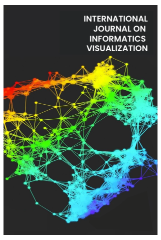
INTERNATIONAL JOURNAL ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv



Computational of Concrete Slump Model Based on H2O Deep Learning Framework and Bagging to Reduce Effects of Noise and Overfitting

Stefanus Santosa [a,*], Yonathan P. Santosa [b,a], Garup Lambang Goro [a], Wahjoedi [a], Jamal Mahbub [a]

[a] *Department of Civil Engineering, Politeknik Negeri Semarang, Indonesia*
[b] *Department of Informatics Engineering, Soegijapranata Catholic University, Indonesia*
Corresponding author: [*]stefanus.santosa@polines.ac.id

***Abstract*— Concrete mixture design for concrete slump test has many characteristics and is mostly noisy. Such data will affect the prediction of machine learning. This study aims to experiment on the H2O Deep Learning framework and Bagging for noisy data and to overfit avoidance to create the Concrete Slump Model. The data includes cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate, age, slump, and compressive strength. Primary data for a concrete mixed design using the fine aggregate material from Merapi Volcano, the hills of Muntilan, and Kalioro. The coarse aggregate was obtained from Pamotan, Jepara, Semarang, Ungaran, and Mojosongo Boyolali Central Java. The cement used Gresik and Holcim products, and the water was from Tembalang, Semarang. The experiment model with one input layer with 7 neurons, one hidden layer with 20 neurons, and one output layer with 1 neuron using activation function TanH, with parameter L1=1.0E-5, L2=0.0, max weight=10.0, epsilon=1.0E-8, rho=0.99, and epoch=800 can achieve RMSE of 2.272. This result shows that after introducing Bagging, the error can be reduced up to 2.5 RMSE, approximately (50% lower) compared to the model without Bagging. The manually tested mixture data was used to model evaluation. The result shows that the model was able to achieve RMSE 0.568. Following this study, this model can be used for further research, such as creating slump design practicum equipment/ application software.**

***Keywords*—Concrete slump design; deep learning; data mining; bagging; overfitting; prediction model.**

Manuscript received 31 Aug. 2022; revised 17 Oct. 2022; accepted 12 Nov. 2022. Date of publication 30 Jun. 2023.

I. INTRODUCTION

Concrete is a mixture of complex materials like cement, water, coarse aggregates, and fine aggregates, with various effects of characteristics of river sand, dune sand, crushed sand [1], and chemical components and mineral admixtures mixed by some formula to improve the quality. Therefore, it becomes difficult to predict the concrete slump with these complex materials [2], [3]. The slump in value is very important. Vakhshouri and Nejadi [4] state that slump also affects the compressive strength of concrete.

Concrete slump prediction research using conventional methods is still popular based on British Standard, American Concrete Institute Method (ACI), etc. [5]. Conventional methods tend to rely on an analysis model which requires ideal conditions and high precision measurements, but those requirements are often hard to fulfill. Thus, a more advanced method for slump testing method must be developed. In the construction industry, digitization, new materials and technologies, and advanced automation are now becoming new trends [6], [7].

The most reliable approach to solving the complex problem of the concrete slump prediction model is by using computational intelligence for data mining. The model can also include machine learning in data mining, similar to how humans solve problems by training and implementing or using evolutionary learning. Those approaches can also be called soft computing approaches. Soft computing deals with imprecise, uncertain, partial truth, and approximation to achieve practicability, robustness, and low solution cost [8] which often happens when designing concrete slump. Soft computing approaches are more accurate than statistical approaches. In their research, Chine concludes that Artificial Neural Network (ANN) is more accurate in predicting concrete slump value than multiple linear regression in concrete mixture design [9]. Chopra also formulated a similar conclusion for ANN that ANN is more accurate in producing estimation than Decision Tree (DT) and Random Forest (RF) [10], but Feng et al. [11] declared that DT more accurate than ANN and Support Vector Machine (SVM). These results, including [9], show that Backpropagation Neural Network (BPNN) is practical for concrete compressive strength prediction [12]. From that preliminary research, ANN shows a good potential to solve concrete mix design problems. Hence, ANN was used as the baseline model further to improve the robustness of the concrete slump model.

State-of-the-art concrete mix design for concrete slump and compressive strength prediction model computation is an evolutionary and deep learning model approach. The evolutionary and deep learning model is more advanced in solving computational problems than other approaches. Wang proposed that Wavelet Neural Networks (WNN) estimation algorithm could analyze and estimate the concrete compressive test data [13]. Deng *et al.* [14] used Convolution Neural Network (CNN) with lower error than BPNN and SVM. Even though this is the case, CNN or CNN Modified [15] is useful only when the positional and spatial information of a certain feature in the data is important such as in image processing [16], [17] or signal processing cases [18], [19]. Another type of neural network, Recurrent Neural Networks (RNNs), is very suitable for dealing with time series data [20].

H2O Deep Learning framework (H2O) will be used to create the model [21] without convolutional layers and max-pooling layers of CNN or RNN. H2O works better than RNN for transactional data because RNN is strong in sequential or time series data [22]. To improve performance of the model, ensemble approaches like Random Forests and Gradient Boosting Regression Trees [23], and Bagging [18], Smoothed Bootstrap Resampling [26], [27] could be used to reduce the negative effect of inherent noise [28] and overfitting. Li states that Random Forests and Gradient Boosting Regression Trees do not improve the prediction quality [23]. Dahiya *et al.* [24] state in their research that their feature selection-enabled hybrid Bagging algorithm (FS-HB) performed best compared with fewer features and tree-based classifier for qualitative datasets. Its performance on numeric data was also better than other standalone classifiers.

Concrete mixture design data for concrete slump tests has many characteristics and is very noisy because of the diversity of origins of the concrete materials. This data will affect the accuracy of the model prediction. The use of H2O Deep Learning without an ensemble method is supposedly unable to overcome the data noise and avoid overfitting. This study aims to experiment with Bagging in case of noisy data and overfitted model. In order to do so, two models with and without Bagging will be tested where overfitting exists and where overfitting does not exist. We used Cross Validation to reduce the prevalence of robust overfitting in adversarial training [29], [30]. All of the produced models will be evaluated and the model with the least error (lowest Root Mean Square Error- RMSE) will be used as a baseline to create virtual slump test tools, practicum equipment, or application software.

II. Materials and Method

In order to achieve the optimum Computational of Concrete Slump Model, this research will be conducted as follows.

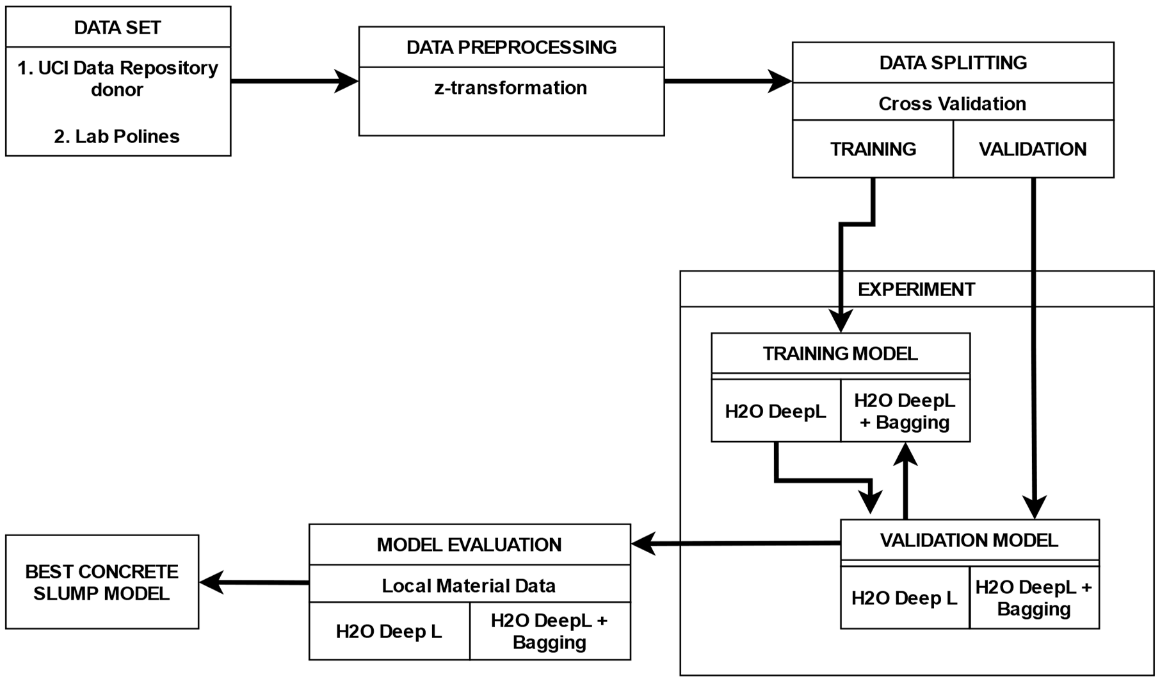


Fig. 1 Research Method

A. Data Collection

Secondary data was collected from UCI Data Repository data set created by I-Cheng Yeh [31], and primary data from Material Laboratory of Politeknik Negeri Semarang. The data consists of the concrete mixed design: cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, fine aggregate, and age.

B. Data Pre-processing and Splitting

Because of the variety of measurements in the data, the data will be pre-processed with z-transform (see Fig.1). Note that the output will not be pre-processed to preserve the model so that it can output the same measurement as the data. The mean μ and the standard deviation σ for each variable are first calculated. Afterward, all the data input is transformed with the following function.

$$x_{new} = \frac{x_{old} - \mu}{\sigma} \tag{1}$$

Before the experiment process was done, the dataset was shuffled and partitioned into k partitions or k subset of data. This data was then split into train set and a validation set using k-fold Cross Validation [32]. One partition is used as the validation set, and the rest (k-1) is the training set.

C. Experiment

The H2O architecture model was designed by testing the number of hidden layers and the most optimal number of neurons to obtain the lowest possible error. Firstly, several hyperparameters were defined, while weights and biases were randomly generated. The hyperparameters are epoch, L1, L2, epsilon, and rho. Secondly, the model was trained by using AdaDelta learning algorithm with the training dataset. Activation functions ReLU and TanH are used as transfer function for each layer to avoid vanishing gradients. Then, the prediction will be used to calculate the loss and backpropagated to optimize the bias and weight.

The H2O neural network model that is used in this research contains one input layer, several hidden layers, and one output layer. Given the input X with n row data (batch of data) represented as a row matrix, and the weight is a matrix W of size (next_layer × prev_layer), each layer was calculated with simple matrix multiplication as follows,

$$\boldsymbol{X} = \begin{bmatrix} \overrightarrow{x_1} \\ \overrightarrow{x_2} \\ \vdots \\ \overrightarrow{x_n} \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix} \tag{2}$$

$$net_h = \boldsymbol{X} \times \boldsymbol{W}^{\top} \tag{3}$$

$$o_h = \sigma(net_h) \tag{4}$$

where net_h is neuron summation and o_h is the output of the neuron summation with activation function applied to it. The activation function $\sigma(\cdot)$ is a non-linear activation function, in this case ReLU [33] and TanH, that are calculated as follows [34],

$$ReLU(x) = \max(0, x) \tag{5}$$

$$TanH(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{6}$$

With our configuration, for instance, 2 hidden layers, the calculation will be as follows:

$$\begin{aligned} t_{h1} &= \boldsymbol{X} \times \boldsymbol{W}_{(i,h_1)}^{\top} + \boldsymbol{b_1} \\ h_1 &= \sigma(net_{h1}) \\ net_{h2} &= \boldsymbol{X} \times \boldsymbol{W}_{(h_1,h_2)}^{\top} + \boldsymbol{b_2} \\ h_2 &= \sigma(net_{h2}) \\ net_o &= \boldsymbol{X} \times \boldsymbol{W}_{(h_2,o)}^{\top} + \boldsymbol{b_o} \\ y &= \sigma(net_o) \end{aligned} \tag{7}$$

Bagging (bootstrap aggregating) was used to improve the performance and robustness of the model over noisy data. The model was replicated b times, and each model was trained using randomly selected data with replacement [35]. By bootstrapping the dataset, each model will adapt to its own sets of data. Take a dataset *D,* for example, we can augment b new datasets from *D*.

$$\begin{gathered} d_i \subset D \\ 1 \le i \le b \\ |d| = |D| \end{gathered} \tag{8}$$

We will assume that there is a dataset $\hat{d} \subset D$ which no noisy data are occurring. The fact that each dataset d is selected randomly with replacements, with enough number of different datasets, we eventually augmented $d_{i\in|b|} \approx \hat{d}$. If data $\hat{d}$ is used to train a model, generalization effect should occur thus reducing the chance of overfitting by noisy data. This step is important because in general, it is hard to single out noise in the data, especially with high dimensional data.

After finishing with Bagging, the result of each model was aggregated with min-pooling based on the RMSE, hence we singled out the model with better sets of data to achieve the lowest error. The validation data set is used to test the current parameters for overfitting. The effect of Bagging was tested by training two models, with and without Bagging, by comparing the RMSE (see Fig. 1).

The error was calculated using RMSE function, calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y} - y)^2} \tag{8}$$

with $\hat{y}$ is the target, y is the predicted output, and N is the size of the minibatch.

D. Model Evaluation

The evaluation phase is the stage of testing the best model of training and validation results with hyperparameters that have been obtained with and without Bagging. The model with the lowest RMSE obtained from this evaluation stage is the Concrete Slump Model, which is ready for the next research about virtual machine development for concrete slump design.

III. Results and Discussion

A. Initial Architecture of Concrete Slump Model

The number of hidden layers (HL) and the number of neurons (N) in each hidden layer are the parameters that determine the architecture of the expected model. Based on the preliminary experiment stage, this experiment is to find the optimal number of neurons with the activation function of TanH, ReLU, Maxout, and with or without Bagging. For other parameters the default parameters from the library were used. This experiment shows that the best training RMSE of the H2O Deep Learning without Bagging Model with activation function TanH is 6.0807934, while ReLU 6.3655844, and Maxout 5.798217 (Fig. 2)

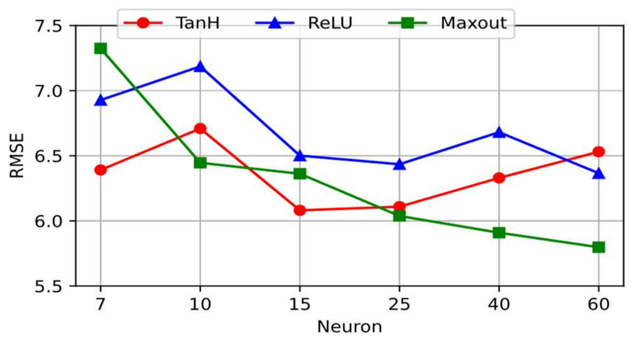


Fig. 2 Optimal number of neurons H2O without Bagging

However, the model with Bagging shows that the RMSE value TanH is 5.957209, ReLU 5,278, and Maxout 4.788992 (Fig.3). The results show that the RMSE tends to decrease when the number of neurons reaches 40-60.

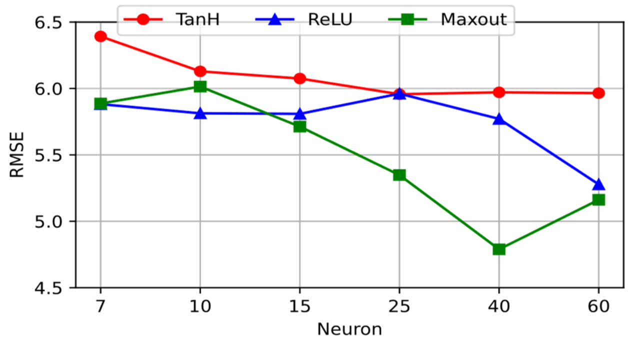


Fig. 3 Optimal number of neurons H2O with Bagging

Based on these experiments, the next step is to find the optimum hidden layer from the proposed model, which lies between 40-60 neurons. In other words, 50 neurons should be used, with ReLU, Maxout, and TanH, with or without Bagging. The hyperparameter used is L1= $1.e^{-5}$, L2=0, epsilon=0.99, and rho = $1.e^{-8}$. The test results from H2O with 50 Neurons architecture (Fig. 4) and H2O+Bagging with 50 Neurons architecture (Fig. 5) show that RMSE values vary on different hidden layers.

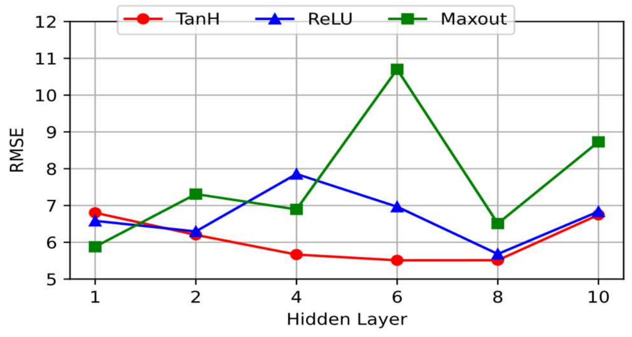


Fig. 4 H2O with 50 Neurons

These results depend on the activation function. The model with the Maxout activation function tends to produce lower RMSE in 1-2 hidden layers but is unstable, while ReLU with 6-9 hidden layers and TanH with 4-6 layers, each of which used 50 neurons. In order to achieve a non-complex architecture model with low computation cost, the experiments with 6 hidden layers and 50 neurons were not continued further. The next study is done for the architecture with one hidden layer with 50 neurons using ReLU and TanH activation function.

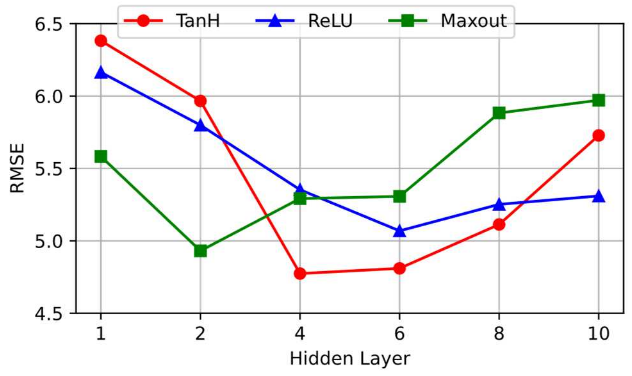


Fig. 5 H2O+Bagging with 50 Neurons

This experiment shows that TanH activation function performs better than ReLU but is insignificant. The error rate decreases from epoch 100, starts to flatten at around epoch 100-500, and is relatively stable at epoch 500-800. This shows that learning outcomes begin to be effective at epoch 500 and reach their optimal value at epoch 750-800 (Fig. 6).

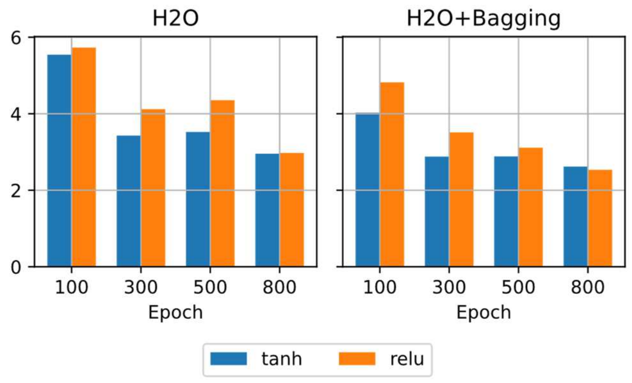


Fig. 6 Number of Epoch

Trials conducted on Epoch 1000 or more did not show better performance. Large epoch value causes the network to train longer with no significant results. The previous experiments need to be completed by experimenting with the best number of neurons in the architecture with 1 hidden layer with the TanH and ReLU activation function only, and with or without Bagging.

B. *Optimize Architecture of Concrete Slump Model*

The next experiment was carried out to obtain the optimal architecture of Concrete Slump Model. This experiment was performed on H2O framewprk with or without Bagging with the number of neurons tested was 10, 20, 30, 40, 50, 60, and 80, 100 at epoch 800. The first test was carried out on 1HL10N architecture, translating to 1 hidden layer and 10 neurons. Based on the Cross Validation algorithm with a partition value k = 10, training was done with data from (k-1) or 9 partitions. The result of training with the maximum epoch of 800 has the lowest error value of 6.542. Bagging has the ability to increase stability and avoid overfitting by reducing variance and noise. The role of Bagging has been proven across all architectures with lower RMSE values than without Bagging.

1) The effects of TanH vs ReLU

Both activation functions show trends of error reduction when the number of neurons is increased on H2O architecture (Fig, 7) and H2O+Bagging architecture (Fig, 8). In ReLU case, the sudden jumps can be observed in neuron size of 30. This means, ReLU requries larger number of neurons to achieve smaller errors.

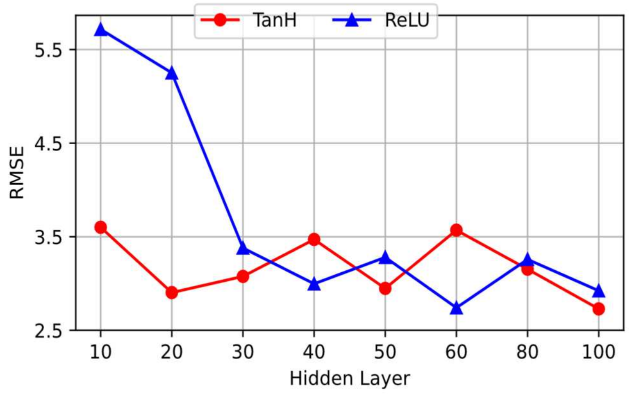


Fig. 7 Training error H2O with TanH and ReLU activation function

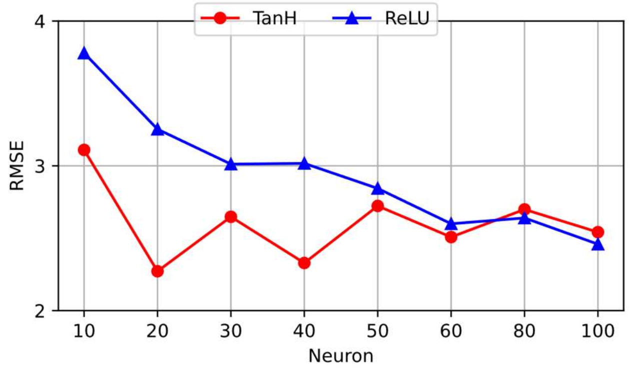


Fig. 8 Training error H2O+Bagging with TanH and ReLU activation function

Though the training process of both models is promising, we see the error increases as the number of neurons is also increased for ReLU case (see Fig. 9 and Fig.10). This is a sign of overfitting.

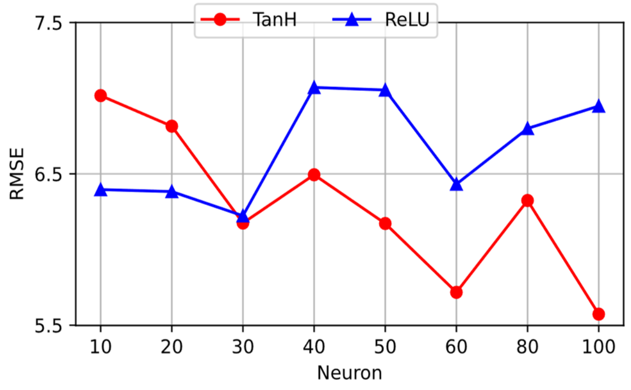


Fig. 9 Validation Error H2O with TanH and ReLU activation function

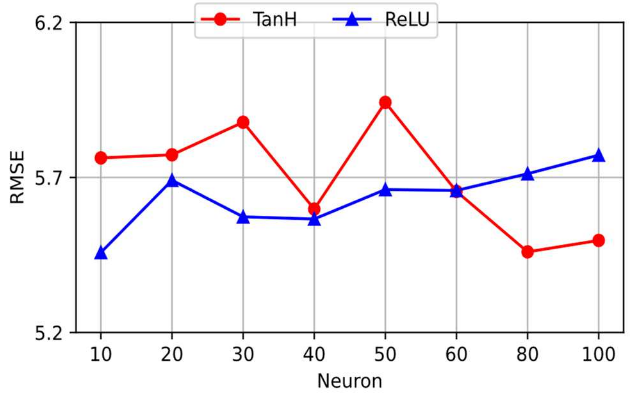


Fig. 10 Validation Error H2O+Bagging with TanH and ReLU activation function

2) The effects of TanH with Bagging

In TanH case, the model does not show an indication of overfitting. By using Bagging, the error variance becomes smaller allowing us to use a lower number of neurons for training (Fig. 11). This also means that we can reduce the computational cost further. By evaluating the validation data (Fig. 12), we can conclude that we can use a lower number of neurons if we use Bagging for training which might not be obvious in the training process.

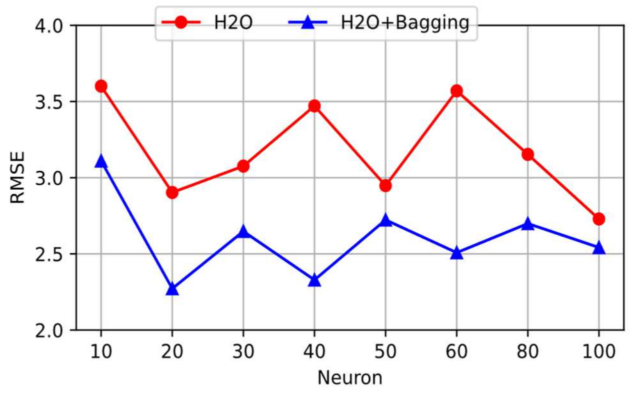


Fig. 11 Training Error with TanH

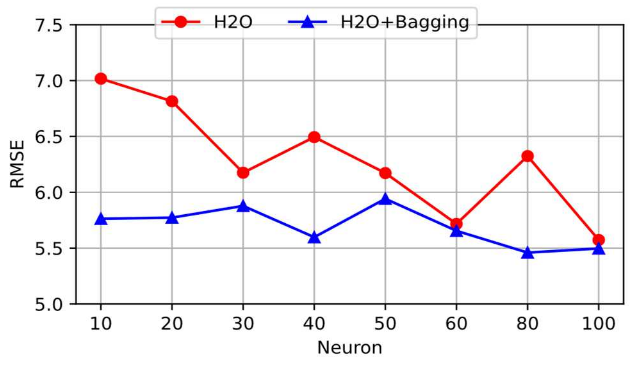


Fig. 12 Validation Error with TanH

3) The effects of ReLU on Bagging

Without Bagging, ReLU requires larger neurons until it reaches the optimum value (see Fig. 13). The biggest improvement can be seen if Bagging was used to train the model. The training results improve, especially for the lower number of neurons.

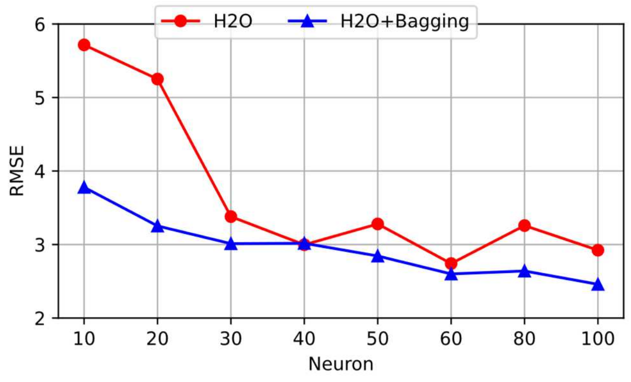


Fig. 13 Training Error with ReLU

Without Bagging, the difference between a number of neurons is more prominent compared to with Bagging. This shows that Bagging can help avoid overfitting, as shown in comparison with ReLU model (Fig. 14). Even though both models show a sign of overfitting, the model with Bagging successfully dampened the effects.

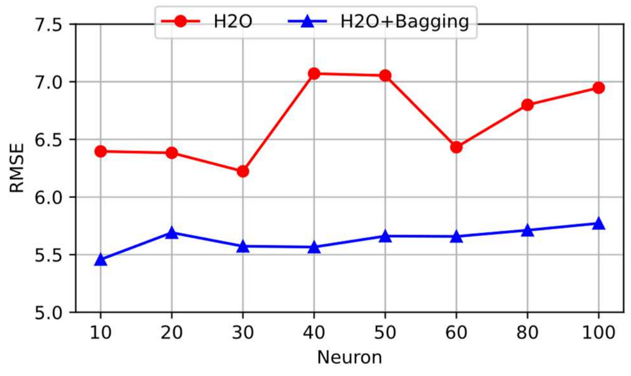


Fig. 14 Validation Error H2O with ReLU

C. Model Evaluation Result

The best-performing concrete slump model was obtained by using 7 input neurons, 1 hidden layer with 20 neurons, and 1 output neuron using H2O Deep Learning Framework with Bagging and TanH activation function. The training parameter was L1 = $1e^{-5}$, L2 = 0.0, max weight = 10.0, $\epsilon = 1e^{1-8}$, $\rho = 0.99$, and epoch = 800 resulting in training RMSE = 2.272.

Concrete mix design data used as testing data to evaluate the model is unseen data by the model. In order to collect the data, we created a new mixture design using the fine aggregate which originated from Merapi Volcano, the hills of Muntilan, and Kalioro Central Java. The coarse aggregate was obtained from Pamotan, Jepara, Semarang, Ungaran, and Mojosongo Boyolali Central Java. The cement was used by Gresik and Holcim products, and the water was from Tembalang, Semarang, Central Java. All the previously mentioned materials were used to produce several mixed designs. The test mixture was molded, cured for 28 days, then manually tested using compressive test machine, resulting in compressive strength of 25 MPa. The manually tested mixture data was used to test the computational Concrete Slump Model. The evaluation shows that the model was able to achieve RMSE 0.568. This test shows that the model could perform well enough, shown by a small error. Hence, this model can be further developed into virtual concrete slump test/ application software.

D. Discussion

From the test series above, both models with TanH and ReLU gives acceptable results regarding the Concrete Slump Model. Both are able to predict with RMSE lower than 3 MPa. With TanH, no signs of overfitting were detected, but when Bagging was introduced to the model, the RMSE continued to drop, achieving better performance across multiple neurons. On the other hand, ReLU shows signs of overfitting. By increasing the number of neurons in 1 hidden layer architecture, we introduce a more complex model, increasing the model variance. As the model variance exceeds the data requirements, the model will fail to predict new data. Hence overfitting occurs. It can be seen from the results that once the number of neurons for ReLU model increases, the model continues to produce higher and higher RMSE. In conclusion, TanH performs better than ReLU on the slump dataset because of overfitting.

To further study the effects of Bagging on an overfitted model, we will look at the results of the ReLU model. We have established that the ReLU model suffers from an overfitting problem. After introducing Bagging to the model, the error was reduced to 2.5 RMSE approximately (50% lower) compared to the model without Bagging. Based on this experiment, we concluded that Bagging can significantly reduce the effects of overfitting.

IV. Conclusion

After a series of tests performed in the concrete slump dataset, each model can predict new data with error to 7 RMSE. TanH activation function performed better than ReLU in predicting concrete slump value. Furthermore, the signs of overfitting can be observed using ReLU activation function. As the results suggest, the overfitting effects can be reduced significantly by using Bagging. The next research this Concrete Slump Model can be applied to build slump design practicum equipment/ application software in a virtual laboratory for civil engineering vocational students.

Acknowledgment

The Research Center and Community Service supported this work under Grant SP DIPA- BLU Nomor 023.18.2.677603/2022 from the Featured Research Areas of Politeknik Negeri Semarang,

References

[1] A. Attia, S. Guettala, and R. Zaitri, “Using mixture design method to optimizing concretes characteristics made with binary and ternary sands,” *World Journal of Engineering*, vol. 18, no. 2, pp. 194–205, Jan. 2021, doi: 10.1108/WJE-05-2020-0184.

[2] M. Timur Cihan, “Prediction of Concrete Compressive Strength and Slump by Machine Learning Methods,” *Advances in Civil Engineering*, vol. 2019, p. e3069046, Nov. 2019, doi: 10.1155/2019/3069046.

[3] N. M. Tuan, Q. V. Hau, S. Chin, and S. Park, “In-situ concrete slump test incorporating deep learning and stereo vision,” *Automation in Construction*, vol. 121, p. 103432, Jan. 2021, doi: 10.1016/j.autcon.2020.103432.

[4] B. Vakhshouri and S. Nejadi, "Prediction of compressive strength of self-compacting concrete by ANFIS models," *Neurocomputing*, vol. 280, pp. 13–22, Mar. 2018, doi: 10.1016/j.neucom.2017.09.099.

[5] C. E. Ekinci, "The Calculation Methods of Compound Of Concrete and A Novel Calculation Method," *Technological Applied Sciences*, vol. 1, no. 1, Art. no. 1, Feb. 2006, doi: 10.12739/10.12739.

[6] T. Wangler, N. Roussel, F. P. Bos, T. A. M. Salet, and R. J. Flatt, "Digital Concrete: A Review," *Cement and Concrete Research*, vol. 123, p. 105780, Sep. 2019, doi: 10.1016/j.cemconres.2019.105780.

[7] J. B. Pereira and G. F. Maciel, "Automated slump test: An effective alternative in predicting rheological properties and an efficient tool for providing the quality control of materials," *Measurement*, vol. 178, p. 109384, Jun. 2021, doi: 10.1016/j.measurement.2021.109384.

[8] P. Gupta and N. N. Kulkarni, "An Introduction of Soft Computing Approach over Hard Computing," 2013.

[9] W.-H. Chine, L. Chen, H.-H. Hsu, T.-S. Wang, and C.-H. Chiu, "Modeling Slump of Concrete Using the Artificial Neural Networks," in *2010 International Conference on Artificial Intelligence and Computational Intelligence*, Oct. 2010, vol. 3, pp. 236–239. doi: 10.1109/AICI.2010.287.

[10] P. Chopra, R. K. Sharma, M. Kumar, and T. Chopra, "Comparison of Machine Learning Techniques for the Prediction of Compressive Strength of Concrete," *Advances in Civil Engineering*, Apr. 12, 2018. https://www.hindawi.com/journals/ace/2018/5481705/ (accessed Jul. 09, 2020).

[11] D.-C. Feng *et al.*, "Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach," *Construction and Building Materials*, vol. 230, p. 117000, Jan. 2020, doi: 10.1016/j.conbuildmat.2019.117000.

[12] C. Liang, C. Qian, H. Chen, and W. Kang, "Prediction of Compressive Strength of Concrete in Wet-Dry Environment by BP Artificial Neural Networks," *Advances in Materials Science and Engineering*, Apr. 02, 2018. https://www.hindawi.com/journals/amse/2018/6204942/ (accessed Jul. 09, 2020).

[13] H. Wang, X. Wang, C. Wang, and J. Xu, "Concrete Compression Test Data Estimation Based on a Wavelet Neural Network Model," *Mathematical Problems in Engineering*, Feb. 11, 2019. https://www.hindawi.com/journals/mpe/2019/4952036/ (accessed Jul. 09, 2020).

[14] F. Deng, Y. He, S. Zhou, Y. Yu, H. Cheng, and X. Wu, "Compressive strength prediction of recycled concrete based on deep learning," *Construction and Building Materials*, vol. 175, pp. 562–569, Jun. 2018, doi: 10.1016/j.conbuildmat.2018.04.169.

[15] R. Hidayat, Hendrick, Riandini, Z.-H. Wang, and H. Gwo-Jiun, "Mask RCNN Methods for Eyes Modelling," *International Journal of Data Science*, vol. 2, no. 2, Art. no. 2, Dec. 2021, doi: 10.18517/ijods.2.2.63-68.2021.

[16] Mambang and F. D. Marleny, "Image Prediction of Exact Science and Social Science Learning Content with Convolutional Neural Network," *JOIV : International Journal on Informatics Visualization*, vol. 6, no. 4, Art. no. 4, Dec. 2022, doi: 10.30630/joiv.6.4.923.

[17] G. S. Saragih, Z. Rustam, D. Aldila, R. Hidayat, R. E. Yunus, and J. Pandelaki, "Ischemic Stroke Classification using Random Forests Based on Feature Extraction of Convolutional Neural Networks," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 10, no. 5, p. 2177, Oct. 2020, doi: 10.18517/ijaseit.10.5.13000.

[18] S. Albawi, T. Abed Mohammed, and S. ALZAWI, "Understanding of a Convolutional Neural Network," Aug. 2017. doi: 10.1109/ICEngTechnol.2017.8308186.

[19] S. S. Berutu, Y.-C. Chen, H. Wijayanto, and H. Budiati, "A Conversion of Signal to Image Method for Two-Dimension Convolutional Neural Networks Implementation in Power Quality Disturbances Identification," *JOIV: International Journal on Informatics Visualization*, vol. 6, no. 4, Art. no. 4, Dec. 2022, doi: 10.30630/joiv.6.4.1529.

[20] D.-X. Zhou, "Universality of deep convolutional neural networks," *Applied and Computational Harmonic Analysis*, vol. 48, Jun. 2019, doi: 10.1016/j.acha.2019.06.004.

[21] M. A. Ghorbani, F. Salmasi, M. K. Saggi, A. S. Bhatia, E. Kahya, and R. Norouzi, "Deep learning under H2O framework: A novel approach for quantitative analysis of discharge coefficient in sluice gates," *Journal of Hydroinformatics*, vol. 22, no. 6, pp. 1603–1619, Sep. 2020, doi: 10.2166/hydro.2020.003.

[22] F. Sirait, M. T. Bin Jusoh, K. Dimyati, and M. F. Bin Md Din, "Determining Optimal Zone Radius of Zone Routing Protocol Based on Deep Recurrent Neural Networks in the Next Generation Wireless Backhaul Networks," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 12, no. 5, pp. 2147–2155, 2022, Accessed: Jan. 23, 2023. [Online]. Available: http://ijaseit.insightsociety.org/index.php?option=com_content&view=article&id=9&Itemid=1&article_id=15747.

[23] X. Li, R. Bai, P.-O. Siebers, and C. Wagner, "Travel time prediction in transport and logistics: Towards more efficient vehicle GPS data management using tree ensemble methods," *VINE Journal of Information and Knowledge Management Systems*, vol. ahead-of-print, Jun. 2019, doi: 10.1108/VJIKMS-11-2018-0102.

[24] S. Dahiya, S. S. Handa, and N. Singh, "A feature selection enabled hybrid-bagging algorithm for credit risk evaluation," *Expert Systems*, vol. 34, p. e12217, May 2017, doi: 10.1111/exsy.12217.

[25] M. Injadat, A. Moubayed, A. B. Nassif, and A. Shami, "Multi-split Optimized Bagging Ensemble Model Selection for Multi-class Educational Data Mining," *arXiv:2006.05031 [cs]*, Jun. 2020, doi: 10.1007/s10489-020-01776-3.

[26] H. Hartono and E. Ongko, "Avoiding Overfitting dan Overlapping in Handling Class Imbalanced Using Hybrid Approach with Smoothed Bootstrap Resampling and Feature Selection," *JOIV : International Journal on Informatics Visualization*, vol. 6, no. 2, Art. no. 2, Jun. 2022, doi: 10.30630/joiv.6.2.985.

[27] C. E. Chandra, S. Abdullah, and S. Devila, "Estimating Indonesian Complete Life Table and Fair Annual Pure Premium Range from Abridged Life Table with Bayesian Method and Bootstrapping," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 12, no. 6, pp. 2226–2236, 2022, Accessed: Jan. 23, 2023. [Online]. Available: http://ijaseit.insightsociety.org/index.php?option=com_content&view=article&id=9&Itemid=1&article_id=14947.

[28] Z. Nematzadeh, R. Ibrahim, A. Selamat, and V. Nazerian, "The synergistic combination of fuzzy C-means and ensemble filtering for class noise detection," *Engineering Computations*, vol. ahead-of-print, Mar. 2020, doi: 10.1108/EC-05-2019-0242.

[29] L. Rice, E. Wong, and Z. Kolter, "Overfitting in adversarially robust deep learning," in *Proceedings of the 37th International Conference on Machine Learning*, Nov. 2020, pp. 8093–8104. Accessed: Jan. 24, 2023. [Online]. Available: https://proceedings.mlr.press/v119/rice20a.html.

[30] T. Chen, Z. Zhang, S. Liu, S. Chang, and Z. Wang, "Robust Overfitting May Be Mitigated by Prop- Erly Learned Smoothening," 2021.

[31] I.-C. Yeh, "Modeling slump flow of concrete using second-order regressions and artificial neural networks," *Cement and Concrete Composites*, vol. 29, no. 6, pp. 474–480, Jul. 2007, doi: 10.1016/j.cemconcomp.2007.02.001.

[32] S. Khaki and D. Nettleton, *Conformal Prediction Intervals for Neural Networks Using Cross Validation*. 2020.

[33] T. Ergen and M. Pilanci, *Training Convolutional ReLU Neural Networks in Polynomial Time: Exact Convex Optimization Formulations*. 2020.

[34] A. Panigrahi, A. Shetty, and N. Goyal, *Effect of Activation Functions on the Training of Overparametrized Neural Nets*. 2019.

[35] A. Giessing and J. Fan, "Bootstrapping lp-Statistics in High Dimensions," *arXiv:2006.13099 [econ, math, stat]*, Jun. 2020, Accessed: Jul. 10, 2020. [Online]. Available: http://arxiv.org/abs/2006.13099 .