Battery Model Calibration with Deep Reinforcement Learning

Ajaykumar Unagar **Tian Yuan** **Manuel Arias-Chao** **Olga Fink**
ETH Zurich
aunagar@ethz.ch {tian,arias,fink}@ibi.baug.ethz.ch

Abstract

Lithium-Ion (Li-I) batteries have recently become pervasive and are used in many physical assets. To enable a good prediction of the end of discharge of batteries, detailed electrochemical Li-I battery models have been developed. Their parameters are typically calibrated before they are taken into operation and are typically not re-calibrated during operation. However, since battery performance is affected by aging, the reality gap between the computational battery models and the real physical systems leads to inaccurate predictions. A supervised machine learning algorithm would require an extensive representative training dataset mapping the observation to the ground truth calibration parameters. This may be infeasible for many practical applications. In this paper, we implement a Reinforcement Learning-based framework for reliably and efficiently inferring calibration parameters of battery models. The framework enables real-time inference of the computational model parameters in order to compensate the reality-gap from the observations. Most importantly, the proposed methodology **does not need any labeled data samples**, (samples of observations and the ground truth calibration parameters). Furthermore, the framework does not require any information on the underlying physical model.The experimental results demonstrate that the proposed methodology is capable of inferring the model parameters with high accuracy and high robustness. While the achieved results are comparable to those obtained with supervised machine learning, they do not rely on the ground truth information during training.

1 Introduction

Recent advancements in Li-I batteries have increased their usage in various applications ranging from electric vehicles to drones and space exploration. Particularly for autonomous systems, it is essential to plan the missions reliably which requires an accurate prediction of the End-of-Discharge (EOD) time for the batteries. However, the currently available battery models [1] suffer from an increasing uncertainty in their EOD predictions over time. This is mainly due to the degradation processes in the battery. The relationship between battery age and the discharge time is non-linear. Hence, sophisticated modeling techniques are required to estimate Battery Degradation Parameters that would be required for the EOD time prediction.

Previous work on Battery aging [2–5] have focused on the understanding of the aging process. Some data-driven methods based on empirical, probabilistic, and learning-based models [6–9] have been proposed for Battery End-of-Life (EOL) prediction or State of Health (SOH) determination. However, these methods suffer from a strong dependence on labeled data, or are computationally expensive. These shortcomings limit their applicability in many real-world problems.

Battery aging models enable to track the aging process over time. The model parameters are then inferred from the empirical observations. This is also referred to as model calibration. Previous works have focused on traditional approaches like Kalman Filter [9, 10], Particle Filter [6], or explicit degradation models (Model-Based Prognostics) [7]. Kalman Filter and Particle Filter approaches do not require an underlying model, however suffer from a high computational burden during application time. Model-Based approaches assume the underlying model of the aging process. In this work, we solve the Battery Model Calibration process using Reinforcement Learning (RL) method, which can work in real-time and does not require an underlying model.

Recent developments in model-free Reinforcement Learning have been applied to various control problems in Robotics [11–14], Water Systems management [15], Computational Biology [16], and AutoML[17]. Model-free RL methods have multiple advantages over traditional methods: (1) RL framework is highly general, the agent can learn to solve tasks without any knowledge of the underlying model. In our case, the agent can learn to calibrate the underlying parameters without any labeled data. (2) The policy learned via RL is robust to model uncertainty. (3) RL methods provide almost real-time performance since it only requires to evaluate the learned policy. Hence, Reinforcement Learning is a compelling alternative to other data-driven methods for Battery Model Calibration.

In this work, we propose to define the Battery Calibration problem as a tracking problem defined by a Markov Decision Process (MDP) and solve it with the Lyapunov-based Maximum-Entropy Reinforcement Learning algorithm. Specifically, we use Lyapunov-based Actor-Critic (LAC) method [18, 19] to provide stable tracking of the parameters. We use the Battery Model from the NASA prognostic model library [20, 21] to simulate our RL environment. It is important to clarify that this library models the physical process of the discharge but not the battery aging process, which is our main focus here. To the best of our knowledge, ours is the first method applying Reinforcement Learning for the battery model calibration [1].

2 Related Work

Model Calibration is essential in many engineering applications relying on simulations. As most of the systems undergo changes over time, physical model parameters need to be re-calibrated. If possible, this calibration process can be done offline by applying a reference loading condition and correlating the observed output with the expected one. However, in situations where it is not possible to calibrate the systems offline, online calibration is crucial.

For operational purposes of Batteries, it is important to have an accurate estimate of the EOD time. There are accurate discharge models for Li-I batteries [20]. These models work based on the physical principles of the discharge process. However, such models are not able to estimate the degradation parameters of the Battery that change over time, and hence battery model real-time calibration becomes essential.

There are three primary ways to assess Battery degradation parameters. (1) Direct estimation from observations (2) Bayesian Tracking principle (3) Model-Based Prognostics based on an explicit aging model.

Firstly, in direct estimation methods, observations are used to learn the mapping from the battery outputs to the degradation parameters. For, example authors in [8] used Support Vector Machine (SVM) model to learn this mapping. There are approaches that use Structured Neural Networks (SNN) [9] to exploit knowledge of the degradation process. Such approaches show promising results in certain scenarios where we can obtain paired samples of observations and degradation parameters.

Secondly, Methods based on Unscented Kalman Filter (UKF) [10] tracks the internal battery state to reduce the observation gap between predicted and actual output. Similar to this, Extended Kalman Filter (EKF) [9] also tracks the internal state of the battery but with a different model for the discharge process. Such tracking algorithms provide model-agnostic parameter estimation. However, these methods are computationally expensive at the application time and suffer from a drift in parameter tracking.

Thirdly, model-based prognostic methods assume an underlying degradation model for the aging parameters as the function of its usage. Authors in [7] used system identification techniques to estimate the parameters of the degradation model. These techniques provide accurate estimates

[1] In this work we use “Model Calibration” and “Degradation Parameter Estimation” terms interchangeably

as long as the physical degradation process follows the assumed model. Reinforcement Learning provides an alternative solution to these approaches while relaxing some of the constraints. Especially, in the scenarios where labeled data is not available, RL can learn from the observations and infer the model parameters.

With deep function approximators and sophisticated exploration techniques, Reinforcement Learning methods have made some significant progress in recent times. In our work, we focus on model-free RL methods based on Actor-Critic (AC) approach [22, 23]. Actor-Critic methods provide a framework for generalized policy iteration algorithms in which two networks (Actor and Critic) are updated continuously. Especially, Maximum Entropy-based RL formulation such as Soft Actor-Critic (SAC) [24, 25] algorithms have shown good performance in different applications [26, 27]. Chao et al. [28] applied a variant of the Maximum Entropy-based RL algorithm for model calibration of Turbofan Engines. In this work, the authors proposed the Lyapunov-Based Critic which to some extent provides stability guarantees which are essential for non-linear dynamical systems. We follow this approach here, by formulating Battery Model Calibration as the tracking problem.

3 Method

3.1 Battery Discharge Model

We follow Li-I Battery Model from NASA Prognostic Model Library [20, 21]. It captures significant electro-chemical processes of the discharge and also models the effect of aging in terms of degradation parameters. However, the model needs to be provided with degradation parameters for accurate estimation of EOD time. The battery state is modeled by seven parameters, and changes over time as a function of input load and degradation state. Here, we just denote the state mathematically and refer the readers to the original paper [20] for the physical meaning of these parameters.

$$\mathbf{x}(t) = [q_{s,p} \;\; q_{b,p} \;\; q_{b,n} \;\; q_{s,n} \;\; V_o^{'} \;\; V_{\eta,p}^{'} \;\; V_{\eta,n}^{'}] \tag{1}$$

The input load at time t is $\mathbf{w}(t)$, and the model predicts the voltage $\mathbf{y}(t) = V$.

There are two main degradation parameters: (a) q^{max} captures the decrease in active Lithium ions, and (b) R_o captures the increase in internal resistance. These parameters are essential for the model dynamics, that are defined as follows:

$$\begin{aligned} \mathbf{x}(t+1) &= f(\mathbf{x}(t), \mathbf{w}(t), q^{max}, R_o) \\ \mathbf{y}(t+1) &= f(\mathbf{x}(t+1), q^{max}, R_o) \end{aligned} \tag{2}$$

Without any knowledge of the battery age, degradation parameters are initialized to "perfect battery" condition values, which are $q^{max} = 7600$, and $R_o = 0.117215$. Using these parameters, the model can estimate the initial state $\mathbf{x}(0)$. As the battery ages, q^{max} decreases while R_o increases. We try to infer these parameters by solving the state-tracking problem using RL. Here, we use the physics-based battery model as our Reinforcement Learning environment. However, in cases where such a model is difficult to obtain, it can be replaced by function approximators or surrogate models.

3.2 Markov Decision Process and Reinforcement Learning

In this paper, we focus on the battery state tracking task which is modeled by a Markov decision process (MDP). An MDP can be described as a tuple, (S, A, c, P, ρ), where S is the set of states, A is the set of actions, $c(s, a) \in [0, \infty)$ is the cost function,and $P(s'|s, a)$ is the transition probability function, and $\rho(s)$ is the starting state distribution. $\pi(a|s)$ is a policy denoting the probability of selecting action a in state s. The state of a system at time t is given by the state $s_t \in \mathcal{S} \subseteq \mathbb{R}^n$, where $\mathcal{S}$ denotes the state space. For our tracking strategy, we define the state at time t as $s_t = [\hat{\mathbf{x}}_t, \mathbf{x}_{t+1}, \mathbf{u}_{t+1}]$. Where, $\hat{\mathbf{x}}_t$ is the model predicted battery internal state and $\mathbf{x}_{t+1}$ is the real Battery state as described in eq.(1). The agent(calibrator) then controls the system's degradation parameters as an action $a_t \in \mathcal{A} \subseteq \mathbb{R}^m$ (e.g, $a_t = q^{max}$ or R_o) according to the policy $\pi(a_t|s_t)$, and resulting in the next state s_{t+1}. The transition of the state is computed by the battery model. The cost function $c(s_t, a_t) = ||\hat{\mathbf{x}}_{t+1} - \mathbf{x}_{t+1}||$ is a feedback signal to the agent.

Our RL algorithm aims to find a policy π which minimizes $J_c(\pi) \doteq \mathbb{E}_{\tau\sim\pi}\sum_{t=0}^{\infty}\gamma^t c(s_t,a_t)$. Here, $\gamma \in [0,1)$ is the discount factor, τ denotes a trajectory ($\tau = (s_0, a_0, s_1, ...)$), and $\tau \sim \pi$ is shorthand for indicating that the distribution over trajectories depends on π: $s_0 \sim \rho$, $a_t \sim \pi(\cdot|s_t)$, $s_{t+1} \sim P(\cdot|s_t,a_t)$.

3.3 Lyapunov-based actor-critic

Since we target the state tracking task, we adopted the Lyapunov-based Actor-Critic (LAC). LAC has been proved to be able to learn policies with guaranteed stability, which is more capable and favorable of handling uncertainties compared to those without such guarantees in nonlinear control problems. Besides, LAC is based on the actor-critic maximum entropy framework [24], which can enhance the exploration of the policy and has been shown to substantially improve the robustness of the learned policy [24]. LAC contains a Lyapunov critic function and a policy network. The Lyapunov critic plays an important role in both stability analysis and the learning of the actor. In view of the requirements in the LAC, we parameterized the Lyapunov candidate function as L_c^{ϕ} and construct the Lyapunov candidate by $L(s,a) = c + \max_{a'}\gamma L(s',a')$. During training, L_c^{ϕ} is updated to minimize the following objective function,

$$J(L_c) = \mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\frac{1}{2}(L_c(s,a) - L_c^{target}(s,a))^2\right] \tag{3}$$

where L_{target} is the approximation target related to the chosen Lyapunov candidate and D is the set of collected transition pairs. The approximation target is given by,

$$L_c^{target} = c + \max_{a'}\gamma L_c^{\phi}(s',a') \tag{4}$$

Then, based on the maximum entropy actor-critic framework, it uses the Lyapunov critic function in the policy gradient formulation. First, the objective of the policy network is summarized as follows:

$$J(\pi) = \mathbb{E}_{\mathcal{D}}[\beta[\log(\pi_\theta(f_\theta(\epsilon,s)|s))] + \lambda(L_c((s',f_\theta(\epsilon,s')) - L_c(s,a) + \alpha_3 c)] \tag{5}$$

where π_θ is the policy parameterized by a neural network f_θ, and ϵ is an input vector consisting of Gaussian noise. The $\mathcal{D} \doteq \{(s,a,s',c)\}$ is the replay buffer for storage of the MDP tuples. In the above objective, β and γ are positive Lagrange multipliers that control the relative importance of policy entropy versus the stability guarantee. And α_3 is a constant for lyapunov energy decreasing objective. As in [29], the entropy of policy is expected to remain above the target entropy $\mathcal{H}_t$. The values of β and λ are adjusted through gradient method, thereby maximizing the objective:

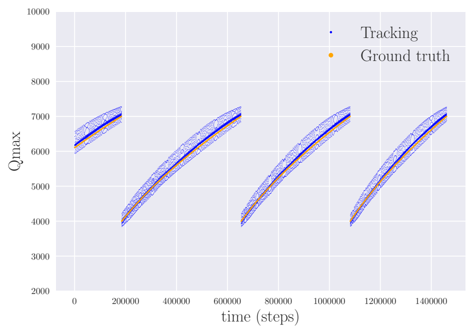
$$J(\beta) = \beta\mathbb{E}_{(s,a)\sim\mathcal{D}}[\log(\pi_\theta(a|s)) + \mathcal{H}_t] \tag{6}$$

and the λ is adjusted by the gradient method, thus maximizing the objective:

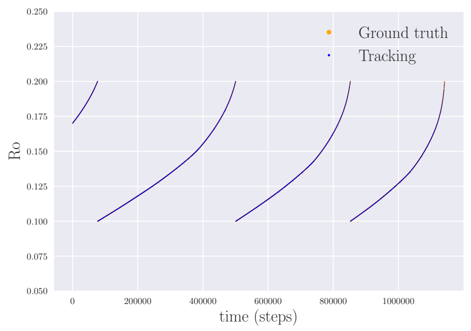
$$J(\lambda) = \lambda(L_c((s',f_\theta(\epsilon,s')) - L_c(s,a) + \alpha_3 c) \tag{7}$$

3.4 Direct Mapping

We also consider a simple fully connected neural network to learn a direct mapping from state s_t to the degradation parameters (a_t in RL setting). However, this is a much simpler problem since it learns from the labelled pairs of "states" and "degradation parameters", which might not be easy to obtain in certain scenario (for every pairs of states and the underlying degradation, the assets need to be measured manually, which is considerable time consuming). Furthermore, the training dataset is required to be representative and cover all the different combinations to enable a reliable ML model. Also, in this setting state s_t is formulated as $s_t = [\mathbf{x}_t, \mathbf{x}_{t+1}, \mathbf{u}_{t+1}]$, and hence both x_t and x_{t+1} are real states. On the other hand, with RL parameter tracking formulation, we do not need such labelled training samples since an actor learns purely based on the rewards observed. Hence, we treat the results obtained with this supervised learning setup as an upper bound for our RL framework performance.



(a) q^{max} tracking



(b) R_o tracking

Figure 1: Battery model parameter calibration with RL

4 Experiments

4.1 Datasets and Model architecture

We use the battery model from NASA Prognostic model library [21] to generate simulated data for our training process. As discussed earlier, we have two degradation parameters to calibrate in a battery model. We generate 5500 trajectories for each degradation parameter by varying the parameter value and load condition within a certain range. We consider each parameter degradation independently, hence, while varying one parameter, we keep the other one constant. Following the approach of [7], we keep degradation parameters constant for a given discharge cycle. Also, we are assuming here that we the future battery load conditions are known for a discharge trajectory.

We use a fully connected neural network as a function approximator for our actor f_θ and Lyapunov critic, L_c. Both networks have 3 fully connected layers with 256 neurons each and LeakyReLU [30] activation functions. For the policy network, we predict two values, mean and std, for each action. After this step, we use the squashed Gaussian policy [24] to sample from the distribution. To ensure that the Lyapunov values are positive, we use the sum-of-square of the final layer activations of the Lyapunov network as Lyapunov values. The parameters β and λ are also updated by using the loss defined in Eq.(6) and Eq.(7). We use Adam optimizer with the learning rate 5x10^{-4}.

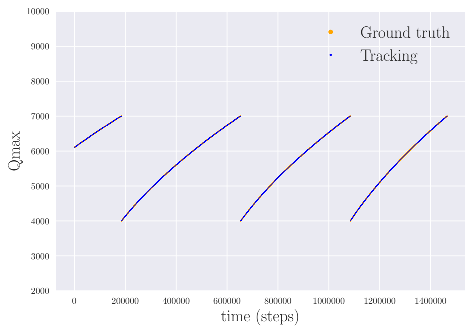
For the Direct Mapping experiment, we use the same architecture as the policy network with the difference that only one output per action is learned since standard deviation is not required. We use the same optimizer and hyper-parameters as described above.

4.2 Results

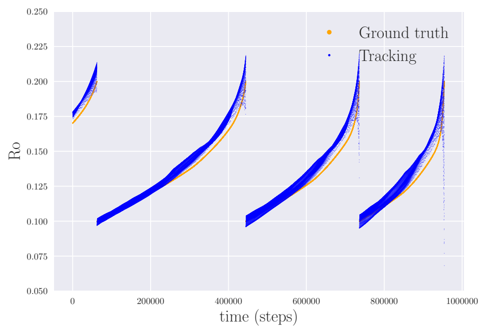
For model evaluation, we divide the dataset randomly in 70% training and 30% testing data We train RL model for one million steps while the direct mapping model has been trained for 30 epochs.

4.2.1 RL Calibration

We compare the inference accuracy of our RL-based approach with the direct mapping approach. As discussed earlier, direct mapping takes the labeled training pairs and hence, is expected to perform better than the indirect inference RL method. Inference trajectories of the degradation parameters are shown in Fig. (1). Even though there is some variance in the inference of the parameter q_{max}, we can see that most of the points are close to the true parameters, while in the case of R_o, tracking works perfectly. We can observe bias in the direct mapping for R_o parameter Fig.(2b). The bias is much worse than the one in the indirect RL inference. Hence, the performance of the RL method is either comparable or better as the direct mapping algorithm. This competitive performance is achieved while purely learning from the interactions and without any access to the ground truth.



(a) q^{max} tracking



(b) R_o tracking

Figure 2: Direct Mapping of Battery state to degradation parameters

5 Conclusion

In this paper, we present a new approach for the Battery model-calibration as a tracking problem. We solve this tracking problem by Lyapunov-based maximum entropy RL framework and show that the inference of this model provides accurate estimates of the model parameters. The performance of the RL framework is comparable to the supervised learning algorithm which requires labeled pairs of state observations and degradation parameters. The indirect inference as performed by the RL algorithm is a much much harder learning problem. Therefore, we propose a valid alternative for the scenarios where training data is limited.

In the future, this method can be extended in the scenarios where the internal state of the model is not easy to obtain. For such cases, we can formulate the problem as a problem of tracking the output voltage. This is a much harder problem compared to the one analyzed here since RL has to learn the internal discharge model along with the degradation process purely from the observed rewards. Furthermore, in the current case study, we observed some variance in the inference of the parameter q^{max}. To tackle this, we can add a consistency penalty loss if two consecutive actions differ significantly.

References

[1] H. Hinz, "Comparison of lithium-ion battery models for simulating storage systems in distributed power generation," *Inventions*, vol. 41, no. 4, 2019.

[2] G. Ning and B. N. Popov, "Cycle life modeling of lithium-ion batteries," *Journal of The Electrochemical Society*, vol. 151, no. 10, p. A1584, 2004.

[3] G. Ning, R. E. White, and B. N. Popov, "A generalized cycle life model of rechargeable li-ion batteries," *Electrochimica acta*, vol. 51, no. 10, pp. 2012–2022, 2006.

[4] S. B. Peterson, J. Apt, and J. Whitacre, "Lithium-ion battery cell degradation resulting from realistic vehicle and vehicle-to-grid utilization," *Journal of Power Sources*, vol. 195, no. 8, pp. 2385–2392, 2010.

[5] A. Barré, B. Deguilhem, S. Grolleau, M. Gérard, F. Suard, and D. Riu, "A review on lithium-ion battery ageing mechanisms and estimations for automotive applications," *Journal of Power Sources*, vol. 241, pp. 680–689, 2013.

[6] B. Saha and K. Goebel, "Modeling li-ion battery capacity depletion in a particle filtering framework," in *Proceedings of the annual conference of the prognostics and health management society*. San Diego, CA, 2009, pp. 2909–2924.

[7] M. Daigle and C. S. Kulkarni, "End-of-discharge and end-of-life prediction in lithium-ion batteries with electrochemistry-based aging models," in *AIAA Infotech@ Aerospace*, 2016, p. 2132.

[8] A. Nuhic, J. Bergdolt, B. Spier, M. Buchholz, and K. Dietmayer, “Battery health monitoring and degradation prognosis in fleet management systems,” *World Electric Vehicle Journal*, vol. 9, no. 3, p. 39, 2018.

[9] D. Andre, A. Nuhic, T. Soczka-Guth, and D. U. Sauer, “Comparative study of a structured neural network and an extended kalman filter for state of health determination of lithium-ion batteries in hybrid electricvehicles,” *Engineering Applications of Artificial Intelligence*, vol. 26, no. 3, pp. 951–961, 2013.

[10] B. Bole, C. S. Kulkarni, and M. Daigle, “Adaptation of an electrochemistry-based li-ion battery model to account for deterioration observed under randomized use,” SGT, Inc. Moffett Field United States, Tech. Rep., 2014.

[11] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, “Reinforcement learning for control: Performance, stability, and deep approximators,” *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[12] V. Kumar, A. Gupta, E. Todorov, and S. Levine, “Learning dexterous manipulation policies from experience and imitation,” *arXiv preprint arXiv:1611.05095*, 2016.

[13] M. P. Deisenroth, C. E. Rasmussen, and D. Fox, “Learning to control a low-cost manipulator using data-efficient reinforcement learning,” *Robotics: Science and Systems VII*, pp. 57–64, 2011.

[14] M. Han, Y. Tian, L. Zhang, J. Wang, and W. Pan, “H_∞ model-free reinforcement learning with robust stability guarantee,” *arXiv preprint arXiv:1911.02875*, 2019.

[15] B. Bhattacharya, A. Lobbrecht, and D. Solomatine, “Neural networks and reinforcement learning in control of water systems,” *Journal of Water Resources Planning and Management*, vol. 129, no. 6, pp. 458–465, 2003.

[16] N. J. Treloar, A. J. Fedorec, B. Ingalls, and C. P. Barnes, “Deep reinforcement learning for the control of microbial co-cultures in bioreactors,” *PLOS Computational Biology*, vol. 16, no. 4, p. e1007783, 2020.

[17] Y. Tian, Q. Wang, Z. Huang, W. Li, D. Dai, M. Yang, J. Wang, and O. Fink, “Off-policy reinforcement learning for efficient and effective gan architecture search,” *arXiv preprint arXiv:2007.09180*, 2020.

[18] M. Han, L. Zhang, J. Wang, and W. Pan, “Actor-critic reinforcement learning for control with stability guarantee,” *arXiv preprint arXiv:2004.14288*, 2020.

[19] Y. Tian, “Model free reinforcement learning with stability guarantee,” 2019.

[20] M. J. Daigle and C. S. Kulkarni, “Electrochemistry-based battery modeling for prognostics,” 2013.

[21] “https://github.com/nasa/prognosticsmodellibrary.”

[22] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[23] V. R. Konda and J. N. Tsitsiklis, “Actor-critic algorithms,” in *Advances in neural information processing systems*, 2000, pp. 1008–1014.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” *arXiv preprint arXiv:1801.01290*, 2018.

[25] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, “Soft actor-critic algorithms and applications,” *arXiv preprint arXiv:1812.05905*, 2018.

[26] J. Wu, Z. Wei, W. Li, Y. Wang, Y. Li, and D. Sauer, “Battery thermal-and health-constrained energy management for hybrid electric bus based on soft actor-critic drl algorithm,” *IEEE Transactions on Industrial Informatics*, 2020.

[27] D. Li, X. Li, J. Wang, and P. Li, “Video recommendation with multi-gate mixture of experts soft actor critic,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 1553–1556.

[28] M. A. Chao, Y. Tian, C. Kulkarni, K. Goebel, and O. Fink, “Real-time model calibration with deep reinforcement learning,” *arXiv preprint arXiv:2006.04001*, 2020.

[29] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, “Soft actor-critic algorithms and applications,” *arXiv preprint arXiv:1812.05905*, 2018.

[30] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical evaluation of rectified activations in convolutional network,” *arXiv preprint arXiv:1505.00853*, 2015.