Deng, Jifei; Sierla, Seppo; Sun, Jie; Vyatkin, Valeriy

Offline reinforcement learning for industrial process control: a case study from steel industry

Published in:
Information Sciences (Elsevier)

DOI:
10.1016/j.ins.2023.03.019

Published: 01/06/2023

Document Version
Publisher's PDF, also known as Version of record

Published under the following license:
CC BY

Please cite the original version:
https://doi.org/10.1016/j.ins.2023.03.019
Offline reinforcement learning for industrial process control: A case study from steel industry

Jifei Deng \textsuperscript{a,}\textsuperscript{*}, Seppo Sierla \textsuperscript{a}, Jie Sun \textsuperscript{b,}\textsuperscript{*}, Valeriy Vyatkin \textsuperscript{a,\textsuperscript{c}}

\textsuperscript{a} Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, Espoo, Finland
\textsuperscript{b} State Key Laboratory of Rolling and Automation, Northeastern University, Shenyang, China
\textsuperscript{c} Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology, Luleå, Sweden

\textbf{ARTICLE INFO}

\textbf{Keywords:}
Offline reinforcement learning
Deep ensemble
Industrial process control
Steel industry
Strip rolling

\textbf{ABSTRACT}

Flatness is a crucial indicator of strip quality that presents a challenge in regulation due to the high-speed process and the nonlinear relationship between flatness and process parameters. Conventional methods for controlling flatness are based on the first principles, empirical models, and predesigned rules, which are less adaptable to changing rolling conditions. To address this limitation, this paper proposed an offline reinforcement learning (RL) based data-driven method for flatness control. Based on the data collected from a factory, the offline RL method can learn the process dynamics from data to generate a control policy. Unlike online RL methods, the proposed method does not require a simulator for training, the policy can be potentially safer and more accurate since a simulator involves simplifications that can introduce bias. To obtain a steady performance, the proposed method incorporated ensemble Q-functions into policy evaluation to address uncertainty estimation. To address distributional shifts, based on Q-values from ensemble Q-functions, behavior cloning was added to policy improvement. Simulation and comparison results showed that the proposed method outperformed the state-of-the-art offline RL methods and achieved the best performance in producing strips with lower flatness.

1. Introduction

In the steel-making industry, strip rolling is a manufacturing process in which steel strips are passed through rolling mills \cite{1}. An industrial process control system monitors the strip in real-time and controls the mills to produce strips with thickness, crown, and flatness specified by the customers. Currently, in real factories, the feedback control loop is designed using the Proportional-Integral (PI) controller \cite{2}. To control the flatness of the strip, a PI controller computes process parameters for the final mill based on the

Abbreviations: RL, Reinforcement Learning; PI, Proportional Integral; DDPG, Deep Deterministic Policy Gradients; SOTA, State-of-the-art; SAC, Soft Actor-critic; TD3, Twin Delayed Deep Deterministic Policy Gradient; BEAR, Bootstrapping Error Accumulation Reduction; SPIBB, Safe Policy Improvement with Baseline Bootstrapping; TD3-BC, Twin Delayed DDPG with Behavioral Cloning; OOD, Out of Distribution; CQL, Conservative Q-learning; IQL, Implicit Q-learning; MDP, Markov Decision Process; SQL, Stable Q-learning; PNN, Probabilistic Neural Network; STD, Standard Deviation; S, State; A, Action; \(L_{ref}\), Unit of Flatness; \(\Delta L\), The difference between the length of a given strip and the reference strip; \(D_{offline}\), Offline Dataset; \(a'\), Out of Distribution Actions; \(Q^\phi\), Value Function Network; \(\bar{Q}\), Target Value Function Network; \(\phi\), Policy Network; \(L\), Objective Function; \(\pi\), Behavior Policy; \(\pi^\beta\), Learned Policy; \(\bar{Q}\), Target Value Function.

* Corresponding authors.
E-mail addresses: jifei.deng@aalto.fi (J. Deng), sunjie@ral.neu.edu.cn (J. Sun).
observed flatness provided by the measuring instrument.

Such a conventional control method requires a complex parameter-tuning step. When building the line, control, and material processing engineers collaborate to tune the controller’s parameters [3]. The trial-and-error method is commonly used in practice when the effectiveness of each parameter is evaluated by running the mills, which could result in wasting strips and casualties for personnel [2]. Moreover, additional procedures for avoiding potential problems and risks are required to ensure safety and efficiency. However, in a real process, quality issues can appear randomly because the process is dynamic and uncertain due to the difficult-to-predict factors, which are fundamental properties of the process that cannot be changed without redesigning it [4,5]. Such controllers are hard to adapt to the changing process dynamics because their parameters are fixed.

Since data-driven methods for control have been increasingly used over the recent years [6,7], this paper studies the usability of such methods for flatness control. This control method learns from the real data how to address the above limitations. Given an environment, which could be a real system or an approximated dynamic model thereof, reinforcement learning (RL) learns a policy through interactions between the agent and the environment [8]. The policy can be thought of as a learning-based method for control and this method has been used in various fields, such as autonomous vehicles [9], drone aircraft [10], and data management [11]. Compared to the conventional methods that require parameter tuning, the RL-based method can understand the process dynamics and learn an intelligent and adaptive control solution for the system [8].

Without the requirements of manual trial-and-error, RL could learn a policy of a complicated system more safely and effectively. In the strip rolling industry, to achieve lower costs and higher product quality, Gamal et al. adopted Deep Deterministic Policy Gradients (DDPG) to train a policy for controlling the roll gap and wire of a hot rolling process [12]. Deng et al. proposed an RL-based control using proximal policy optimization and ensemble learning to control the flatness of cold rolling [2]. Zeng et al. proposed a DDPG-based control method to address the disturbances and uncertainties in thickness and tension in an unsteady rolling process [13]. Although online RL and its application in control [2,12–14] are popular, specific settings are necessary for training in different circumstances. In industry, especially in safety-critical cases, the RL agent is not allowed to interact with the real system, because various actions including dangerous ones would be generated that could damage the machines or even harm people. To address this problem, black-box models were adopted to model the process [2,13], and training and evaluation were carried out on these models.

According to the above analysis, the online model-free RL methods rely on the environment heavily, so the performance of the policy could be highly influenced by the accuracy of the environment. For industrial cases, physics-based simulators and black-box models are common methods for modeling, achieved through approximating and simplifying the real systems. To build high-fidelity physics-based simulators, domain knowledge is needed, so the real system needs to be fully analyzed. Black-box models require a series of steps, including data processing, feature selection, model selection, and hyperparameter tuning, which must be performed by experienced engineers. Thus, physics-based simulators and black-box models are biased. Although a good performance is obtained after being trained, the bias problem makes the policy unable to be deployed to control the real system. The gap between the environment and reality, which is also called “reality gap”, is a demanding problem for industrial applications of RL [15].

Considering the principles of PI control and online RL methods, this paper studies a novel offline RL method for flatness control. With the ability to be adaptive to the rolling condition, online RL methods were modified to be offline to learn a policy from a fixed dataset rather than through interacting with the environment [16]. Since real data from the factory are available, like supervised learning, offline RL methods extract a policy that formulates a mapping from states to actions. In industry, such methods can generate safer and more accurate policies without the use of a biased environment.

The goal of proposing offline RL for flatness control in strip rolling is to address the bottlenecks of the traditional control methods and online RL methods, which are summarized as follows:

1. Offline RL learns an adaptive policy from the process data, while the traditional PI control method requires expert knowledge of the specific industrial cases and complicated parameter tuning.
2. Offline RL learns from a static dataset to directly extract information to generate a policy, while online RL requires a model of the real process for training, using simplification, approximation, and black-box methods, the reality gap between the model and the real system reduces reliability.

This paper proposes a novel offline RL method to learn a stable and safe policy for flatness control of a cold rolling line. The data were obtained from the control system of an operational steel mill. Based on Twin Delayed DDPG with behavior cloning [17], ensemble Q-functions are used in policy evaluation to address problems in uncertainty estimates. The obtained stable Q-values are used for policy improvement. Moreover, a data-driven black-box model of the strip rolling line is used to evaluate the trained RL policy. The main intended contributions of this paper are as follows:

1. To learn a reliable policy, a data-driven control method was proposed using offline RL to extract information from a pre-collected dataset rather than interacting with a simplified environment. A novel offline RL method was proposed to obtain stable Q-values for policy improvement.
2. To ensure safety, a data-driven evaluation was adopted for comparison and simulation to support the deployment readiness of the policy.
3. Comparisons with the state-of-the-art (SOTA) methods and the existing PI controllers at the case study factory were carried out, the proposed method showed the best performance in flatness control of strip rolling.

The rest of this paper is organized as follows. The existing methods, SOTA offline RL methods, and their applications in control are
reviewed in Section 2. Section 3 discusses the basic principles of offline RL and the proposed method. A case study, including the details of the data and production line, as well as experimental settings, is presented in Section 4. Section 5 analyzes the results of the proposed method. Section 6 concludes the paper and describes future research plans.

2. Literature review

Conventional controllers such as PI are popular in industrial process control. However, to improve control performance and product quality, and to lower manual engineering costs, the limitations of the conventional controllers should be addressed. An intelligent, adaptive, and easy-to-implement control method is required to address the problems of trial-and-error-based parameter tuning and multi-field cooperation for system building. With an RL-based data-driven control method, the policy could understand the process dynamics to find out the optimal solutions [2,5]. In [2], based on a black-box model, an online RL method was proposed for flatness control in strip rolling. Such online RL methods rely on the approximated dynamic models, and although they proved the feasibility of RL in strip rolling control, the learned policy can be unsafe and unreliable for deployment.

Compared with online RL which needs interactions between the agent and its environment, offline RL is similar to supervised learning and can extract the information from a pre-collected dataset to learn a policy [16]. This means an environment is not required, and the limitations caused by the simplified environments do not apply. Especially in industries, real data are available, but a high-fidelity environment is difficult to build. Thus, offline RL is a potentially promising tool for flatness control in strip rolling. Behavior cloning (BC) directly learns a policy by using supervised learning on observation-action pairs, it is easy to implement, but the generalization of the policy is limited, and it cannot recover from errors [18]. In practice, off-policy RL algorithms (e.g., soft actor-critic (SAC) [19], twin delayed deep deterministic policy gradient (TD3) [20]) can technically be trained using a fixed dataset. However, the accuracy of the Q-function regression targets depends on the estimate of the Q-values for actions that are outside the distribution of actions on which the Q-function has been trained. When the learned policy differs substantially from the behavior policy, this discrepancy can result in highly erroneous target Q-values [16,21]. This problem is further exacerbated by policy improvement, in which the learned policy is optimized by maximizing the Q-values.

To address the above problem, Bootstrapping Error Accumulation Reduction (BEAR) [21], Safe Policy Improvement with Baseline Bootstrapping (SPIBB) [22], and Twin Delayed DDPG with Behavioral Cloning (TD3-BC) [17] were proposed based on constraining the learned policy to the support of a behavior policy without making any changes to the estimation of Q-values. In addition to constraining the policy, another approach is to add a regularizer to the Q-function to avoid overestimation for out of distribution (OOD) actions, as in Conservative Q-learning (CQL) [23], and Implicit Q-learning (IQL) [24]. These methods mainly focus on lowering or removing the estimation of OOD actions using complicated algorithmic modifications.

Unlike the general online [19,20] and offline [17,21–24] RL algorithms that were designed and evaluated using benchmark environments, industrial users prefer practical methods that can maintain a balance between effectiveness and complexity. Offline RL has been applied to different fields, such as recommendation systems [25], smart grids [26], power control [27], robotics [28], and autonomous driving [29]. However, there is a lack of offline RL research for process industry applications such as strip rolling.

In this paper, motivated by the problems discussed above, a novel offline RL framework is proposed for strip rolling control. Owing to the real data collected from the factory, the learned policy of offline RL can be safer, more reliable, and more practical for deployment.

3. Strip rolling

Flatness control in strip rolling is chosen as the case to evaluate the RL-based data-driven method. In Fig. 1, the finishing rolling process consists of five finishing rolling mills, the strips will move through each mill. The controller will compute process parameters for the mills after receiving the flatness values measured by the sensor. Readings are taken from four flatness sensors to obtain the state vector, which is defined as $S = (s^0, s^1, s^2, s^3)$. Process parameters include rolling force, bending force of the work roll and intermediate roll, and the roll gap tilting are recorded by the process control system, which is defined as $A = (a^0, a^1, a^2, a^3)$. The flatness of a strip is computed as follows [30]:

![Fig. 1. An example of a cold rolling line.](image-url)
\[ I_f = \left( \frac{\Delta L}{L_{ref}} \right) \times 10^5 \]  

where \( L_{ref} \) is the length of the strip as a reference and \( \Delta L \) is the difference between the length of a given strip and the reference strip.

As discussed in Sections 1 and 2, the current control method in the factory is based on the PI controller, which has three main limitations: expert knowledge is required, parameter tuning is inefficient, and the controller is less adaptive to the rolling conditions [31]. Moreover, hard-to-control factors of the rolling process, such as the vibration of the running mills, the changing temperature of the roller, and the product surface, can affect the control performance and the quality of strips. To satisfactorily capture the impact of process parameters and hard-to-control factors on the flatness of the strip, a new control method is needed for the strip rolling control to avoid redesigning the control logic and system.

Table 1 describes the parameters of the studied production line. The RL controller receives the flatness values (states) and outputs the process parameters (actions). In the strip rolling industry, optimal strips should have flatness values of 0, and the control target is to compute new actions for the mills to produce strips with flatness values as close to zero as possible.

### 4. Methods

#### 4.1. Offline RL for flatness control

According to Sections 1 and 2, offline RL applications have been studied in the steel-making industry and other fields [2,12–15,17–29,32]. As shown in Fig. 2, the left part shows the general framework for the online RL application. Firstly, experts from specific domains should collaborate to build a simulator for the studied industrial process, which requires simplification, discretization, and trial-and-error. Secondly, an RL method is used to train a policy by interacting with the simulator. Thirdly, the learned policy will be evaluated by the simulator. In other words, the learned policy is fully trained and evaluated by the simulator.

However, for online RL in industries, especially in the safety–critical processes (e.g., strip rolling), simplification, discretization, and approximation could generate bias between the simulator and the real process, and modeling is time-consuming. Moreover, a new simulator is needed when studying a new target. Therefore, the learned policy is unsafe to be deployed in real systems. The real process limitations: expert knowledge is required, parameter tuning is inefficient, and the controller is less adaptive to the rolling conditions [31] . Moreover, hard-to-control factors of the rolling process, such as the vibration of the running mills, the changing temperature of the roller, and the product surface, can affect the control performance and the quality of strips. To satisfactorily capture the impact of process parameters and hard-to-control factors on the flatness of the strip, a new control method is needed for the strip rolling control to avoid redesigning the control logic and system.

To address the problems of online RL, offline RL for a safety–critical process is studied in this paper. In Fig. 2, the right part shows an offline RL training framework. Firstly, process data are collected from the server of the control system, the data contain process and quality parameters that will be used for training. Secondly, offline RL is trained using the data without interacting with an environment. Thirdly, a model-based evaluation is adopted to evaluate the policy. Compared to online RL, this framework can directly learn from real industrial data rather than building a simulator. Thus, the bias problem in online RL can be reduced. Since the data are collected from the real process, key information about the process is captured in the data.

#### 4.2. Offline RL

A Markov Decision Process (MDP) can be defined by a tuple \((S, A, R, P, \gamma)\), where \(S\) is the state space, \(A\) is the action space, \(R\) is the reward function, \(P\) is the transition dynamics and \(\gamma\) is the discount. RL is used to train a policy by maximizing the expected cumulative discounted reward in the MDP. \(D = \{(s, a, r, s')\}\) is a dataset of tuples from trajectories. In an actor-critic algorithm, the Bellman operator is used to update the parametric Q-function \(Q_\theta(s, a)\) in policy evaluation, which is shown as follows:

\[ B^\theta Q = r + \gamma P^\theta Q \]  

where \(P^\theta Q = \mathbb{E}_{s' \sim P(s'|s,a)}[Q(s', a')]\). Based on the Q-values, the parametric policy \(\pi_\theta(a|s)\) is improved via policy improvement which maximizes the expected Q-values. Policy evaluation and policy improvement are shown as follows:

\[ \text{Policy Evaluation : } \tilde{Q}^{k+1} \leftarrow \arg\min_{\theta} \mathbb{E}_{s,a,s' \sim D} \left( \left( r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')} \left[ Q_\theta^k(a'|s') \right] \right) - Q_\theta(s,a) \right)^2 \]  

### Table 1

**Data description of states and actions.**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Actions</strong></td>
<td></td>
</tr>
<tr>
<td>Rolling force</td>
<td>7056–7448 kN</td>
</tr>
<tr>
<td>Bending force of work roll</td>
<td>65–104 kN</td>
</tr>
<tr>
<td>Bending force of intermediate roll</td>
<td>77–180 kN</td>
</tr>
<tr>
<td>Roll gap tilting</td>
<td>-0.01–0.18 mm</td>
</tr>
<tr>
<td><strong>States</strong></td>
<td></td>
</tr>
<tr>
<td>Flatness 1</td>
<td>-4.5–21 l</td>
</tr>
<tr>
<td>Flatness 2</td>
<td>-10–15 l</td>
</tr>
<tr>
<td>Flatness 3</td>
<td>-10–7.5 l</td>
</tr>
<tr>
<td>Flatness 4</td>
<td>-20–0.5 l</td>
</tr>
</tbody>
</table>
Policy Improvement: $\hat{\pi}_{k+1} \leftarrow \text{argmax}_\pi \mathbb{E}_{s \sim D, a \sim \pi_k(s | \cdot)} [\hat{Q}_{k+1}(s, a)]$ (4)

In offline settings, the agent can only learn from a static dataset $D$. The agent cannot correct the actions by taking them and obtaining the reward from the environment. OOD actions $a'$ in Eq. (1) are not existing in $D$, which are generated by the learned policy $\hat{x}$ and used to compute $\hat{Q}$, resulting in highly erroneous target Q-values. Meanwhile, the policy is optimized by maximizing the $\hat{Q}$, further exacerbating this problem.

As discussed in Section 2, CQL addresses the problem of Q-values overestimation by learning conservative Q-values in policy evaluation [23] while IQL directly removes OOD actions from policy evaluation [24]. Different from CQL and IQL, which have complex regularizers and terms, BEAR, SPIBB, and TD3-BC constrain the learned policy to force it to approach the behavior policy [17,21,22]. Especially TB3-BC, which has a behavior cloning term on top of TD3, is called a minimalist approach to offline RL [17]. It is simple but effective, and that is potentially significant in the industry.

4.3. Stable Q-learning

In this paper, considering the industrial application, a safe, simple, effective, and adaptive control method is sought after. As discussed in Section 4.1, an offline RL framework will be adapted to the strip rolling by learning from the real data. In this section, considering the uncertainty estimation, a novel offline RL method (stable Q-learning, SQL) is proposed for this framework to obtain a safe and reliable policy for the studied problem.

In RL, deep ensemble achieved the best performance in addressing uncertainty estimation [20]. In offline RL, ensemble Q-functions can be used to address the overestimation of Q-values. Each Q-function is updated separately, due to the random initialization, and pessimistic Q-value estimates can be obtained from the different outputs of Q-functions [33]. In this paper, the purpose aims to improve the generalization of neural networks for estimating stable Q-values. $N$ Q-functions will be updated separately using a minibatch, and the objective of policy evaluation in the proposed method is shown as follows:

$$L(\theta) = E_{s,a',r \sim D} \left[ \left( r(s,a) + \gamma \min_{i=1,\ldots, N} \mathbb{E}_{a' \sim \pi_k(a' | s')} [\hat{Q}^i_{\theta_k}(a' | s')] - Q_{\theta_k}(s,a) \right)^2 \right]$$ (5)

where $\theta$ and $\hat{\theta}$ denote the parameters of the $i^{th}$ Q-network and its target network. Since TD3 was designed to have two Q-functions and two target Q-functions, minimum target Q-values are chosen for policy evaluation for all Q-functions. This paper will explore addressing the uncertainty estimation by adding more Q-functions and corresponding target Q-functions. Because the parametric Q-
functions are approximated by neural networks, they can be fundamentally biased, decreasing the generalization capability of the network.

Compared to the policy improvement in TD3, which only uses the output of one Q-function to optimize the policy, by averaging all the Q-values, the policy improvement for SQL is shown as follows:

$$\pi^{k+1} \leftarrow \text{argmax}_\pi E_{s \sim \mathcal{D}, a \sim \pi^{k+1}} \left[ \hat{Q}^{k+1}_\text{new}(s, a) \right]$$  \hspace{1cm} (6)

$$\hat{Q}^{k+1}_\text{new}(s, a) = \frac{1}{N} \sum_{i=1}^{N} \hat{Q}^{k+1}_i(s, a)$$  \hspace{1cm} (7)

Moreover, a policy regularizer in TD3-BC [17] is added to SQL, pushing the policy towards favoring actions contained in the dataset $D$. The new policy improvement step is:

$$\pi^{k+1} \leftarrow \text{argmax}_\pi E_{s \sim \mathcal{D}, a \sim D} \left[ \frac{\alpha \hat{Q}^{k+1}_\text{new}(s, \pi^k(s))}{\left( \sum_{(s, a_i) \in \mathcal{D}} |Q(s, a_i)/M| \right)} - (\pi^k(s) - a)^2 \right]$$  \hspace{1cm} (8)

where $(\pi^k(s) - a)^2$ is a behavior cloning term, given the dataset of $M$ transitions of $(s, a)$, $\alpha$ is a hyperparameter, and a normalization term based on the average absolute Q-values is used to normalize the Q-values. Algorithm 1 shows the pseudocode of SQL.

```
Algorithm 1

Initialize $N$ Q-function $Q_i$, a policy $\pi_\theta$, create target networks $\tilde{Q}_i, \tilde{\pi}_\theta$, offline RL dataset $D$ generated by behavior policy $\pi_\theta$

for $t = 1, \ldots, T$ do
  Sample a batch $B(s, a, r, s', d)$ from $D$
  for $i = 1, \ldots, N$ do
    Adopt the batch $B$ to compute Q-value for each Q-function
    Update Q-function with gradient descent on objective from Equation (5)
    Store the Q-values for each Q-function
  end for
  Compute $\hat{Q}_\text{new}$ using Q-values of Q-functions (Equation 7)
  Select $s$ and $a$ from batch $B$
  Compute behavior cloning term using $(\pi^k(s) - a)^2$
  Train $\pi_\theta$ with gradient descent on Equation (8)
end for

Save the policy for evaluation
```

4.4. Experiment setting

Due to the flatness adjustment used in the production line [34], the flatness values at different reference points at the same time step have a similar trend. The performance of flatness at each time step can be estimated by averaging the values at different reference points. The reward function is designed as follows:

$$r = -\frac{1}{4} \sum_{i=1}^{4} |s'_{i}|/4$$  \hspace{1cm} (9)

where $s'$ is the $i$th state, and four reference points are studied in the studied strip rolling line. Using this reward function, the learned policy will be trained to generate actions that can produce strips with flatness values close to 0.

MDP is a discrete-time stochastic control process, which can be defined by a tuple $(S, A, R, P, \gamma)$. In the flatness control case, $S$ (state) is the flatness value of strips, which needs to be optimized. $A$ (action) is the process parameters (rolling force, bending force of the work roll and intermediate roll, and roll gap tilting), which are used to control the rolling mill, ensuring the mills produce strips with desired flatness. $R$ (reward function) is used to compute the indicator of the strip quality. $P$ (transitions) represents the model of the process, given action, the state can be obtained using $P$. $\gamma$ is a discount for computing discounted return in RL, it determines the present value of future rewards [8].

Although the bias problem can be reduced by adopting offline RL, it is risky to deploy the policy to the real system for evaluation. In this paper, a black-box data-driven approach is developed to model the real process. A probabilistic neural network (PNN) [35] is trained, receiving the chosen action as input, and outputting the next state as output. To address uncertainty, the output is a parameterized Gaussian distribution, which is conditioned on $s$ and $a$ is shown as follows:

$$f = Pr(s_{t+1}|s, a) = N(\mu_\theta(s, a), \sum_a (s, a))$$  \hspace{1cm} (10)

The mean value and the standard deviation are parameterized by $\theta$. As suggested by [36], the model is learned to predict the difference to the current state, $\Delta s_{t+1} = f(s, a)$.  

226
In this paper, the proposed method (SQL) will be compared to the SOTA methods (BC, CQL, IQL, TD3-BC). First, key hyperparameters of the proposed SQL method are studied to find the optimal setting, including the ensemble number $N$ of Q-functions, and $\alpha$ in the policy improvement step. Then, SQL will be compared with SOTA. Finally, an experiment of the proposed method will be analyzed and compared with the existing method used in the factory.

5. Results

5.1. Analysis of SQL

This section investigates the parameter sensitivity of the proposed method (SQL). The number of Q-functions and $\alpha$ in policy improvement were studied separately by grid search. Fig. 3(a) presents the results of the analysis of the number of Q-functions, where values of {2, 4, 6, 8, 10} were tested, and $\alpha$ was fixed at 2.5, as recommended in [17]. The performance of the models exhibits a similar trend as the number of Q-functions was increased from 2 to 10, with a decrease noted around the 20th timestep. Nonetheless, an increase in reward was observed when the number of Q-functions was raised from 2 (blue line) to 4 (orange line) during the first 20 timesteps. A temporary dip in reward occurred between the 21st and the 30th timestep, but the reward thereafter increased, yielding the highest final reward among all models. In contrast, as the number of Q-functions was further increased from 4 to 10, an overall decrease was observed. Based on the above findings, the model with 4 Q-functions was deemed to achieve the best performance.

The effect of $\alpha$ on the performance of SQL was analyzed in Fig. 3(b). The models were evaluated using 4 Q-functions, and varying values of $\alpha$, including {1.5, 2.5, 3.5, 5, 10, 20, 30}. Results show that an increase in the values of $\alpha$ leads to reductions in fluctuations in reward. Particularly, when $\alpha$ was set to 20, a significant reduction in fluctuation was observed. The model with $\alpha = 20$ also achieved the highest final reward of 60. The key hyperparameters of the proposed method that achieved the optimal performance are shown in Table 2.

5.2. Comparison of the SOTA methods

In this section, the proposed method (SQL) was compared with SOTA methods (BC, CQL, TB3-BC, IQL) in terms of reward, the comparison results are shown in Fig. 4. The reward was designed to assess the overall flatness. First, the absolute values of flatness at different points (state 1 to 4) were computed. Then, the mean value of the absolute values was obtained. Finally, the inverse of the mean value was used as the reward. For the studied case, the optimal flatness is 0, and thus, the flatness is expected to be as close to 0 as possible, leading to high reward values. In RL, the reward is used as an indicator in training, and the high-quality strips will have high reward values.

Although BC has an uptrend from the 10th to the 20th step, IQL has an uptrend from the 35th to the 50th step, they have an overall downtrend during the training process, suggesting that they are not effective in addressing the problem. CQL exhibits a steady increase in reward from −92 to −78 with a decreasing shadow area. Compared to CQL, TD3-BC has a higher final reward of −75, but it has a huge fluctuation during the training process. The reward decreased from the 10th to the 20th step and then increased, before the 40th step, the large shadow area indicates the performance is unsteady. Notably, the shaded area of TD3-BC decreased significantly from the 40th step, meaning that the standard deviation decreased when using different randoms to repeat the training. For SOTA methods, CQL has an overall steady performance, but the final reward is lower than TD3-BC. TD3-BC has a huge fluctuation during the training process, but the final reward is higher, and the shadow area is smaller at the end of training.

The results presented in Fig. 4 demonstrate the superior performance of SQL compared to SOTA methods. Firstly, SQL successfully mitigated the decrease in reward observed in TD3-BC from the 10th to the 20th step, and reduced the uncertainty, as evidenced by the smaller shadow area. Secondly, from the 1st to the 20th step, SQL exhibits a comparable trend in reward to CQL with a higher reward.

![Fig. 3. Hyperparameters analysis of SQL.](image-url)
throughout the process, resulting in a final reward of −62. In contrast, CQL has a lower uncertainty, as indicated by its smaller shadow area, but the final reward of −78 is significantly lower than that of SQL (−62). The above findings support the conclusion that SQL is the optimal control method for the studied case.

5.3. Experiment analysis

In this section, simulation results of the existing method (PI) commonly used in the factory and the proposed SQL method were analyzed. For the states, either positive or negative values are acceptable, and the optimal value is 0. In other words, the optimization goal is to produce high-quality strips with flatness values as close to 0 as possible. The results are shown in Fig. 5. For state 1, 80 % from the 1st to 800th step) of the flatness values are negative, SQL demonstrated a remarkable ability to improve the flatness by pushing the values closer to 0. Similar performance was observed in state 4. In state 2 and state 3, over 90 % of the flatness values are positive, and SQL successfully reduced the values, demonstrating its ability to improve strip quality. Results in Fig. 5 suggest that SQL can always push the values towards 0, regardless of their initial values being negative or positive. The above findings indicate that SQL is a promising tool for addressing the studied problem.

Another evaluation of the performance of the proposed method (SQL) and the existing method (PI) was performed by computing the average and standard deviation (STD) of absolute flatness values in each state. As per the established quality standards, high-quality steel strips should have flatness values close to 0. Fig. 6 shows the comparison of the average flatness of the two methods in four states. Results indicate that the average flatness values generated by SQL in three states are lower than those produced by PI. The reductions in average flatness are 8.48 %, 5.48 %, and 5.13 % for state 1, 2, and 3, respectively. Even though the flatness of state 4 produced by SQL increased by 0.27 %, it is still lower than that of state 3 (1.331). The balance between the overall and individual flatness was obtained. Fig. 7 shows the STD of the flatness in four states. The results indicate that the STD of the flatness values produced by SQL in all four states is lower than that of PI, with reductions of 7.69 %, 6.55 %, 10.53 %, and 2.66 % for states 1 to 4, respectively. A lower STD indicates improved steadiness in the performance of the strips, leading to smoother steel strips. In conclusion, the results of the evaluation of the average and STD of the flatness values demonstrate that the proposed method (SQL)

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes in hidden layer 1</td>
<td>256</td>
</tr>
<tr>
<td>Number of nodes in hidden layer 2</td>
<td>256</td>
</tr>
<tr>
<td>Policy learning rate</td>
<td>3e−4</td>
</tr>
<tr>
<td>Q-function learning rate</td>
<td>3e−4</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
</tr>
<tr>
<td>Discount</td>
<td>0.99</td>
</tr>
<tr>
<td>Batch size</td>
<td>256</td>
</tr>
<tr>
<td>Exploration noise</td>
<td>0.1</td>
</tr>
<tr>
<td>Policy noise</td>
<td>0.2</td>
</tr>
<tr>
<td>Number of Q-functions</td>
<td>4</td>
</tr>
<tr>
<td>Alpha</td>
<td>20</td>
</tr>
<tr>
<td>Target network update rate</td>
<td>0.005</td>
</tr>
<tr>
<td>Evaluation frequency</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2

Key hyperparameters of the proposed method.

Fig. 4. Comparison of SQL and SOTA methods.
produces steel strips with flatness values closer to 0 and improved steadiness compared to the existing method (PI). The purpose of this paper is to propose a novel and adaptive method for the changeable manufacturing condition. The proposed method and findings are important in safety-critical applications and are expected to improve the quality standards and increase the exporting power of the factory.

Fig. 5. Experiment of the proposed and the existing methods.

Fig. 6. Average flatness of the strips produced by PI and SQL.
6. Conclusion and future work

This paper studied a novel offline RL method for flatness control in the strip rolling industry. Using the pre-collected dataset for training rather than an approximated environment, the proposed method reduced the bias of the environment. Q-ensemble was adopted to address the problem of uncertainty estimates of Q-values. A behavior cloning term was added to the policy improvement step to push the policy towards favoring actions contained in the dataset. From the comparison, the reward of the proposed method increased without sharp drops and has the highest final value, it can be concluded that it can learn a policy more stably and effectively than the SOTA methods. From the experiment, the results showed that the proposed method can generate strips with flatness with more values close to 0 than the traditional control method, the lower STD means the performance is more stable.

Next steps of our work will focus on the reality gap between the experiment and the real process. Although the simulator for training is avoided, the offline RL used in this paper highly relies on real data, which means that the quality of the data can affect the reliability of the policy. Before being deployed to the system, two plans are available to improve the policy. Firstly, to collect more data from the real system, and to make the policy fully explored to find optimal solutions to the control problem. Secondly, an additional step (e.g., transfer learning) can be adopted to optimize the policy.

CRediT authorship contribution statement

Jifei Deng: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – original draft. Seppo Sierla: Formal analysis, Validation, Writing – review & editing. Jie Sun: Data curation, Writing – review & editing. Valeriy Vyatkin: Formal analysis, Investigation, Supervision, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgements

This work was supported by China Scholarship Council (No. 202006080008), the National Natural Science Foundation of China (Grant Nos. 52074085 and U21A20117), the Fundamental Research Funds for the Central Universities (Grant No. N2004010), and the LiaoNing Revitalization Talents Program (XLYC1907065).

References
