Korean J. Chem. Eng., 36(3), 325-332 (2019)
DOI: 10.1007/s11814-018-0204-8

pISSN: 0256-1115
eISSN: 1975-7220

Development of surrogate model using CFD and deep neural networks to optimize gas detector layout

Kyeongwoo Jeon*, Seeyub Yang*, Dongju Kang*, Jonggeol Na, and Won Bo Lee*,†**

*School of Chemical and Biological Engineering, Seoul National University, Gwanak-ro 1, Gwanak-gu, Seoul 08826, Korea
**Clean Energy Research Center, Korea Institute of Science and Technology (KIST),
5 Hwarang-ro 14-gil, Seongbuk-gu, Seoul 02792, Korea

(*Received 11 April 2018 • accepted 5 December 2018*)

Abstract−To reduce damage arising from accidents in chemical processing plants, detection of the incident must be rapid to mitigate the danger. In the case of the gas leaks, detectors are critical. To improve efficiency, leak detectors must be installed at locations after considering various factors like the characteristics of the workspace, processes involved, and potential consequences of the accidents. Thus, the consequences of potential accidents must be simulated. Among various approaches, computational fluid dynamics (CFD) is the most powerful tool to determine the consequences of gas leaks in industrial plants. However, the computational cost of CFD is large, making it prohibitively difficult and expensive to simulate many scenarios. Thus, a deep-neural-network-based surrogate model has been designed to mimic FLACS (FLame ACceleration Simulator), one of the most important programs in the modeling of gas leaks. Using the simulated results of a proposed surrogate model, a sensor allocation optimization problem was solved using mixed integer linear programming (MILP). The optimal solutions produced by the proposed surrogate model and FLACS were compared to verify the efficacy of the proposed surrogate model.

Keywords: Gas Detector Allocation, Optimization, Milp, Computational Fluid Dynamics, FLACS, Artificial Neural Network, Surrogate Model

INTRODUCTION

The leakage of toxic gases is a common type of accident in chemical industry that can be extremely fatal if proper emergency mitigation measures are not employed. Extensive dispersion of leaked toxic gases can trigger serious fatalities, affecting both site operators and nearby residents. For example, the methyl isocyanate gas leak and dispersion in Bhopal, India, in 1984 killed about 3800 people and caused thousands of injuries. In addition, the leakage of flammable gases can lead to fire or explosion, resulting in jet fire or fireballs. Therefore, gas leakage must be detected quickly to enable the correct emergency response. To mitigate the effects of these accidents, gas detectors are the most useful safety devices. The use of a large number of gas detectors can improve detection performance. However, as the number of detectors increases, the likelihood of false alarms, as well as cost, increases. Moreover, the location of a gas detector is strongly associated with its performance. Thus, the number and positions of gas detectors must be determined considering these important factors.

To solve the detector allocation problem, accurate gas dispersion models for predicting gas concentrations must be implemented, and the calculated consequences of leakage accidents must be considered to determine the number and positions of gas detectors. There are some gas dispersion models such as black box model and Gaussian dispersion model, and simulators tools such as ALOHA and PHAST. These methods are quite accurate in flat terrain that does not have any obstacles. However, they do not consider three-dimensional obstacles or terrain effects. Computational fluid dynamics (CFD) is the most accurate model for gas diffusion, because CFD can reflect the impact of obstacles and terrain around an accident site. Several studies have verified the accuracy of CFD simulations by comparison with real experimental data [1-4].

Several researchers have studied the detector allocation problem using CFD simulations. Hamel et al. [5] optimized the location of sensors for the detection of chemical, biological, and nuclear attacks in urban areas using CFD. Berry et al. [6] handled the optimal detector allocation problem of urban water networks using mixed integer linear programming (MILP) and CFD. Legg et al. [7,8] studied the optimization of the gas detector layout problem based on stochastic programming and, later, conditional values and CFD. Benavides-Serrano modified Legg's previous work to include the uniform or non-uniform probability distribution functions governing the uncertainty of sensor detection [9,10]. Benavides-Serrano evaluated several gas detector optimization methods using quantitative assessment [11].

The most attractive advantage of CFD-based gas detector optimization research is its high accuracy, but CFD is very time-consuming and computationally expensive. Thus, it is impossible to simulate enough scenarios, which may reduce the reliability of MILP-based methods. Davis et al. [12] and Vázquez-Román et al. [13] solved the gas detector layout problem using only eight sets of CFD results. Gomes et al. [14] also solved a similar problem with only

†To whom correspondence should be addressed.
E-mail: wblee@snu.ac.kr

32 sets of results.

To reduce the drawbacks of CFD, surrogate models or reduced-order models have been employed. In the field of chemical safety, Wang et al. [15] developed a surrogate model of liquid natural gas (LNG) gas dispersion based on Gaussian process regression (GPR) and segmented principal component transform-principal component analysis (SegPCT-PCA) combined with the FLame ACceleration Simulator (FLACS) and Hamersley sampling. Margheri and Sagaut [16] used c-APK technology (anchored-ANOVA-POD/Kriging) and CFD to develop a surrogate model for toxic gas dispersion. Previously, we verified that the performance of a surrogate model based on machine learning is close to that of CFD [17].

The objective of this study is to solve problems of large computational cost by reducing the number of required simulations, while maintaining high accuracy. We used CFD to create gas dispersion scenarios for high accuracy. To solve the computational cost problem, a surrogate model based on a deep neural network was developed based on CFD data, and several additional scenarios were created using the surrogate model. This makes the computational cost of CFD simulations dramatically reduced. Finally, sensor allocation was optimized based on the original scenarios and the additional generated scenarios.

METHOD

The whole process of optimization is shown in Fig. 1. First, design of experiments (DoE) are carried out on the surrogate model. As a sampling method, Latin hypercube sampling (LHS) is used. This is a widely used meta-modeling technique for obtaining high quality samples. Of course, there are state-of-the-art sampling methods that perform better than LHS. However, because this study aims to overcome problems resulting from small number of samples or problems arising from sampling method, the general sampling method was used. In addition, deep neural network (DNN) regression is used as a surrogate model fitting method. Using the LHS, M_1 base scenarios with different inputs are generated. Then, to find the detection time of the gas detectors, all the base scenarios are solved by CFD. The CFD results are used to train a surrogate model based on DNN. After the model has been developed, it is used to generate M_2 extended scenarios. (M_1+M_2) scenarios are used to solve MILP, which determines the optimal sensor position. In this study, M_1=30 and M_2=300 is used to avoid situations where the number of potential detectors is greater than the number of leakage scenarios, while maintaining a realistic number of CFD simulations.

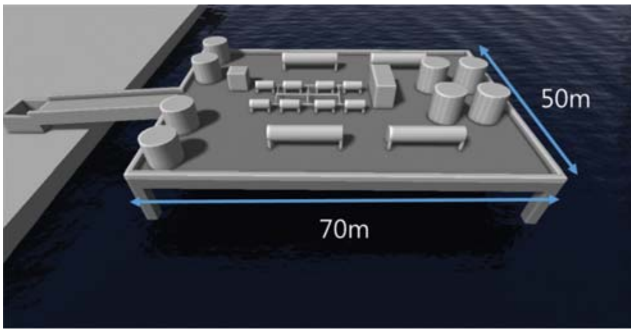

Fig. 2. Geometry of an LNG storage.

1. Mathematical Formulation

FLACS was originally developed to simulate explosions, but the current version of FLACS (v10.7) can also calculate compressible fluid flow by solving three-dimensional Reynolds-averaged Navier-Stokes (3D RANS) equations, which are widely used in other CFD studies. These 3D RANS equations with three conservation equations (mass, energy, and momentum) are numerically solved based on the finite volume model and the k-ε turbulence model [18] in non-uniform Cartesian coordinates. In addition, to calculate obstacles that are smaller than the sub-grid and reduce computational costs effectively, distributed porosity concepts are applied.

2. CFD Setting

The geometry of the LNG storage facility, shown in Fig. 2, was derived from the pool spread simulation included in FLACS. The methane, ethane, and propane volume composition of the LNG used in the simulation was 95 : 4 : 1.

The grid used in the simulation was created based on validated guidelines (FLACS v10.7 user manual) [19]. Hansen et al. [20] validated FLACS against experimental data sets based on this method. In the gas dispersion simulation, a uniform cell of 1-1.5 m in the main domain of interest (except for large release (100-200 kg/s) and jet release) is recommended. In extended domains outside the region of interest, the cell size can be gradually stretched, but it is recommended that grid-stretching factors be kept below 1.2. Based

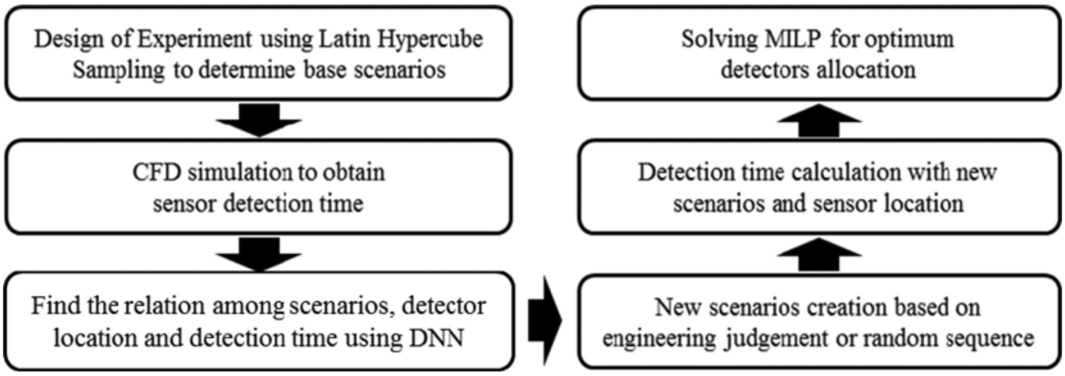
Design of Experiment using Latin Hypercube Sampling to determine base scenarios
CFD simulation to obtain sensor detection time
Find the relation among scenarios, detector location and detection time using DNN
New scenarios creation based on engineering judgement or random sequence
Detection time calculation with new scenarios and sensor location
Solving MILP for optimum detectors allocation

Fig. 1. Process of detector layout optimization with surrogate model.

on the guidelines, the grid was set to 1×1×1 m in the x, y, and z directions in the core region, which grows by a factor of 1.2, generating 196,236 cells.

As scenario inputs, there are five variables that can be classified into two categories, meteorological variables and leak variables. U_x and U_y are meteorological variables: the wind velocity in the x and y directions, respectively. The leak variables are $\dot{m}$, S_x, and S_y, representing the mass rate (kg/s), the *x*-coordinate (m) of the leak point, and the *y*-coordinate (m) of the leak point, respectively. Thirty scenarios were generated by changing these five variables within a certain range using LHS (-3.5 m/s$<U_x$, $U_y<3.5$ m/s, 0 kg/s$<\dot{m}<100$ kg/s, -35 m$<S_x<35$ m, -25 m$<S_y<25$ m).

The results of the LHS are shown in Fig. 3. Each point is a leak source location, and the size of the point indicates the leak rate of LNG. The length and direction of the arrow indicate the wind speed and wind direction, respectively.

The detectors have 1587 potential positions based on uniform placement within the geometry (23, 23, and 3 in the x, y, and z directions, respectively). This was combined with 30 scenarios to generate 47,610 base samples. The five variables for creating the leak scenario and the x, y, and z coordinates of the potential detectors were combined as DNN input variables.

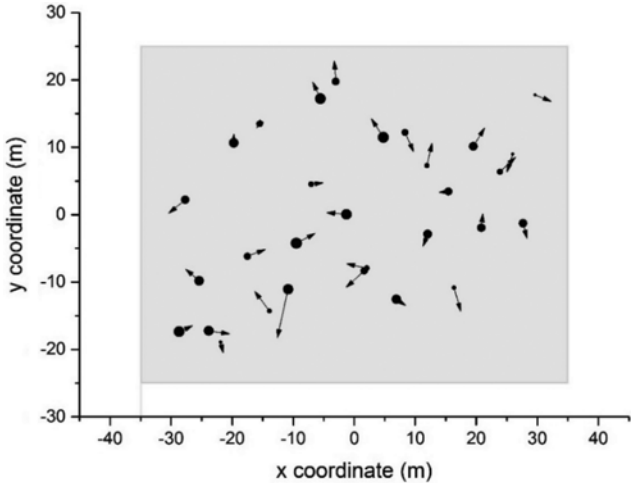

Fig. 3. Latin hypercube sampling result (30 base leak scenarios).

3. DNN Regression

Before entering the training phase, the data samples were separated into detected samples and undetected samples. The minimum detectable value of the sensor was set to 20% of the lower flammable limit (LFL) value of the LNG gas. The detection time was defined as the moment when the concentration of LNG gas in the potential detector location exceeds 20% LFL. Some of the sensors detect the leaked LNG gas and some fail. Detectors that fail to detect the leak do not have an output value, i.e., a detection time. Scenarios where the leak was detected were defined as detected samples, and scenarios where the leak was not detected were defined as undetected samples. Because the scenarios to be generated through the final surrogate model are detected scenarios, only the detected scenarios were used for the training steps. Undetected scenarios were excluded because they do not have output values and cannot be used for training. Instead, the undetected scenarios were used to check the accuracy of the results.

The structure of the DNN is shown in Fig. 4. In the training step, 70% of the detected data sets were used as training sets, 15% were used as validation sets, and 15% were used are test sets. There are eight input variables (the three (x, y, z) coordinates for the sensor position, two (x, y) coordinates indicating the source location, two (x, y) coordinates for the wind direction, and one value indicating the leak rate) and one output variable (detection time). The detection times used in the training step are all log scaled. It is important to predict the short detection time in DNN accurately because a short detection time has a greater effect in the MILP step. The main object of MILP phase is to find the position of the detectors, which makes the average detection time minimum. That means that optimization is determined by a combination of short detection times, and if these values change, the result could also change. However, even if the prediction of the long detection

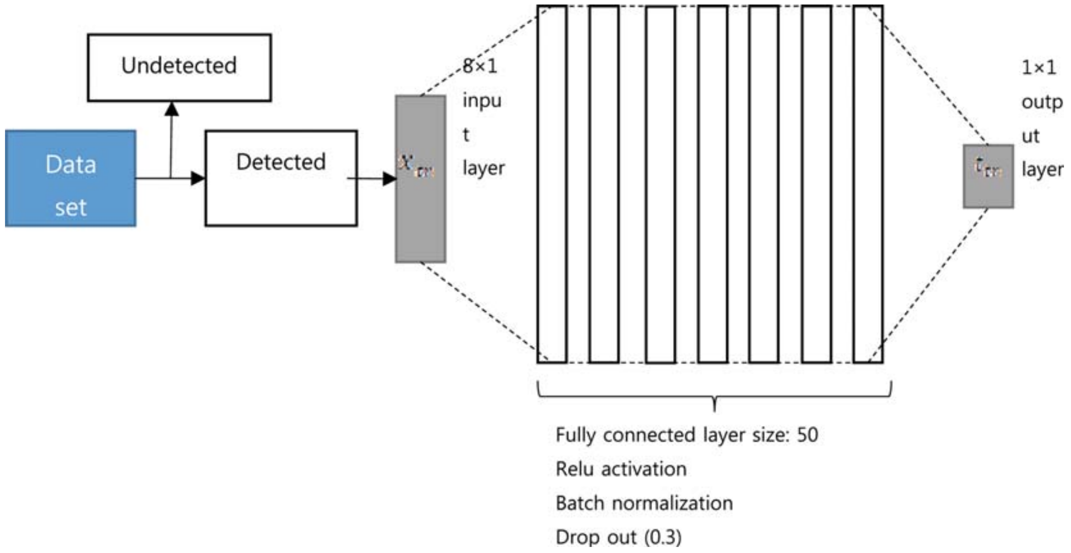

Fig. 4. Structure of deep neural network.

time is little inaccurate, it is still excluded from MILP (of course, if a large error occurs in long detection time prediction, it can affect the optimization result). Thus, the log scale is used to give weight to the accuracy of the short detection time.

A fully connected layer was used with batch normalization in the DNN training step. Having a drop-out ratio of 0.3, each layer was set to a size of 50. In the drop-out method, weights of unnecessary nodes are ignored by excluding a certain percentage of nodes for each training step. And, the ratio of randomly selected nodes to be excluded at each training step is called the drop-out ratio. That is, by eliminating unnecessary nodes in a sufficiently large layer size, overfitting is prevented For each training step, the number of iterations and the learning rate were set to 2880 and 0.0001. One important factor for regression is the number of hidden layers. Using too many hidden layers increases the likelihood of overfitting the model, while using too few hidden layers results in underfitting [21]. To determine the number of layers, a layer sensitivity analysis was performed. The root mean square error (RMSE) was calculated as the number of layers was increased, and the results are shown in Fig. 5. The RMSE values start to converge from five layers and have a minimum value of 0.163 at seven layers. Thus, we chose to use seven layers. The DNN was coded in MATLAB, and the "trainNetwork" routine was used as the training function.

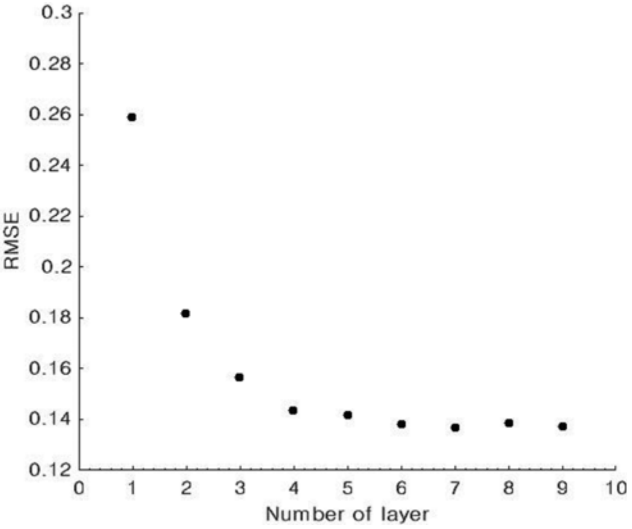

Fig. 5. Number of layer sensitivity analysis based on RMSE.

In the case of a DNN with seven layers, the regression took 4 min when using a single core of an Intel i5 (3.30 GHz) processor with 6 GB DDR3 RAM.

4. Detector Allocation Optimization

A detector allocation optimization problem was formulated using MILP. The equations for MILP follow the work of Legg [7], and details are given below. The variables are summarized in Table 1.

$$\min \Sigma_{a\in A}\alpha_a \Sigma_{i\in \mathcal{L}} \mathrm{d}_{a,i}\mathrm{x}_{a,i} \tag{1}$$

s.t.

$$\Sigma_{l\in L}\mathrm{s}_l \le \mathrm{p} \tag{2}$$

$$\mathrm{x}_{a,i}\le \mathrm{s}_l \qquad \forall \mathrm{a}\in \mathrm{A},\ \mathrm{i}\in \mathcal{L}_a \tag{3}$$

$$\Sigma_{i\in \mathcal{L}_a}\mathrm{x}_{a,i}=1 \qquad \forall \mathrm{a}\in \mathrm{A} \tag{4}$$

$$\mathrm{s}_l\in\{0,1\} \qquad \forall l\in \mathrm{L} \tag{5}$$

$$0\le \mathrm{x}_{a,i}\le 1 \qquad \forall \mathrm{a}\in \mathrm{A},\ \mathrm{i}\in \mathcal{L}_a \tag{6}$$

The objective function to find the minimum value of the mean detection time for each scenario, a, is shown in Eq. (6). The set A={1, 2, 3, …, M} means the set of leak scenarios, and the parameter α_a is the possibility of a leak scenario, a∈A. In this formulation, it is assumed that all scenarios occur with equal possibility, which is α_a=1/M. The set L={1, 2, 3, …, N} represents the potential sensor locations, and the subset $\mathcal{L}_a \subseteq$ L represents sensor locations affected by scenario a. The parameter $\mathrm{d}_{a,i}(\mathrm{i}\in\mathcal{L}_a)$ is the expected damage coefficient. Here, $\mathrm{d}_{a,i}$ is defined as the detection time of sensor i in scenario a, and it is pre-calculated in the CFD simulations. Because detection is triggered at values greater than 20% LFL of the leaked gas, some sensors fail to detect the gas. In this case, the damage coefficient value is set to the maximum simulation time (999 s). The parameter $\mathrm{x}_{a,i}$ is a binary variable and an indicator for location i that first detects scenario a. If the sensor at location i succeeds in detecting gas first, then $\mathrm{x}_{a,i}$ has a value of 1; otherwise, it is 0. Therefore, the term $\Sigma_{i\in\mathcal{L}_a}\mathrm{d}_{a,i}\mathrm{x}_{a,i}$ means the minimum time required to detect a leak in scenario a.

The term s_l is a binary variable and has a value of 1 if the sensor is present in potential position l or 0 otherwise. Constraint (2) ensures that the sum of all existing sensors cannot exceed p, which means the maximum number of detectors. Eq. (3) makes sure that if there is no existing sensor at location i, location i cannot be the first position to detect the gas. Eq. (4) means that at least one sensor succeeds in detecting the gas in scenario a.

Table 1. Problem notation

Symbol	Meaning
A={1, 2, 3, …, M}	Leak scenarios
L={1, 2, 3, …, N}	Potential detector locations
$\mathcal{L}_a$	Sensor locations affected by scenario a
α_a	The possility of leak scenario a
$\mathrm{d}_{a,i}$	Damage coefficient for leak scenario a at location i
s_l	Binary variable indicating whether a sensor is installed at location l or not
p	Maximum number of detectors
$\mathrm{x}_{a,i}$	Indicator for location i that first detects scenario a

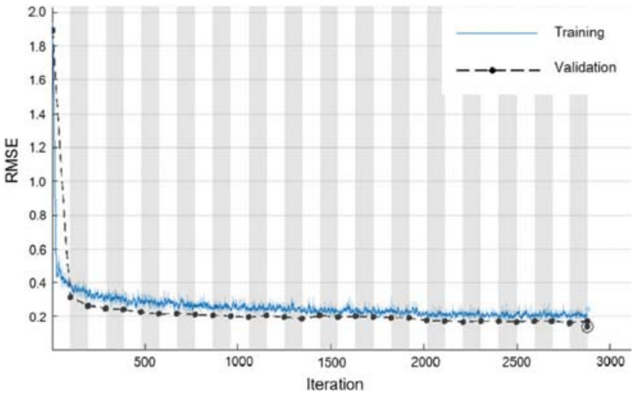

Fig. 6. RMSE change according to iteration during DNN training.

This MILP formulation was coded in MATLAB, and "intlinprog" was used as the solver. Intel i5 (3.30 GHz) processor with 6 GB DDR3 RAM was used as the computing resource for the present study. In total, 13 s were required to solve the MILP (based on original 30 scenarios) without parallel processing.

RESULTS AND DISCUSSION

1. DNN Results

All detected data sets were divided into the training set, validation set, and test set in a ratio of 0.7 : 0.15 : 0.15. The DNN model, which has seven layers, was trained in MATLAB based on the divided data set. After DNN regression had completed, the training aspect of the model was carried out to check for underfitting or overfitting. The RMSE value was used as a criterion for determining convergence, and the values for the training set and validation set were measured for each iteration. The results are shown in Fig. 6. The final iteration number was 2900, which was chosen through trial and error when determining the layer number. Because the gradient vanishing problem is reduced by applying batch normalization and rectified linear unit (Relu) activation functions, the RMSE value stabilizes quickly within 100 iterations. Thereafter, the RMSE value is slightly decreased, and there is almost no change after 1500 iterations. In addition, because the difference between the validation set and the training set is almost the same, the model is considered to be well fitted.

Next, to check the accuracy of the DNN model prediction, the detection time (t_v) of the validation set was compared with the detection time predicted by the DNN model (t_p) through the inputs of the validation set (x_v). The results are shown in Fig. 7. When t_v and t_p are regressed to the center line, most of the data are well centered. In particular, the t_v values are more accurately predicted in intervals less than 50 s than longer intervals. This is considered to be an effect of the scaling method, which focuses on short detection times. The data set was regressed to the t_p=t_v line to check the accuracy of the model numerically, and the R^2 is 0.897. Thus, the results obtained by applying log scaling are better than those obtained by the commonly used normalization scaling method (R^2= 0.880).

To check the effect of data scaling, the DNN model was trained without scaling under the same conditions, including the number of layers. As shown in Fig. 8, the data is shown to be more distributed around the center than that of the case when the scaling was applied. In particular, the predicted result tends not to fall below a certain value (28 s), which is an effect arising from the bias generated within the model.

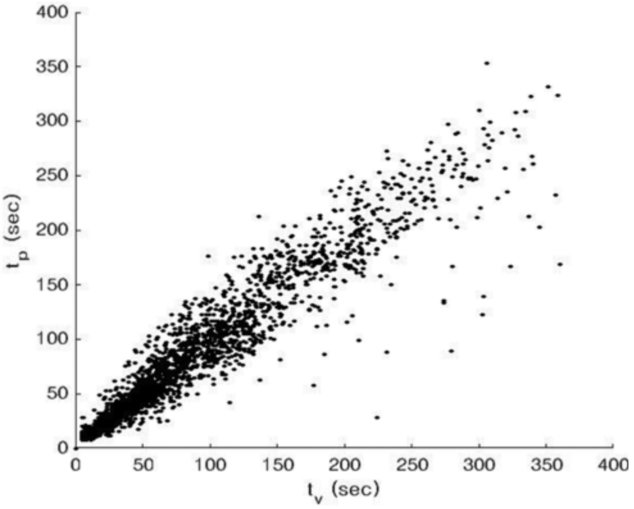

Fig. 7. t in validation set (t_v) vs predicted t (t_p).

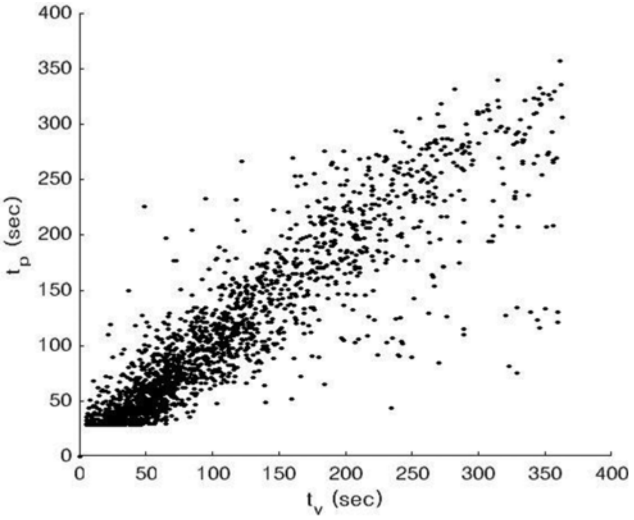

Fig. 8. t_v vs t_p without scaling.

Next, it is necessary to confirm that the DNN model can predict results accurately, including undetected cases. In the case of undetected cases, a detection time value is not generated because the concentration of LNG gas at the location does not exceed 20% LFL. However, the model must yield a detection time as a result. Thus, it is impossible to obtain detection failure in the model, so we defined an undetected scenario. This was defined as when the detection time generated by the DNN model exceeded a predetermined value. A confusion matrix was used to evaluate the performance of the model to classify detected and undetected scenarios. An interval of 40 s was used as a criterion for classification because detection times exceeding 40 s do not affect MILP optimization.

Table 2. Confusion matrix of the trained network

		Predicted result		Overall accuracy
		Undetect (0)	Detect (1)	
Target	Undetect (0)	True Negative 42446 (89.2%)	False Positive 402 (0.8%)	99.1%
	Detect (1)	False Negative 230 (0.5%)	True Positive 4532 (9.5%)	95.2%
Overall accuracy		99.5%	91.85%	98.7%

The results are shown in Table 2. In the matrix, false positive indicates detected cases in the DNN model but not in the CFD model. The false positive rate (the proportion of all negatives in CFD that still yield positive outcomes in DNN model) is about 0.9%. In contrast, false negative means cases detected in the CFD model but not in the DNN model. The false negative rate (the proportion of positives in CFD which yield negative outcomes in DNN model) is about 4.8%. The overall accuracy is 98.7%.

2. Sensor Allocation Result

To solve the MILP for sensor allocation optimization, a detection time at each position is required. The DNN model was used to generate the detection times required for MILP. However, it is important to determine how many extended samples should be generated. As the number of samples increases, the reliability of MILP also increases. Too many samples, however, increases the load of the optimization calculations, thus losing the advantages of this method. The mean detection time, i.e., the result of MILP, was calculated with an increased number of samples to determine an appropriate sample number. Samples were generated through LHS by changing five variables (two (x, y) coordinates for the source location, two (x, y) coordinates for the wind direction, and one value for the leak rate), as for the original data, and the numbers are 100, 300, 500, and 1,000, respectively. This was combined with 1587 potential sensor locations (x, y, z coordinates) and input to the DNN. The number of detectors deployed in the MILP was fixed at 30. The results are shown in Fig. 9. Since the results do not change significantly after the number of samples exceeds 300, this number of samples is sufficient.

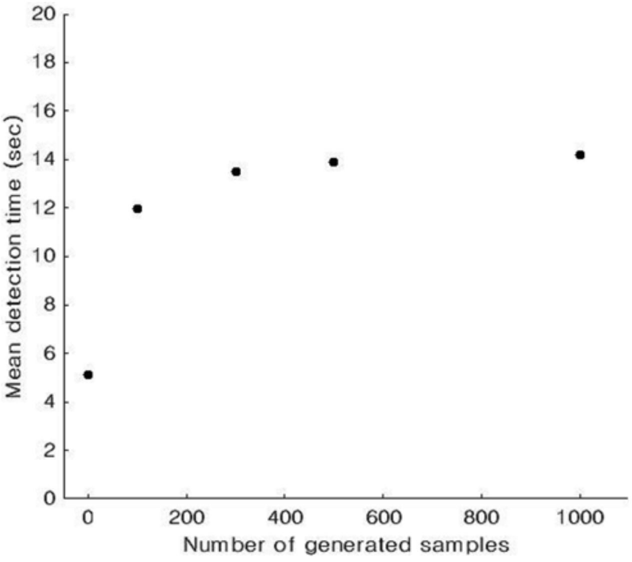

Fig. 9. Mean detection time according to number of samples.

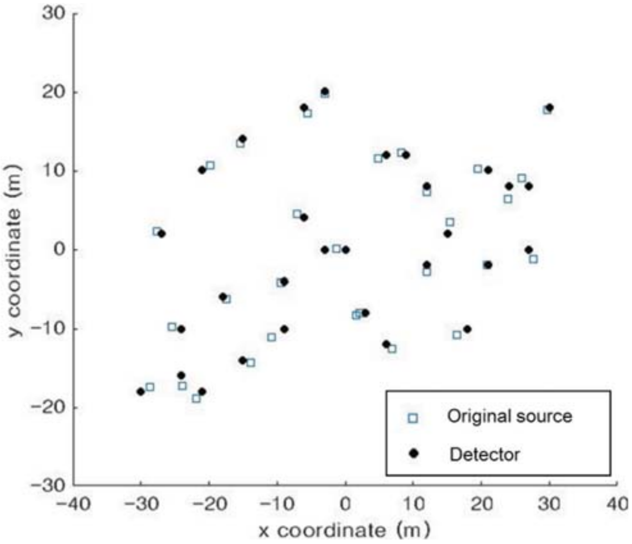

Fig. 10. Optimization result with 30 scenarios.

The MILP formulation was solved by applying the 300 generated data points. To see how the increase in the number of samples affected the optimization, the result was compared to the MILP result based only on the original 30 scenarios. The optimization results for 30 detectors are shown in Figs. 10 and 11. In the result based on the original scenarios, because the number of sources and detectors is the same, the detectors are installed at the same positions, as shown in Fig. 10. However, if the number of samples is sufficiently large, the detectors are positioned to cover all cases, as shown in Fig. 11; in particular, detectors are located in areas that the LHS did not cover, such as the lower right and upper left.

Fig. 12 shows how the detection time decreases when the posi-

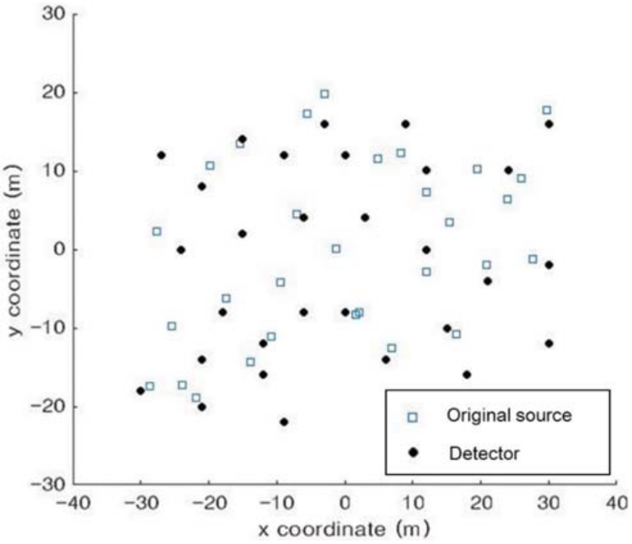

Fig. 11. Optimization result with extended scenarios.

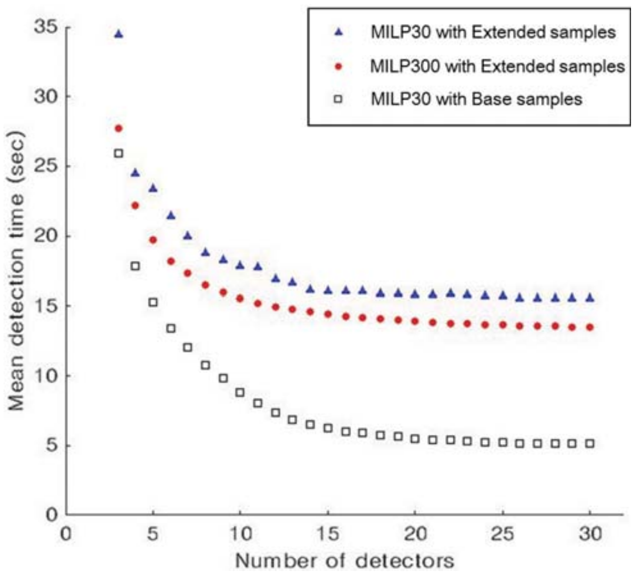

Fig. 12. Results of mean detection time.

tion of the detectors is changed. The mean detection time is the average minimum detection time measured for each sample, which is measured by increasing the number of detectors from 3 to 30. MILP30 indicates the optimized detector positions based on the base sampling, and MILP300 indicates the optimized positions of the detectors achieved by adding 300 generated samples. When MILP30 is applied to the base samples, the detection time is the lowest because the detectors are located at the source positions. However, the results are different when additional samples are applied. The detector positions based on the base samples converged in 17.5 s on average using extended sampling, which is 2 s slower than the 15.5 s required when the detector positions were based on the extended sampling.

CONCLUSION

This paper presents a method to avoid insufficient sampling while reducing computational load for gas detector allocation optimization. Thirty CFD based samples were generated in FLACS from an input set generated by LHS. In each sample, the detection time was measured at 1587 potential detector locations. Based on this dataset, a surrogate model based on a deep neural network was developed, and the data set was divided into a training set, validation set, and test set in a ratio of 0.7 : 0.15 : 0.15. The number of layers for the DNN was determined through sensitivity analysis by measuring the RMSE value with increasing layer number. To verify the accuracy of the DNN regression, the results of the validation set were compared with the DNN results generated using the input conditions of the validation set. The detector locations were optimized by solving an MILP problem formulated based on the results generated by the DNN. As a result, the average minimum detection time was reduced by 2 s, which is a 12% reduction, compared to the case without a surrogate model.

The proposed surrogate model method for optimizing detector location is advantageous when the generation of base scenarios takes a long time, as well as when the number of samples is low. In our surrogate model, the calculation time is reduced. Thus, this study contributes to reducing computational costs. In the future, state-of-the-art machine learning models and sampling methods will be used to yield more accurate results. In addition, if methods that can quantitatively calculate the probability of an accident are developed in the future, they can be applied to generate accident scenarios more realistically.

ACKNOWLEDGEMENT

This work was supported by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (No. 20162220100030).

NOMENCLATURE

ε : dissipation of turbulent kinetic energy [m^2/s^3]
k : turbulent kinetic energy [m^2/s^2]
L : potential detector locations
A : leak scenarios
$\mathcal{L}_a$: sensor locations affected by scenario a
α_a : probability of leak scenario a
$d_{a,i}$: damage coefficient for leak scenario a at location i
p : maximum number of detectors
s_l : binary variable indicating whether a sensor is installed at location *l* or not
$x_{a,i}$: indicator for location i that first detects scenario a

Abbreviations

DNN : deep neural network
CFD : computational fluid dynamics
DoE : design of experiment
FLACS : flame acceleration simulator
LFL : lower flammability limit
LHS : latin hypercube sampling
LNG : liquefied natural gas
MILP : mixed integer linear programming
RMSE : ROOT mean square error
QRA : quantitative risk analysis
RANS : reynolds averaged navier stokes

REFERENCES

1. S. R. Hanna, M. J. Brown, F. E. Camelli, S. T. Chan, W. J. Coirier, S. Kim, O. R. Hansen, A. H. Huber and R. M. Reynolds, *Bull. Amer. Meteorol. Soc.*, **87**, 1713 (2006).
2. S. R. Hanna, O. R. Hansen, M. Ichard and D. Strimaitis, *Atm. Environ.*, **43**, 262 (2009).
3. K. J. Long, F. J. Zajaczkowski, S. E. Haupt and L. J. Peltier, *JCP*, **4**, 881 (2009).
4. Z. T. Xie, P. Hayden and C. R. Wood, *Atm. Environ.*, **71**, 64 (2013).
5. D. Hamel, M. Chwastek, B. Farouk, K. Dandekar and M. Kam, Proceedings of the 2006 IEEE International Workshop on, 38 (2006).
6. J. Berry, W. E. Hart, C. A. Phillips, J. G. Uber and J.-P. Watson, *J.*

Water Res. Plan. Man., **132**, 218 (2006).

7. S. W. Legg, A. Benavides-Serrano, J. D. Siirola, J.-P. Watson, S. Davis, A. Bratteteig and C. D. Laird, *Comput. Chem. Eng.*, **47**, 194 (2012).
8. S. W. Legg, C. Wang, A. J. Benavides-Serrano and C. Laird, *J. Loss Prev. Process. Ind.*, **26**, 410 (2013).
9. A. J. Benavides-Serrano, S. W. Legg, R. Vázquez-Román, M. Mannan and C. D. Laird, *Ind. Eng. Chem. Res.*, **53**, 5355 (2013).
10. A. J. Benavides-Serrano, M. Mannan and C. D. Laird, *AIChE J.*, **62**, 2728 (2016).
11. A. J. Benavides-Serrano, M. Mannan and C. D. Laird, *J. Loss Prev. Process. Ind.*, **35**, 339 (2015).
12. S. Davis, O. R. Hansen, F. Gavelli and A. Bratteteig, Using CFD to Analyze Gas Detector Placement in Process Facilities. In: GexCon (2015).
13. R. Vázquez-Román, C. Díaz-Ovalle, E. Quiroz-Pérez and M. S. Mannan, *J. Loss Prev. Process. Ind.*, **44**, 633 (2016).
14. E. G. Gomes, R. de Andrade Medronho and J. V. B. Alves, Gas Detector Placement in Petroleum Process Unit Using Computational Fluid Dynamics. International Journal of Modeling and Simulation for the Petroleum Industry, 8 (2014).
15. K. Wang, T. Chen, S. T. Kwa, Y. Ma and R. Lau, *Comput. Chem. Eng.*, **69**, 89 (2014).
16. L. Margheri and P. Sagaut, *J. Comput. Phys.*, **324**, 137 (2016).
17. J. Na, K. Jeon and W. B. Lee, *Chem. Eng. Sci.*, **181**, 68 (2018).
18. B. E. Launder and D. B. Spalding, *Comput. Meth. in Appl. Mech. Eng.*, **3**, 269 (1974).
19. GexCon AS, FLACS v10.7 Users Manual (2017).
20. O. R. Hansen, F. Gavelli, M. Ichard and S. G. Davis, *J. Loss Prev. Process. Ind.*, **23**, 857 (2010).
21. I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press (2016).