Convolutional Neural Network as a Steady-state Detector for Real-time Optimization

Vinh Phuc Bui Nguyen, Jose Matias and Johannes Jäschke
Chemical Engineering Department
Norwegian University of Science and Technology
Trondheim, Norway
Email: johannes.jaschke@ntnu.no

***Abstract*— A new online steady-state detection (SSD) method based on convolutional neural networks has been developed. It mimics how humans carry out SSD by visually investigating measurement plots. The method is tested on synthetic and real-life data. We also investigate how it interacts with steady-state real-time optimization and affects its performance on a digital twin. The proposed method has comparable performance to traditional SSD methods; however, it is more intuitive to tune.**

I. INTRODUCTION

Maximizing operating profit has always been a critical issue to chemical plants, especially as increasingly higher standards are imposed on the manufacturing and the products. Among many process optimization techniques developed to address this problem, steady-state Real-Time Optimization (SRTO) is the most widespread in industry [1]. In SRTO, the profit maximization problem is described as following:

$$\begin{aligned} \min_{u} \quad & J(u,x,d) \\ \text{s.t.:} \quad & f(u,x,d)=0, \quad h(u,x,d)\leq 0 \end{aligned} \tag{1}$$

where x represents states of the plant, d represents disturbances, J is the function that we wish to optimize, $f(u,x,d)=0$ is the steady-state model of the plant, and $h(u,x,d)$ are the operating constraints. SRTO solves the problem in (1) to obtain inputs $u^\star$ that optimize the plant performance. Usually, values of disturbances d are unknown and must be estimated from steady-state plant measurements. If transient data is used, the d estimates are prone to be incorrect, and the computed $u^\star$ is sub-optimal or even worsens the performance [2]. Thus, SRTO needs a steady-state detector (SSD) to indicate when the plant is at steady-state and trigger the estimation and optimization steps.

SSD has a significant impact on the performance of SRTO [1], and although there are many SSD methods, most of them rely on numerical statistical tests. Refer to Section IV for more details on such methods. Humans, in contrast, naturally use the graphical method to carry out SSD. We investigate measurement plots with our vision, infer whether they represent steady-state or transient measurements, and determine the status of the plant. As the graphical method can have advantages over numerical methods in some tasks [3], in this paper, we will explore the possibility of carrying out SSD via image classification.

We will employ Convolutional Neural Networks (CNNs) to classify the measurement plots in two categories: steady-state or transient. CNNs were chosen due to their excellent performance in image classification. We test our method in three case studies, where its performance is compared to two traditional SSD methods. In the first case study, we use synthetic data representing typical plant operating conditions. The goal is to assess how the SSD methods behave in face of different disturbances (e.g. ramp disturbances, step disturbances, etc.) as well as identify patterns in the misidentification of the plant condition. In the second case study, we use experimentally measured flowrate data from a lab-scale rig for showing the impact of real data on our method. The third case study is used for illustrating that CNN-SSD can be indeed combined with an SRTO loop. We apply them to a digital twin of the lab-scale rig, where our SSD method is used for triggering the economic optimization. The final case study is used for assessing how our method impacts the overall performance of SRTO.

II. CONVOLUTIONAL NEURAL NETWORKS

CNN is a class of neural networks that are commonly applied in computer vision. They usually have many layers with a large number of parameters to be learned, so their training requires enormous amounts of data and resources. Instead of training a new CNN for a desired purpose, it is a common practice to modify and reuse an established CNN with proven performance [4]. There are many such CNNs, and we select the VGG-16 model shown in Fig. 1.

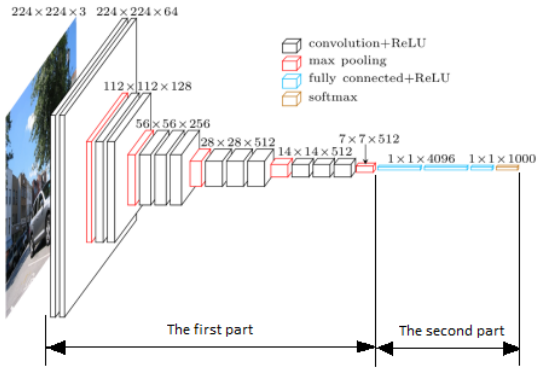

Fig. 1: The architecture of VGG-16. Adapted from [5]

As indicated in the figure, VGG-16 consists of two parts. When an image is shown to VGG-16, the first part generates a "feature vector" corresponding to that specific image. The second part then takes this vector as its input and returns the image's category. It is this second part that we should modify and retrain when reusing VGG-16 in our specific application. The first part is kept untouched. Neither its architecture nor the values of its parameters change.

III. CNN FOR STEADY-STATE DETECTION

For our new application, we replace the second part, the image feature classification step, by the k-Means clustering algorithm. We call this new CNN architecture VGG-16k and will use it in our study as a steady-state detector. Simply speaking, the function of the k-Means algorithm is to partition a number of data points, which are in the form of vectors, into k clusters (with k specified by users). Each data point is assigned to only one cluster. Cluster assignments are based on the similarity between data points, which is quantified as the Euclidean distance between two points. Thus, the smaller the Euclidean distance of two data points, the higher the chance that they will be in the same cluster. One problem with the k-Means algorithm is that the number of clusters k has to be specified by users in advance. In our case, setting $k = 2$ is usually sufficient as we have 2 types of data (steady-state and transient). If a system has different forms of transient behaviors, e.g. great fluctuation vs. ramp response, k should be larger (see Case Study 1).

A. VGG-16k IMPLEMENTATION

Fig. 2 represents how to carry out SSD on a single-measurement system using VGG-16k when $k = 2$. First, we "calibrate" VGG-16k offline with samples of steady-state (SS) and transient (TS) plots (Fig. 2-left). VGG-16k will convert the sample plots into feature vectors and simply divide the vectors into two groups without qualifying them as steady-state or transient. We have to check the members of each group and determine the correspondence. As the split is not perfect, a group can contain both SS and TS vectors. Thus, we decide the name of a group to be SS if most of its members are SS vectors and vice versa. If SS and TS vectors do not differ significantly in numbers, i.e. no clear clustering, we need to calibrate again after changing the method's parameters (scale of the axis and data length), which will be mentioned shortly.

For the online implementation (Fig. 2-right), at each time step t, we create a plot (referred to as plot A) of the N most recent plant measurements. Then, the calibrated VGG-16k extracts the feature vector A_v of plot A. Next, the k-Means algorithm determines to which group the plot A belongs to, based on the similarities between A_v and the vector members of each group (SS and TS). We can then infer whether the plant is stationary or not from the group assigned to A. At the next time step $t+1$, the data window is shifted forward, a new plot is created, and we run VGG-16k again. For a complete description of the CNN architecture and the k-Means algorithm implementation, please refer to [6].

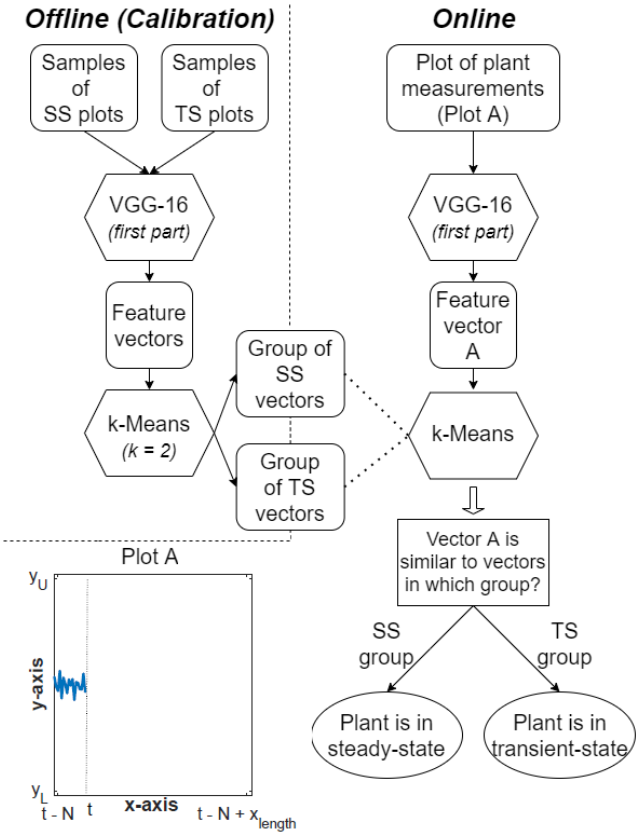

Fig. 2: How to use VGG-16k to carry out SSD. t indicates the current time step

The tuning parameters of the method are the properties of the plots fed to VGG-16k, including the scales of the axes and the data window length N. The data window length N should be as small as possible since, with a large data window, the measurement plots are prone to contain different behaviors (transient and stationary) and the performance of the VGG-16k decreases. The y-axis and x-axis limits ($[x_{length},\ y_L,\ y_U]$) can also be adjusted to support the visual judgement of the plant state (Fig. 2 – bottom left). As it is likely that a parameter set appropriate to human vision could also work for the CNN, tuning can be done by plant operators by selecting sets of parameters that enable them to identify SS and TS plots precisely. This easy-to-tune property is an advantage of our method in return for the larger data size, i.e. pixels from images vs. time series.

IV. STATISTICAL STEADY-STATE DETECTION METHODS

We selected two statistical SSD methods as the base to evaluate the our method's performance. These two have been used as benchmarks for SRTO evaluation in [1]. The first method, or SSD1 as we refer to, is proposed by [7]. The second method, which we call SSD2, is from [8]. We will briefly introduce the algorithms of these two methods for single-output systems. For multiple-output systems, we can apply them to each measured variable and consider that the system is stationary if some or all of them are at steady-state.

A. SSD1

At each time step t, carry out these steps:

- Step 1: Compute the filtered measurement $y_{f,t}$ from the current plant measurement y_t using a moving average filter as in (2), with λ_1 $(0 < \lambda_1 \leq 1)$ as the filter factor.

$$y_{f,t} = \lambda_1 \cdot y_t + (1 - \lambda_1) \cdot y_{f,t-1} \tag{2}$$

- Step 2: Compute a measure of variance $s^2_{1,t}$ based on the difference between the current measurement and the filtered trend, $(y_t - y_{f,t-1})$. Then, apply a first-order filter to the previously computed $s^2_{1,t}$ with λ_2 $(0 < \lambda_2 \leq 1)$ as the filter factor.

$$s^2_{1,t} = \lambda_2 \cdot (y_t - y_{f,t})^2 + (1 - \lambda_2) \cdot s^2_{1,t-1} \tag{3}$$

- Step 3: Compute $s^2_{2,t}$, which is a measure of variance of two consecutive measurements. Filter the value using the same procedure as in Step 2, with λ_3 $(0 < \lambda_3 \leq 1)$.

$$s^2_{2,t} = \lambda_3 \cdot (y_t - y_{t-1})^2 + (1 - \lambda_3) \cdot s^2_{2,t-1} \tag{4}$$

- Step 4: Use the R-statistics to compare $s^2_{1,t}$ and $s^2_{2,t}$ by computing the weighted variance ratio R_t.

$$R_t = \frac{(2 - \lambda_1) \cdot s^2_{1,t}}{s^2_{2,t}} \tag{5}$$

- Step 5: Compare the result of (5) with R_{crit}, which is a tuning parameter of the method. If $R_t > R_{crit}$, then we conclude that the plant is unlikely to be at steady-state, and vice versa.

The authors suggest setting the values of the filtered factors $\lambda_1, \lambda_2, \lambda_3$ and R_{crit} to be $0.2, 0.1, 0.1$ and 2, respectively. However, the parameters may need further tuning.

B. SSD2

At each time step t, carry out these steps:

- Step 1: Sample n most recent measurements y_{t-k} $(k = 0, 1, 2, ..., n-1)$. The authors suggest selecting $n = \beta \cdot \tau / \tau_s$. Here, τ and τ_s are the time constant of the system and the sampling interval, respectively. β is a tuning parameters in the range of $[3; 5]$.
- Step 2: Assume that the measurement at time $(t-k)$ can be written as in (6). Here, $m \cdot (t-k)$ is the deterministic drift component, μ is the mean of the hypothetical stationary process, and a_{t-k} following $\mathcal{N}(0, \sigma_a)$ is the white noise component.

$$y_{t-k} = m \cdot (t - k) + \mu + a_{t-k} \tag{6}$$

- Step 3: Estimate the slope value m.

$$m = \frac{1}{n-1} \cdot \left(\sum_{k=0}^{n-2} (y_{t-k} - y_{t-k-1})\right) \tag{7}$$

- Step 4: Estimate μ and σ_a.

$$\mu = \frac{1}{n} \cdot \left(\sum_{k=0}^{n-1} y_{t-k} - m \cdot \sum_{k=0}^{n-1} (t - k)\right) \tag{8}$$

$$\sigma_a = \sqrt{\frac{1}{n-2} \cdot \sum_{k=0}^{n-1} (y_{t-k} - m \cdot (t - k) - \mu)^2} \tag{9}$$

- Step 5: Calculate n absolute differences $|y_{t-k} - \mu|$ $(k = 0, 1, 2, ..., n-1)$. Perform t-test on these absolute differences to determine the steady-state flags p_{t-k} as in (10). Here, t_{crit} is the Student's critical value at a significance level α and a degrees-of-freedom n.

$$p_{t-k} = \begin{cases} 1, & \text{if } |y_{t-k} - \mu| \leq t_{crit} \cdot \sigma_a \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

- Step 6: Use p_{t-k} values from Step 5 to calculate P_t, the probability of the system being at steady-state, as in (11). Compare P_t with the cut-off value P_{thres}, where P_{thres} is a tuning parameter. If $P_t \leq P_{thres}$, we conclude that the plant is unlikely to be at steady-state, and vice versa.

$$P = \frac{\sum_{k=0}^{n-1} p_{t-k}}{n} \cdot 100\% \tag{11}$$

The tuning parameters here are n (or β) and t_{crit}.

V. CASE STUDIES

A. Case Study 1

*1) **Methodology**:* Here we test the performance of the proposed method, which will be referred to as CNN, in common scenarios of plant control and with the presence of noisy measurements. We also compare its performance with SSD1 and SSD2. The operation scenarios are defined as shown in Fig. 3. The figure contains 13 intervals, and each represents one operating scenario presented in Table I. Then, we combine data in Fig. 3 with different noise types listed in Table I to get the signals in Fig. 4a-Fig. 4c, and use them in Case Study 1.

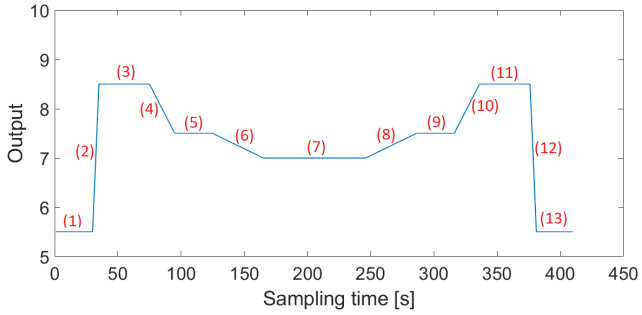

Fig. 3: The original plot in Case Study 1

Note that we do not carry out SSD on the complete plots in Fig. 4a-Fig. 4c. We use a sliding window containing $N = 20$ data points and $x_{length} = 100$, $y_L = 5$, $y_U = 10$. Then, we map the plot label (TS or SS) to the most recent measurement in the sliding window and keep shifting it forward until the end of the simulation. Here we also set $k = 3$ instead of $k = 2$, corresponding to step disturbances (TS), ramp disturbances (TS), and stationary (SS).

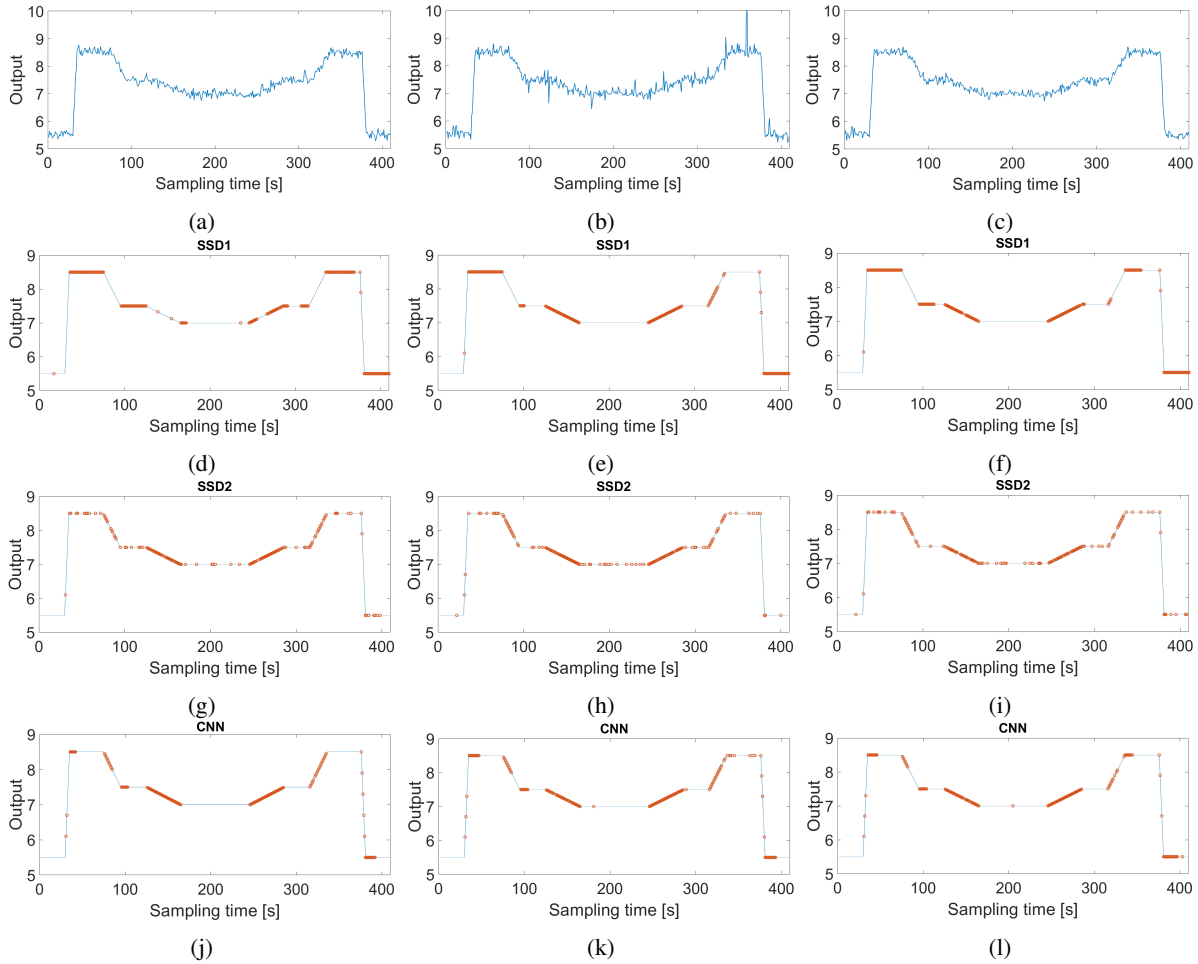

Fig. 4: SSD results of Case Study 1. From top to bottom: measurements, SSD1 results, SSD2 results, and CNN results. From left to right: white noise, heavy-tailed noise, and colored noise. Whenever the most recent measurements are fed to a SSD and it gives incorrect classification, we mark that instance by an orange dot.

TABLE I: Case Study 1: operating scenarios and noise types

Operating scenario	Represented in Fig. 3 as intervals
Step response	2, 12
Long ramp response	6, 8
Short ramp response	4, 10
Long steady-state period	3, 7, 11
Short steady-state period	1, 5, 9, 13
Noise type	**Noise distribution**
White noise (Fig. 4a)	$\mathcal{N}(0, 0.01)$
Noise following heavy-tailed distributions (Fig. 4b)	α-stable distribution ($\alpha = 1.5$, $\beta = \delta = 0$, $\gamma = 0.0725$)
Colored noise (Fig. 4c)	Exponential moving average filter (filter coefficient $\alpha = 0.9$) applied to random variables $\sim \mathcal{N}(0, 0.01)$

*2) **Results & Discussion**:* In Table II, we present the percentages of errors in the simulation, including Type I (indicate 'TS' when the process is at steady-state), Type II (not indicate 'TS' when the process is transient), and total errors (sum of Type I and Type II)

For SRTO applications, Type I errors in steady-state detection may decrease its potential economic benefits, since the optimization is not triggered as frequently as desired. On the other hand, Type II errors may lead to sub-optimal updates

TABLE II: Type of errors in percentage.

		SSD1	SSD2	CNN
White noise	Total	48.1%	36.6%	36.6%
	Type I	*40.4%*	*10.7%*	*6.9 %*
	Type II	*7.7 %*	*25.8%*	*29.7%*
Heavy-tailed noise	Total	44.5%	37.9%	38.9%
	Type I	*19.4%*	*11.0%*	*11.0%*
	Type II	*25.1%*	*26.9%*	*27.9%*
Colored noise	Total	48.8%	35.3%	38.1%
	Type I	*28.1%*	*11.5%*	*12.3%*
	Type II	*20.7%*	*23.8%*	*25.8%*

of the manipulated variables because the SRTO cycle begins when the process is not at steady-state. Table II indicates that CNN and SSD2 have a smaller percentage of total errors and a similar distribution between errors Type I and II. In turn, SSD1 has a smaller percentage of errors Type II.

Results in Fig. 4 confirm the findings in Table II; however, they show an advantage of CNN method over SSD2. Consider the error distributions of the two methods represented in Fig. 4g - Fig. 4l. While the error distribution of SSD2 does not have any clear pattern, the CNN method typically makes errors at the beginning of each interval. This pattern suggests

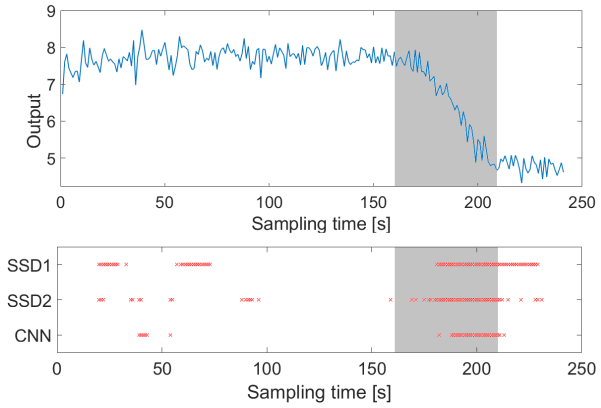

Fig. 5: Case Study 2: shaded area represents transient periods based on subjective evaluation. Top plot shows the experimental data (FI102). Bottom plot shows the transient flags (marked by red dots) given by each method.

that the CNN method has higher detection consistency, which can be beneficial to SRTO's performance. If CNN is used in the SRTO cycle, it will less likely trigger the estimation and optimization processes in the middle of a transition period. The figures also show that correctly classifying the long ramp response is difficult for all three methods.

B. Case Study 2

*1) **Methodology**:* After testing the methods using synthetic data, we apply CNN to real-life data collected from an experimental rig. The rig is a small-scale physical model of a gas-lift subsea oil-well network with 3 wells in parallel. A detailed description of the system containing the equipment size and the flowrate ranges can be found in [9]. Briefly, the experimental rig contains an "oil reservoir" that is represented by a tank and a centrifugal pump. The reservoir outlet pressure is fixed during the experiments using a PI controller that adjusts the pump rotation. After the reservoir, horizontal pipes emulate the oil wells on the seabed. The wells are then connected to vertical pipes called risers, they stand in for the pipelines directing the fluids from the seabed to the separation units on the surface. Pressurized gas is injected in the wells and provides an external source of energy for increasing the wells' production.

The pressure on top of the risers and the liquid production of each well are measured; however, we only use the liquid flowrate of one of the wells for testing the SSD methods. For Case Study 2, we use the experimental conditions proposed in [9]. Fig. 5 shows the data collected, which consists of 241 data points with a sampling interval of 5 seconds. For this application, we use $N = 20$ (100 seconds). The plot fed to the CNN had limits $x_{length} = 110$ seconds, and $y = [5, 10]$ L/min. Since we did not know exactly when the system was stationary, we relied on human judgment as the standard to evaluate SSD1, SSD2, and CNN methods. We consider that the system was transient between 14 - 17.5 minutes, approximately. For all other periods, it was stationary.

*2) **Results & Discussion**:* Results of Case Study 2 are shown in Fig. 5. The results given by all three methods were similar to each other and to our subjective judgment. But SSD1 was slower than the other two methods in recognizing when the system re-entered the stationary period. That is expected since the method relies on filters. Hence, rapidly tracking changes in the process can be challenging. Despite the efforts spent in properly tuning the filter factors λ, the final choice tends to be conservative, leading to behaviors like in Fig. 5. On the other hand, SSD2 error distribution did not present any clear pattern as in Case Study 1. Meanwhile, the CNN method required a longer interval than the other two methods for detecting when the system entered the transition period. However, as in the previous case study, the CNN method had the highest consistency in its predictions among the methods.

C. Case Study 3

*1) **Methodology**:* Here we investigate the effect of CNN in the SRTO performance using a *digital twin* of the rig in Case Study 2. The SSD method is a key part of the SRTO cycle. Since SSD acts like a gatekeeper deciding whether or not the cycle is triggered, it may influence the variability of the optimization results and the economic performance [1].

In [9], the authors compare the effect of different SSD methods on the SRTO performance when applied to the rig briefly described in Case Study 2. In Case Study 3, we keep all the settings of the SRTO in [9] but, instead of running the tests in the rig, we use a digital twin system for evaluating the CNN. The digital twin was developed to study how different production optimization methods interact with the rig. It contains: *a)* an accurate dynamic model of the experimental system; *b)* models for the PI-controllers; *c)* sensor models, where the measurement noise magnitude was carefully defined. The latter feature is important for testing the effect of measurement noise in the SSD method before it is implemented on the rig[1].

The SRTO's economic objective was to maximize the total production J (different oil prices in \$/L were considered for each well, i.e. $J = 20\ Q_{l,well_1} + 10\ Q_{l,well_2} + 30\ Q_{l,well_3}$). The optimizing inputs are the setpoints of the air injection valves $Q^{SP}_{g,well_{1,2,3}}$. The signals that are analyzed by the CNN are the liquid flowrates ($Q^{SP}_{g,well_{1,2,3}}$). If all three are considered at steady-state, we trigger the SRTO cycle.

For implementing our method, we add a plot generator to the system. At each sampling instant, the generator reads the digital twin measurements and translate them into plots, each one containing the last 18 measurements ($N = 18$), $x_{length} = 110$ seconds, and $y = [5, 10]$ L/min. Since the system sampling time is 1 second, the sliding window size is approximately of the same order of magnitude as the relevant process time constant for economic optimization ($\tau_{opt} \approx 20$ seconds [9]). For the training phase, we use plots obtained from a previous SRTO run with SSD2 as the steady-state detector.

[1]The codes are available at:
`http://github.com/Process-Optimization-and-Control/ProductionOptRig`

*2) **Results & Discussion**:* The results are shown in Figure 6. The first plot presents the system disturbances used in the simulation, and the second shows the profiles of the measurements used for steady-state detection. The resulting SSD flag pattern (3^{rd} plot) during the simulations was similar to Case Study 1, where all methods erroneously indicated that long ramp periods were static. Note that, if the measurement plots of the whole simulation (as in the 2^{nd} plot) are analyzed, the long ramp trend can be visualized. However, when the system state is assessed within an interval comprising the process time constant of interest, the plots are visually at steady-state.

A question that arises then is if the method is right by defining these intervals as static. It is worth then assessing the behavior of the optimizing decisions u, which are the setpoints of the gas flowrates ($Q^{SP}_{g,well_{1,2,3}}$ - 4^{th} plot). Despite being triggered during the ramp period, SRTO was always able to find a solution for the economic optimization problem (i.e. both the model adaptation and economic optimization problems converged). More importantly, the new solutions led to a better economic performance (5^{th} plot). This can be clearly seen in the increasing trends between 8 - 14 minutes, and after 16 minutes.

The apparent contradiction in the results brings us to a bigger discussion of how to define "stationarity" when, in fact, real processes are never at steady-state. The "stationarity" definition is often done subjectively based on the practitioner knowledge [1]. However, this decision should be based on how the new measurements degenerate the model adaptation and induce the occurrence of bias in the optimization decisions.

Independently on how to define "stationarity" properly, choosing the tuning parameters for SSD methods is challenging. They are typically interrelated, and the practitioner will be hardly aware of how the change in one parameter affects the method's performance. For example, in SSD1, it is difficult to infer if one should change only λ_1, or λ_1 and λ_2 together. Also, it is difficult to determine which choice is more beneficial to the performance. One of the advantages of our method is that we simply use pre-selected images of stationary and transient periods in the tuning step, which requires much less experience from the practitioner.

VI. Conclusion

The CNN method is a potential alternative to statistical SSD ones. Besides being robust to different types of noise, it has comparable performance to the traditional statistical methods but higher detection consistency, which can be beneficial for SRTO applications. Although the method is sophisticated, only the first setting-up would require specialized personnel since its tuning is intuitive, making it easier to use and debug, which can facilitate its future industrial implementation.

Although we did not try CNN on multi-output cases, scaling up could be straightforward, e.g. by generating more plots for each additional measurement. It should be noted that, we do not aim to replace the classical methods completely. We could use them together: CNNs for the measurements that require frequent re-tuning, and SSD1 for the others.

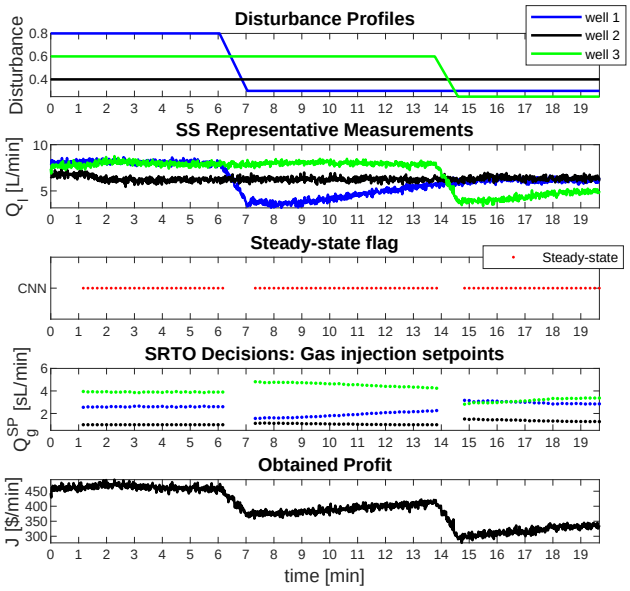

Fig. 6: Results of the CNN combined with SRTO applied to a digital twin of the lab-scale rig.

There are questions remaining to be addressed. For example, what happens if we use other CNN architectures and classification methods instead of VGG-16 and k-clustering, or if we change some implementation details, such as making the window length N and the length of the x-axis equal, rather than different as in this study.

ACKNOWLEDGMENT

The authors acknowledge financial support from the Norwegian Research Council, SUBPRO, grant number: 237893.

References

[1] M. Camara, A. Quelhas, and J. Pinto, Performance evaluation of real industrial RTO systems, *Processes*, vol. 4, no. 4, p. 44, 2016.

[2] S. Engell, Feedback control for optimal process operation, *Journal of Process Control*, vol. 17, no. 3, pp. 203–219, 2007.

[3] NIST and SEMATECH, e-Handbook of Statistical Methods. Maryland: U.S National Institute of Standards and Technology (NIST), 2012.

[4] E. Mohamed, Deep learning for vision systems. New York: Manning, 2020.

[5] (2021) VGG in tensorflow. [Online]. Available: https: `//www.cs.toronto.edu/~frossard/post/vgg16/`

[6] V. P. B. Nguyen, Application of machine learning in economic optimization, Master's thesis, Norwegian University of Science and Technology, Trondheim, 2021.

[7] S. Cao and R. R. Rhinehart, Critical values for a steady-state identifier, *Journal of Process Control*, vol. 7, no. 2, pp. 149–152, 1997.

[8] D. Kelly and J. Hedengren, A steady-state detection (SSD) algorithm to detect non-stationary drifts in processes, *Journal of Process Control*, vol. 23, pp. 326–331, 2013.

[9] J. Matias, J. P. de Castro Oliveira, G. A. Le Roux, and J. Jaschke, Real-time optimization with persistent parameter adaptation applied to experimental rig, *IFACPapersOnLine*, vol. 54, no. 3, pp. 475–480, 2021.