ICIC Express Letters
Volume **15**, Number **7**, July **2021**

 ISSN 1881-803X
pp. **737–746**

ADVERSARIAL MULTIVARIATE REGRESSION AND AUTO-REGRESSION FOR PREDICTING COMPLETION TIME OF CONTAINER TERMINALS OPERATION

Imam Mustafa Kamal[1] and Hyerim Bae[2,*]

[1]Big Data Department
[2]Department of Industrial Engineering, Major in Industrial Data Science & Engineering
Pusan National University
2, Busandaehak-ro 63 beon-gil, Geumjeong-gu, Busan 46241, Korea
imamkamal@pusan.ac.kr; *Corresponding author: hrbae@pusan.ac.kr

Received November 2020; accepted January 2021

Abstract. *In port container terminals, loading and unloading are the most important operations. Their lateness will be costly for both vessel and terminal owners. Moreover, port operations are many and various, and can be complex as well, since some of them can be run simultaneously in order to minimize overall operation time. Given the complexity of data obtained from a container terminal, we argue that handling it by means of a standard regression analysis will prove challenging. Therefore, in this paper, we propose a completion time prediction method for loading and unloading operations in container terminals. Specifically, for that purpose, we introduce, herein, adversarial multivariate regression (AMR) and adversarial multivariate auto-regression (AMAR). Extensive experiments and comparisons revealed that both AMR and AMAR outperform the existing multivariate regression methods.*

Keywords: Regression, Generative adversarial networks, Deep learning, Forecasting, Container terminal

1. **Introduction.** In 2017, *Shipping Statistics and Market Review* (*SSMR*) reported that within 15 years, the world merchant fleet's tonnage has more than doubled [1]. In the context of such traffic, a port has only limited resources and space for handling of containers and the related transactions [2]. Therefore, more efficient port operations are necessary. Loading and unloading operations are the most important in container terminals. Accurate prediction of loading and unloading times is desirable, as it can be helpful for scheduling, process optimization, and simulation. Note too, that lateness of loading or unloading operations can incur costly penalties for both vessel and container terminal owners.

The data generated by a terminal operating system is very complex, since there are many and complex operations that occur simultaneously. For instance, for vessel stability, heavier containers must be put in a lower stack. In contrast, in the storage yard, heavy containers are commonly positioned in a higher stack, since from there, they can be loaded first into the vessel's lower stack(s). Therefore, without good planning, the operational times of loading and unloading can be unnecessarily time-consuming and expensive. Note too, that whereas operations can occur simultaneously, there are many vessels and containers needed to be served while resources are often limited. Given such conditions, the data generated by terminal operation systems (TOS) can be very complex and challenging to deal with.

Nowadays, deep learning, since its introduction in [3,4], is showing promising results in handling of complex problems, and is already widely employed in the industry [2,5], finance [6,7], and biomedical [8,9] industries. To the best of our knowledge, not many researchers

DOI: 10.24507/icicel.15.07.737

have implemented AI or deep learning for container terminal operations. Therefore, in this paper, we propose adversarial multivariate regression (AMR) and adversarial multivariate auto-regression (AMAR) which is trained in an adversarial manner for the prediction of container loading and unloading completion times in container terminals. Both AMAR and AMR outperform the traditional regression method such as linear, lasso, and ridge regression. Moreover, they can result in better accuracy than deep neural network which is not trained in an adversarial manner.

The rest of this paper's discussion is organized as follows. Section 2 presents related work. Sections 3 and 4 discuss the proposed method and experimental results, respectively. Finally, Section 5 draws conclusions and looks ahead to future work.

2. **Related Work.** Several studies related to container terminal operations have been conducted. Kourounioti et al. predicted the dwell time of containers using artificial neural networks [10]. For maximization of container terminal productivity, Ahn et al. proposed coordinated real-time scheduling for optimization of container terminal operations [11]. Luo et al. incorporated particle swarm optimization and a genetic algorithm to optimize container terminal operations [12]. Kamal et al. transformed categorical data (event-logs) to the RGB format for prediction of container processing times using PixelCNN [2].

Some researchers have employed adversarial learning to solve regression problems in various fields. Javanmard et al. provided precise and comprehensive understanding of the role of adversarial training in the context of linear regression with Gaussian features [13]. Zhang et al. predicted age based on facial images using a conditional adversarial autoencoder [14]. Rezagholiradeh and Haidar tried to solve the semi-supervised learning problem with generative adversarial networks (GANs) for regression [15]. Tong et al. studied the problem of adversarial linear regression with multiple learners [16]. Unlike the previous research, in this study, we introduce adversarial learning networks to solve multivariate regression problems.

3. **Method.** In this section, the problem definition for container loading and unloading operations, the proposed model for prediction of container completion time, and the architecture of our proposed method are defined.

3.1. **Problem definition.** Container flows from the yard to the vessel and from the vessel to the yard in a container terminal are depicted in Figure 1. For simplicity, we merely visualize three-yard bays and one vessel, wherein under real conditions, there would be many more than three yards in a bay, multiple vessels, and a more complex yard-bay design. There are three main activities in the loading or unloading operation, namely, yard activity, movement from the yard to the vessel or from the vessel to the yard, and vessel-loading/unloading activity. Each activity has a corresponding resource for its execution. For instance, in the case of loading, yard activity is executed by a yard crane that picks up the container and puts it on the truck. Then, the truck will move it to the berthing location. While in the berthing location, the quay crane will pick up the container and put it in the given position on the vessel.

As shown in Figure 1, given t_0, which represents the starting of a corresponding activity (yard-crane movement or quay-crane movement for loading and unloading activity, respectively), we try to predict the completion operation time (t_N). For simplicity, we introduce an event as a single loading or unloading operation instance. Therefore, a loading event consists of yard-crane movement, truck movement, and quay-crane movement; an unloading event, on the other hand, consists of quay-crane movement, truck movement, and crane movement.

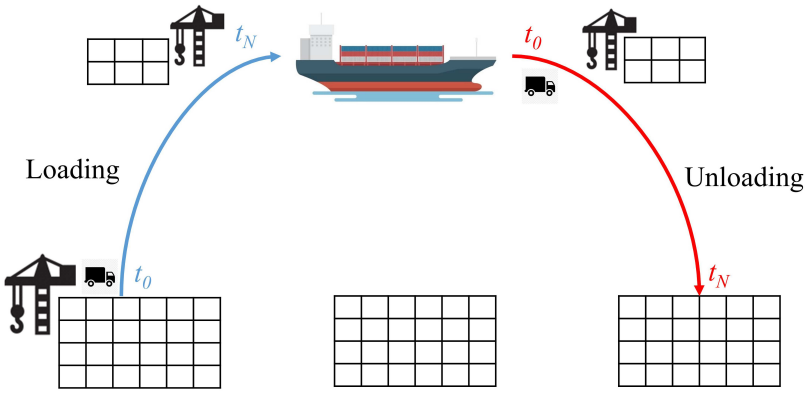

FIGURE 1. Container flows from the yard to vessel and from vessel to yard for loading and unloading, respectively

3.2. **Models.** Suppose that $x_1, x_2, \ldots, x_n$ represents the attribute of a corresponding container, such as container weight, yard-bay number, and yard-tier number, where n represents the total number of attributes. Given those independent variables, we can predict y as a dependent variable, y representing the processing time of the corresponding container for each activity. We adopt the conditional GAN [17] model to solve multivariate regression adversarially given the label y, as denoted in Equation (1), where regressor (R) tries to minimize the error between $\hat{y}$ and y, given x. Simultaneously, discriminator (D) tries to maximize the difference between $\{x, \hat{y}\}$ and $\{x, y\}$.

$$\min_R \max_D V(D,R) = \mathrm{E}_{x\sim p_{data}(x,y)}\left[\log D\left(\{x,\hat{y}\},\{x,y\}\right)\right] + \mathrm{E}_{\hat{y}\sim p_{\hat{y}}(\hat{y})}\left[\log\left(1 - D\left(R\left(\hat{y}|x\right)\right)\right)\right] \quad (1)$$

The AMR architecture is shown in Figure 2. Our AMR model is like a basic generative adversarial network (GAN) with a conditional version (CGAN). However, it has different inputs that are treated to solve a multivariate regression case. In the regressor, we utilize a deep neural network (DNN) model designed to solve the regression task, while the discriminator is also a DNN mode, but it is dedicated to distinguishing between the real (y) and a generated one ($\hat{y}$).

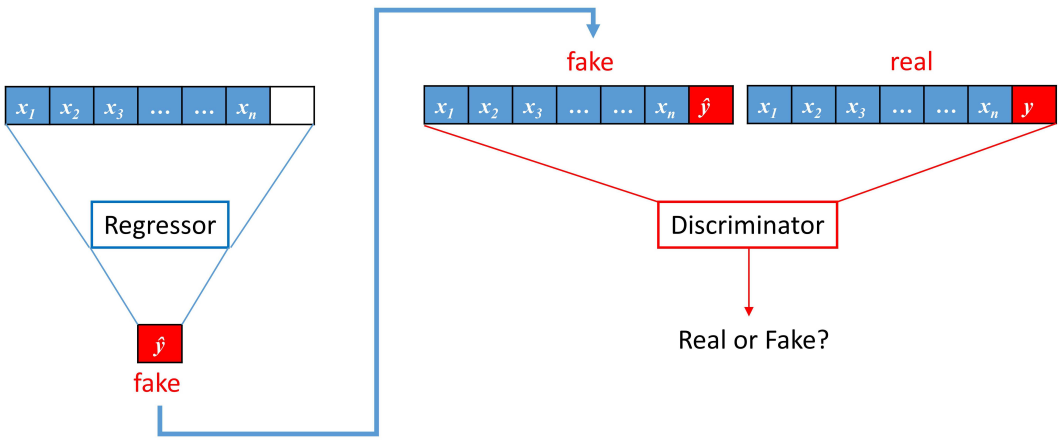

FIGURE 2. AMR architecture

The AMAR model is depicted in Figure 3. Unlike the AMR model, here, the output consists of three rows in the two-dimensional format representing an event consisting, for the loading operation, of yard-crane movement, truck movement, and quay-crane

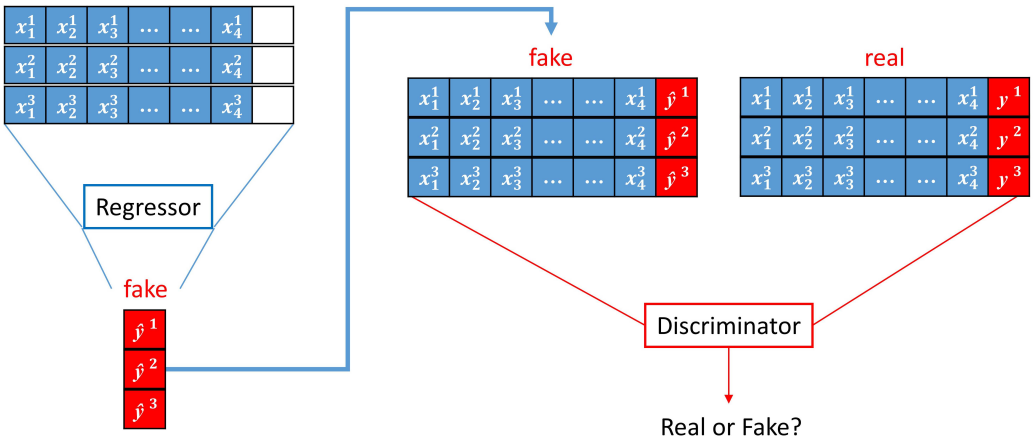

FIGURE 3. AMAR architecture

movement, respectively, and vice versa for the unloading operation. There in the AMAR model, in a single process, we can have three outputs of $\hat{y}$: $\hat{y}^1$, $\hat{y}^2$, and $\hat{y}^3$, for the processing time of yard-crane movement, truck movement, and quay-crane movement, respectively. We were motivated to use two-dimensional data in order to find the spatial relation among activities in a single event, while the relations among events can be found in the sequence data. Therefore, the shuffling data mechanism was not implemented here.

3.3. **Architecture.** We conducted an extensive experiment to determine the optimal architectures of AMR and AMAR. The deeper networks were pruned for overfitting and high computational cost, while the swallower networks were pruned for under-fitting. The optimal architectures and hyper-parameters of the AMR and AMAR models are provided in Table 1. Regressor and discriminator networks utilize random normal initializers with a standard deviation of 0.02. The optimizer for the discriminator is Adam, while RMSprop is for the regressor networks. The loss function of the discriminator network is binary cross-entropy, because a discriminator acts as a binary classifier distinguishing between the real dataset and the fake one; meanwhile, the loss function of the regressor network is mean square error (MSE). MSE was chosen because our regressor minimizes the errors of y and $\hat{y}$. They are both run for 300 epochs, which takes longer than a standard deep learning model, since our adversarial learning between two networks is more difficult to converge.

TABLE 1. AMR and AMAR architectures

Layer	AMR		AMAR	
	Regressor	Discriminator	Regressor	Discriminator
1	Dense(units = 32)	Dense(units = 32)	Conv1D(filters = 64, kernel_size = 3)	Conv1D(filters = 64, kernel_size = 3)
	Batch Norm.	Batch Norm.	Batch Norm.	Batch Norm.
	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)
	Relu	Relu	Relu	Relu
2	Dense(units = 16)	Dense(units = 16)	Dense(units = 32)	Dense(units = 16)
	Batch Norm.	Batch Norm.	Batch Norm.	Batch Norm.
	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)	Dropout(0.5)
	Relu	Relu	Relu	Relu
3	Dense(units = 1)	Dense(units = 5)	Dense(units = 3)	Dense(units = 3)
	Tanh	Sigmoid	Tanh	Sigmoid

4. **Experiments.** In this section, we provide the dataset, data pre-processing, results, and discussion of our research.

4.1. **Data pre-processing.** The dataset was a one-year recording of loading and unloading operations obtained from Busan port, South Korea. It contains 994,416 records, with the following attributes: container weight, full or empty container (full or empty), activity name (yard-crane movement, truck movement, quay-crane movement), activity type (loading or unloading), twin activity (twin or single operation), yard-bay number, yard-row number, yard-tier number, and processing time for each activity. Some categorical attributes such as full or empty container, activity name, activity type and twin activity are converted to dummy variables. The processing time for each activity was normalized to zero center (-1 to 1). For validation purposes, we split the dataset into 80% training data and 20% testing data.

The correlations between the dependent (processing time) and independent (attributes) variables are shown in Table 2. Except activity name, all of the attributes have a low correlation value against the completion time. There are three attributes with negative correlation values, namely container weight, twin activity and yard-tier number. Based on the data, the less heavy a container is, the longer its processing time (with low correlation) is. Accordingly, a full container (0) has a shorter processing time than does an empty container (1). Activity type represents loading (0) and unloading (1). We can infer that unloading time is slightly faster than loading time. Twin activity represents a binary variable: 1 represents a not-twin operation (a single process), while 0 represents a twin operation. Even though the correlation value is not too high, it can be said that a twin operation has a shorter completion time. As for yard-tier number, the higher the tier, the faster the processing time. This is due to the fact that, for a container in a lower tier (stack), some handling operations will be needed if there are other containers on top of it.

Table 2. Correlations between processing time and various container attributes

Attribute	Correlation
Container weight	−0.00215799
Full or empty container	0.000276810
Activity name	Significant based on ANOVA test
Activity type	0.000689177
Twin activity	−0.00271425
Yard-bay number	0.000300959
Yard-row number	0.000290543
Yard-tier number	−0.00013958

4.2. **Results and discussion.** To assess the robustness of AMR and AMAR, we conducted a comparison with standard regression models including linear regression, lasso regression, and ridge regression. Further we compared our proposed method with the DNN model. Root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percentage error were utilized to evaluate the respective accuracies.

Table 3 shows a comparison between our proposed method and the other regression methods. We can reveal that a traditional regression method such as linear, lasso or ridge regression is not able to accurately predict the completion time of a loading or unloading operation. This is confirmed by Figures 4, 5 and 6. Because the container loading and unloading dataset is very complex, as shown in Table 2, the correlation between the independent and dependent variables is very low. In contrast, AMAR outperforms the other methods in terms of RMSE and MAE. In terms of MAPE, on the other hand,

TABLE 3. Error-rate comparison between the proposed method (AMR, AMAR) and existing regression models

Model	RMSE	MAE	MAPE
Linear regression	4.88332	4.88332	76.03237
Lasso regression	4.88326	3.39685	76.03914
Ridge regression	4.88436	3.39729	75.95312
DNN	3.83464	1.73871	33.27271
AMR	3.78080	1.73694	**32.26859**
AMAR	**3.36037**	**1.52563**	36.54715

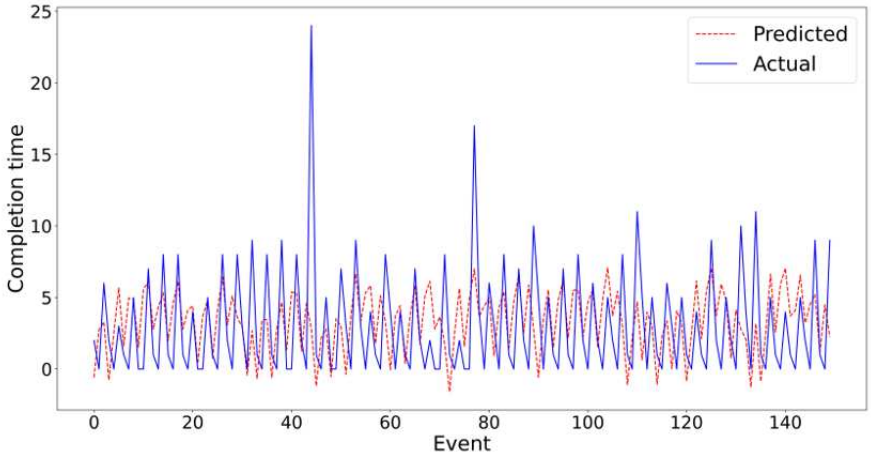

FIGURE 4. Linear regression

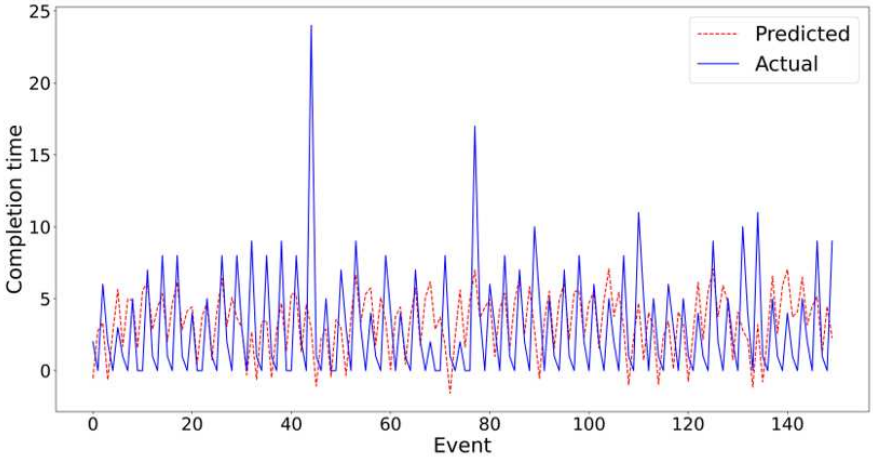

FIGURE 5. Lasso regression

AMR has the most minimum value. The prediction by DNN has a high magnitude value compared with our AMAR, while AMR has the lowest magnitude as shown in Figures 7, 8, and 9. The reason that our AMAR had a better prediction than AMR is that it can learn spatial relations inter-event. Note that, in a single input, AMAR contains multiple multivariate data. Moreover, compared with the DNN, our AMR and AMAR regressor networks are reinforced by discriminator networks for prediction of $\hat{y}$ during training.

The convergence of AMR and AMAR is shown in Figures 10 and 11, respectively. While $D1$ represents the discriminator against the real dataset, $D2$ corresponds to the discriminator against the fake dataset, and R represents the regressor networks. In AMR,

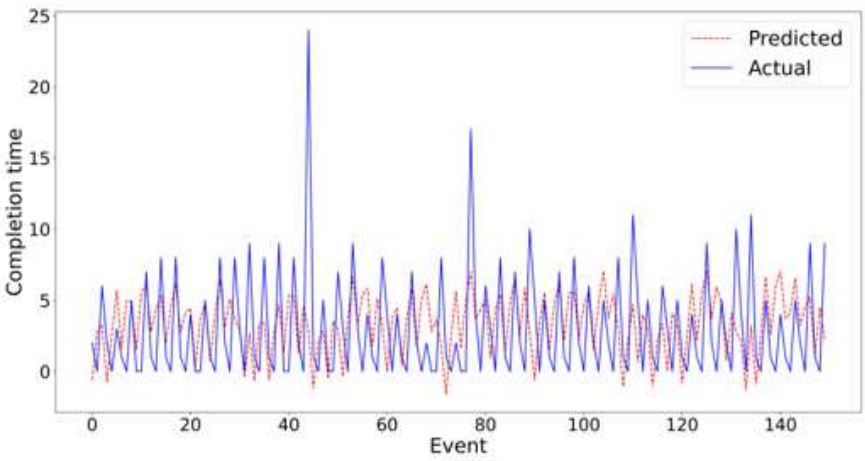

FIGURE 6. Ridge regression

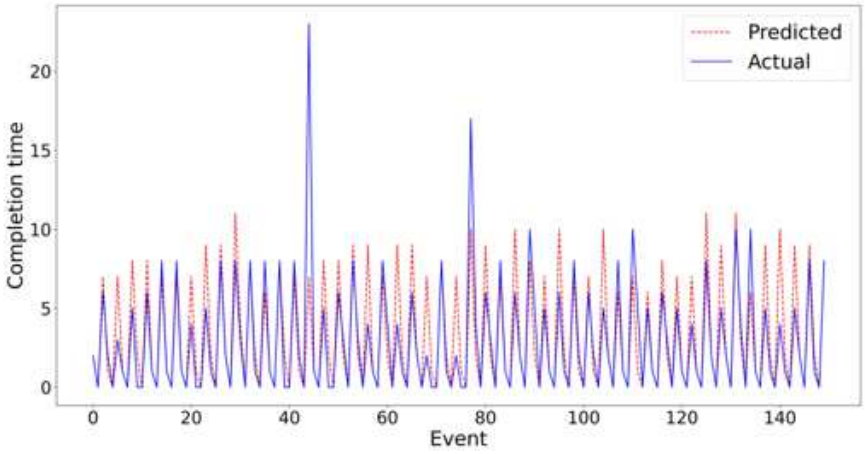

FIGURE 7. Deep neural networks (DNN)

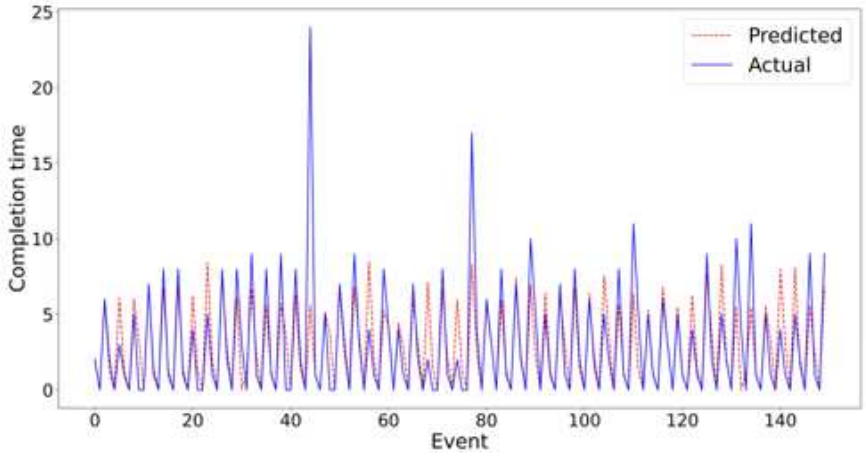

FIGURE 8. AMR

the R converges after 100 epochs, and while the $D1$ loss is slightly steady after 10 epochs, it cannot distinguish between the real and the fake dataset. Meanwhile, in AMAR, the R converges faster after 40 epochs, and $D2$ and $D1$ have low values after 10 epochs; however, whereas $D2$ has a higher value than does $D1$, the discriminator has more difficulty distinguishing the fake dataset from the real one.

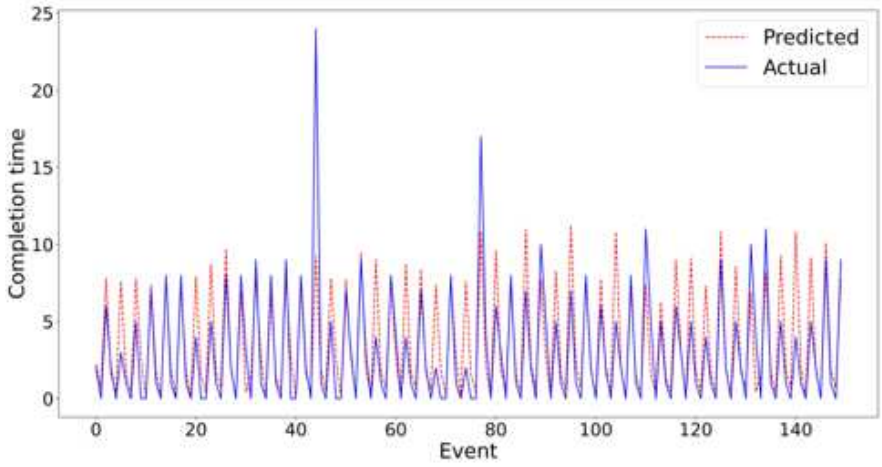

FIGURE 9. AMAR

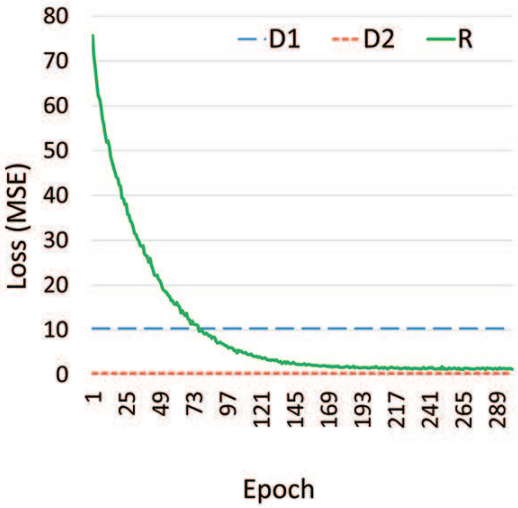

FIGURE 10. AMR convergence during 300 epochs

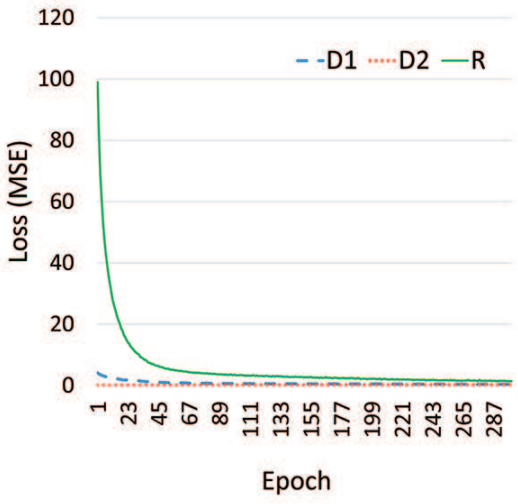

FIGURE 11. AMAR convergence during 300 epochs

5. **Conclusions and Future Work.** Recently, adversarial learning has been widely utilized to solve complex problems. In container terminals, predicting completion time is important for making good scheduling and optimizing the overall operational process. However, a container terminal has a complex process, since all of the resources have a conflict of interest in serving both the vessel and hinterland transportation. In this paper, we proposed an adversarial learning approach to predict the completion time of the container process. We introduced two models, namely AMR and AMAR. Both of them outperform the existing multivariate regression methods, including linear regression, ridge regression, lasso regression, and DNN. In the future, we plan to employ AMR for various other problems.

Acknowledgment. This research was suppoted by the MSIT (Ministry of Science and ICT), Korea, under the Grand Information Technology Research Center support program (IITP-2020-2016-0-00318) supervised by the IITP (Institute for Information & communications Technology Planning & Evaluation) and in part of the research projects of 'Development of IoT Infrastucture Technology for Smart Port' funded by the Ministry of Oceans and Fisheries, Korea.

REFERENCES

[1] ISL (Institute of Shipping Economics and Logistics), *Shipping Statistics and Market Review (SSMR)*, vol.61, Bremen, Germany, 2017.

[2] I. M. Kamal, H. Bae, N. I. Utama and C. Yulim, Data pixelization for predicting completion time of events, *Neurocomputing*, vol.374, pp.64-76, 2020.

[3] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, Gradient-based learning applied to document recognition, *Proc. of the IEEE*, vol.86, no.11, pp.2278-2324, DOI: 10.1109/5.726791, 1998.

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, Generative adversarial nets, *NIPS*, pp.2672-2680, 2014.

[5] I. M. Kamal, R. A. Sutrisnowati, H. Bae and T. Lim, Gear classification for defect detection in vision inspection system using deep convolutional neural networks, *ICIC Express Letters, Part B: Applications*, vol.9, no.12, pp.1279-1286, 2018.

[6] I. M. Kamal, H. Bae, S. Sunghyun and H. Yun, DERN: Deep ensemble learning model for short- and long-term prediction of baltic dry index, *Applied Sciences*, vol.10, 2020.

[7] I. M. Kamal, H. Bae, S. Sim, H. Kim, D. Kim, Y. Choi and H. Yun, Forecasting high-dimensional multivariate regression of Baltic Dry Index (BDI) using Deep Neural Networks (DNN), *ICIC Express Letters*, vol.13, no.5, pp.427-434, 2019.

[8] O. Ronneberger, P. Fischer and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, *arXiv.org*, arXiv: 1505.04597, 2015.

[9] I. M. Kamal, N. A. Wahid and H. Bae, Gene expression prediction using stacked temporal convolutional network, *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Busan, Korea, pp.402-405, DOI: 10.1109/BigComp48618.2020.00-41, 2020.

[10] I. Kourounioti, A. Polydoropoulou and C. Tsiklidis, Development of models predicting dwell time of import containers in port container terminals – An artificial neural networks application, *Transportation Research Procedia*, vol.14, pp.243-252, 2016.

[11] E. Y. Ahn, K. Park, B. Kang and A. K. R. Ryu, Real time scheduling by coordination for optimizing operations of equipments in a container terminal, *The 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI2007)*, Patras, pp.44-48, DOI: 10.1109/ICTAI.2007.138, 2007.

[12] J. X. Luo, D. Wu, Z. Ma, T. Chen and A. Li, Using PSO and GA to optimize schedule reliability in container terminal, *2009 International Conference on Information Engineering and Computer Science*, Wuhan, 2009.

[13] A. Javanmard, M. Soltanolkotabi and H. Hassani, Precise tradeoffs in adversarial training for linear regression, *arXiv.org*, arXiv: 2002.10477, 2020.

[14] Z. Zhang, Y. Song and H. Qi, Age progression/regression by conditional adversarial autoencoder, *arXiv.org*, arXiv: 1702.08423, 2017.

[15] M. Rezagholiradeh and M. A. Haidar, Reg-Gan: Semi-supervised learning based on generative adversarial networks for regression, *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, AB, pp.2806-2810, DOI: 10.1109/ICASSP.2018.8462534, 2018.

[16] L. Tong, S. Yu, S. Alfeld and Y. Vorobeychik, Adversarial regression with multiple learners, *arXiv.org*, arXiv: 1806.02256, 2018.
[17] M. Mirza and S. Osindero, Conditional generative adversarial nets, *arXiv.org*, arXiv: 1411.1784, 2014.