50)
	$\bar{N}_{virgin}$	10.50	11.48	(1.09)	8.26	(0.79)
	$\bar{N}^{RL}_{stock}$	8.77	9.12	(1.04)	5.64	(0.64)
	$\bar{N}^{C}_{stock}$	8.35	9.40	(1.13)	1.58	(0.19)
	$\bar{G}^{RL}_{reward}$	15.60	14.70	(0.94)	16.75	(1.07)
	$\bar{G}^{C}_{reward}$	16.91	14.20	(0.84)	13.53	(0.80)

The numbers in brackets show the relative comparison to the Base Case, where Agent A is directly trained on Environment A

6 Discussion

In the following, we summarize our main insights, discuss limitations, and propose ideas on how to introduce RL agents in real-life markets.

6.1 Main Insights

Our main insights can be summarized as follows:

- RL algorithms can successfully be applied to complex recommerce markets with unknown underlying dynamics regarding consumers' and competitors' behaviors.
- RL agents are able to clearly outperform commonly established rule-based agents.
- In our experiments, at most a few thousand episodes were necessary to train the agents.
- PPO and SAC performed best in duopoly as well as in oligopoly scenarios.
- The default hyperparameters of the RL algorithms worked well; hardly any tuning was necessary.
- Steady-states of controlled markets are obtained after about a few hundred periods.
- The non-observability of both the number of resources in use and the competitors' inventories is not critical; results hardly depend on whether they are part of the state space.
- Applying self-play allows finding robust pricing strategies which are effective against different competitor strategies, even ones not seen in training.
- Our numerical examples show that changes in the parameters or the setup lead to good-natured and plausible solutions, which verifies the general applicability of the model.
- Agents can be successfully applied to incompletely known markets by pre-training them on auxiliary markets that are calibrated based on realized market data of the (hidden) target market.

6.2 Limitations and Extensions

The lack of more or better benchmark strategies and alternative RL algorithms, cf. Section 3.3.2, is a limitation. Also, results are to some degree stochastic and a larger number of runs would have to be used to quantify mean values and their standard deviations accurately. Further, the successful calibration of auxiliary training environments deserves further analysis. However, in the existing framework, alternative competitor strategies, calibration techniques, and other RL algorithms can be easily added and tested in greater detail. Moreover, the basic model could also be extended to capture more complex settings. For instance, for each firm, we may additionally consider a technology state serving as a sustainability image (cf. greenness, signaling, etc.), which increases demand. Further, this state could be stimulated via corresponding investment efforts and otherwise depreciates over time. Another research direction is to include strategic consumer behavior.

7 Conclusion

In this paper, we have proposed a market simulation framework for recommerce markets under competition. The framework has modular components, which allow to study different pricing strategies in different market scenarios. The simulation is designed in such a way that self-learning RL algorithms can be easily integrated and compared. Further, we have studied the performance of state-of-the-art RL algorithms in different recommerce markets. We find that PPO and SAC performed best while being tested in various market setups as well as different information structures.

Detailed analyses of the policies obtained allow to better understand resource flows in recommerce markets and to infer managerial insights. Further, we show that RL agents can be promising candidates for practical applications as long as a sufficient amount of historic data is available and synthetic training environments can be calibrated accurately.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s12599-023-00841-8.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Bertsekas DP (2019) Reinforcement learning and optimal control. Athena Scientific, Nashua

Bocken NM, de Pauw I, Bakker C, van der Grinten B (2016) Product design and business model strategies for a circular economy. J Indust Prod Eng 33(5):308–320

Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J, Zaremba W (2016) Openai gym. arXiv preprint arXiv:1606.01540

Chen F, Lu A, Wu H, Dou R, Wang X (2022) Optimal strategies on pricing and resource allocation for cloud services with service guarantees. Comput Ind Eng 165(107):957

Chen M, Chen ZL (2015) Recent developments in dynamic pricing research: multiple products, competition, and limited demand information. Prod Oper Manag 24:704–731

Colony GF (2005) As I.T. goes, so goes Forrester? New York Times https://www.nytimes.com/2005/02/18/business/yourmoney/as-it-goes-so-goes-forrester.html. Accessed 21 June 2022

Commoner B (1972) The environmental cost of economic growth. Popul Resour Environ 3:343–63

den Boer AV (2015) Dynamic pricing and learning: historical origins, current research, and new directions. Surv Oper Res Manag Sci 20:1–18

DiMicco JM, Maes P, Greenwald A (2003) Learning curve: a simulation-based approach to dynamic pricing. Electron Commer Res 3(3–4):245–276

Fujimoto S, van Hoof H, Meger D (2018) Addressing function approximation error in actor-critic methods. CoRR abs/1802.09477, arXiv:1802.09477

Gerpott T, Berends J (2022) Competitive pricing on online markets: a literature review. J Reven Pricing Manag 21:596–622

Gönsch J (2014) Buying used products for remanufacturing: negotiating or posted pricing. J Bus Econ 84:715–747

Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: ICML 2018, Proceedings of machine learning research, vol 80, pp 1856–1865

Hawlitschek F (2021) The future of waste management. Bus Inf Syst Eng 63:207–211

Hill A et al (2018) Stable baselines. https://github.com/hill-a/stable-baselines, accessed 21 June 2022

Kastius A, Schlosser R (2022) Dynamic pricing under competition using reinforcement learning. J Reven Pricing Manag 21:50–63

Kephart JO, Hanson JE, Greenwald A (2000) Dynamic pricing by software agents. Comput Netw 32(6):731–752

Kirchherr J, Reike D, Hekkert M (2017) Conceptualizing the circular economy: an analysis of 114 definitions. Res Conserv Recycl 127:221–232

Klein R, Koch S, Steinhardt C, Strauss A (2020) A review of revenue management: recent generalizations and advances in industry applications. Europ J Oper Res 284:397–412

Maestre R, Duque JR, Rubio A, Arévalo J (2018) Reinforcement learning for fair dynamic pricing. In: Arai K, Kapoor S, Bhatia R (eds) Intelligent Systems and Applications - Proceedings of the 2018 Intelligent Systems Conference, IntelliSys 2018, Advances in Intelligent Systems and Computing. Springer, Heidelberg, vol 868, pp 120–135

Mnih V et al (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529–533

Mnih V et al (2016) Asynchronous methods for deep reinforcement learning. In: International conference on machine learning, PMLR, pp 1928–1937

Paszke A et al (2019) PyTorch: An imperative style, high-performance deep learning library. In: Wallach H et al (eds) Advances in neural information processing systems 32, pp 8024–8035

Rabe L (2020) Reuse und Secondhand in Deutschland. Wuppertal Institut. https://de.statista.com/statistik/daten/studie/1248873/umfrage/bevorzugter-kanal-fuer-den-verkauf-von-secondhand-produkten-in-deutschland. Accessed 21 June 2022

Salinas D, Flunkert V, Gasthaus J, Januschowski T (2020) DeepAR: probabilistic forecasting with autoregressive recurrent networks. Int J Forecast 36(3):1181–1191

Savaskan RC, Bhattacharya S, Van Wassenhove LN (2004) Closed-loop supply chain models with product remanufacturing. Manag Sci 50(2):239–252

Schlosser R, Boissier M (2018) Dealing with the dimensionality curse in dynamic pricing competition: using frequent repricing to compensate imperfect market anticipations. Comput Oper Res 100:26–42

Schlosser R, Boissier M (2018b) Dynamic pricing under competition on online marketplaces: a data-driven approach. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining pp 705–714

Schlosser R, Richly K (2019) Dynamic pricing under competition with data-driven price anticipations and endogenous reference price effects. J Reven Pricing Manag 18:451–464

Schlosser R, Chenavaz R, Dimitrov S (2021) Circular economy: joint dynamic pricing and recycling investments. Intl J Prod Econ 236:108117

Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347

Shihab SAM, Wei P (2022) A deep reinforcement learning approach to seat inventory control for airline revenue management. J Reven Pricing Manag 21:1–17

Silver D, Lever G, Heess N, Degris T, Wierstra D, Riedmiller M (2014) Deterministic policy gradient algorithms. In: ICML'14, Vol. I, p 387-395

Silver D et al (2017) Mastering the game of go without human knowledge. Nature 550(7676):354–359

Stahel WR (2016) The circular economy. Nature 531(7595):435–438

Statista (2020) Wie äußert sich bei ihnen der fokus auf nachhaltige mode beim shopping? Statista Research Department https://de.statista.com/statistik/daten/studie/1179997/umfrage/umfrage-unter-verbrauchern-zu-nachhaltigemmodekauf-in-deutschland/. Accessed 21 June 2022

Strauss AK, Klein R, Steinhardt C (2018) A review of choice-based revenue management: theory and methods. Europ J Oper Res 271:375–387

Sutton RS, Barto AG (2018) Reinforcement learning - an introduction. In: Adaptive computation and machine learning, 2nd edn. MIT Press, Cambridge

Talluri KT, Van Ryzin GJ (2006) The theory and practice of revenue management. Springer, Heidelberg

Teh YW et al (2017) Distral: robust multitask reinforcement learning. In: Advances in neural information processing systems 30: Annual conference on neural information processing systems 2017, pp 4496–4506

Thomas O et al (2020) Global crises and the role of BISE. Bus Inf Syst Eng 62:385–396

Tsao Y, Beyene TD, Thanh V, Gebeyehu SG (2022) Power distribution network design considering dynamic and differential pricing, buy-back, and carbon trading. Comput Ind Eng 172(108):567

Turan B, Pedarsani R, Alizadeh M (2020) Dynamic pricing and fleet management for electric autonomous mobility on demand systems. Transp Res Part C: Emerg Technol 121(102):829

van de Geer R, den Boer A, Bayliss C et al (2019) Dynamic pricing and learning with competition: insights from the dynamic pricing challenge at the 2017 INFORMS RM & pricing conference. J Reven Pricing Manag 18:185–203

Weinhardt C et al (2021) Welcome to economies in IS! Bus Inf Syst Eng 63:325–328

Wen D, Xiao T, Dastani M (2022) Pricing strategy and collection rate for a supply chain considering environmental responsibility behaviors and rationality degree. Comput Ind Eng 169(108):290

Yang Y, Chu W, Wu C (2022) Learning customer preferences and dynamic pricing for perishable products. Comput Ind Eng 171(108):440

Zhu Z, Lin K, Zhou J (2020) Transfer learning in deep reinforcement learning: a survey. CoRR abs/2009.07888, arXiv:2009.07888