Utilizing High Performance Computing to Improve the Application of Machine Learning for Time-Efficient Prediction of Buildings’ Daylighting Performance

Rania Labib

Department of Architecture, School of Architecture, Prairie View A&M University
(A member of the Texas A&M system), Texas, USA

ralabib@pvamu.edu

Abstract. Architects often investigate the daylighting performance of hundreds of design solutions and configurations to ensure an energy-efficient solution for their designs. To shorten the time required for daylighting simulations, architects usually reduce the number of variables or parameters of the building and facade design. This practice usually results in the elimination of design variables that could contribute to an energy-optimized design configuration. Therefore, recent research has focused on incorporating machine learning algorithms that require the execution of only a relatively small subset of the simulations to predict the daylighting and energy performance of buildings. Although machine learning has been shown to be accurate, it still becomes a time-consuming process due to the time required to execute a set of simulations to be used as training and validation data. Furthermore, to save time, designers often decide to use a small simulation subset, which leads to a poorly designed machine learning algorithm that produces inaccurate results. Therefore, this study aims to introduce an automated framework that utilizes high performance computing (HPC) to execute the simulations necessary for the machine learning algorithm while saving time and effort. High performance computing facilitates the execution of thousands of tasks simultaneously for a time-efficient simulation process, therefore allowing designers to increase the size of the simulation’s subset. Pairing high performance computing with machine learning allows for accurate and nearly instantaneous building performance predictions.

1. Introduction

Building energy consumption accounts for about 30% of global energy consumption [1]. In order to support energy-efficient built environment design, planners, architects, engineers, and designers began exploring the energy performance of their designs using a new set of increasingly popular building performance simulation tools. These tools are often integrated in parametric modeling environments, which allows for parametric geometric modeling that helps designers to produce alternative
configurations of their initial design using various parameters to control geometric and constructive aspects of architectural models [2].

The integration of building performance simulations tools in parametric modeling environments facilitates the exploration of various building performance criteria such as daylighting, thermal comfort, and energy consumption. Investigating the performance of all variations to select the most energy-efficient configuration is crucial in the early design phase. However, this process is often considered complex and time-consuming. For example, the annual daylighting performance of one building design configuration usually requires manual effort to prepare the model file for this type of simulation. Additionally, the simulation processing can take one to two hours on a desktop computer. Similarly, annual glare analysis can take a few hours to calculate the daylight glare probability value for each daylight hour during the year (equal to 4380 hours). It becomes almost impossible to carry out large-scale simulations to analyze the performance of various configurations due to time constraints related to project deadlines [3]. Therefore, there is high demand for frameworks that could speed up the simulation process.

There has been an increased interest in the application of machine learning (ML) to predict various performance aspects of the built environment such as energy consumption, daylighting harvesting, and thermal comfort. To predict the performance of various configurations of a parametric model, instead of running performance simulations for each configuration, ML algorithm requires the execution of only a small subset of the building performance simulations (BPS), therefore speeding up the process of BPS. Researchers have found that ML algorithms, specifically the deep learning algorithm known as neural networks (NN), accurately predict various performance aspects of the built environment [4–6]. For example, Neto and Fiorelli used a NN model to predict the thermal performance of the administration building of the University of Sao Paulo. The researchers propose NN showed a fair agreement between the predicted energy consumption values and the actual values, with an average error of about 10% [7]. Similarly, Wong et al. examined the use of a NN model to predict the energy consumption and artificial lighting levels loads of an office building. The researchers used a parametric building model that had nine variables as the input parameters, four of which are related to the external weather conditions (daily average dry-bulb temperature, daily average wet-bulb temperature, daily global solar radiation, and daily average clearness index), four variables represented the building envelope designs (solar aperture, daylight aperture, overhang, and side-fins projections), and one variable that is a day-type variable (i.e., weekdays, Saturdays, and Sundays). The NN model was used to estimate daily electricity use for cooling, heating, and electric lighting. The accuracy metric for the NN-modeled cooling, heating, electric lighting, and total building electricity use was 0.994, 0.940, 0.993, and 0.996, respectively, indicating the excellent strength of the model’s predictive ability [6].

Although NN allows the use of a small subset of the BPS, executing a small subset of daylighting simulations can be a time-consuming process. Therefore, the use of a HPC environment becomes a necessity when applying a NN framework to a parametric model. HPC facilitates the execution of various computing processes in parallel on individual computing nodes that are part of a computing cluster that is usually hosted on the cloud. Researchers have shown that HPC provides an economical solution for executing large-scale computing processes. [8–10].

2. Limitations of the existing BPS process

- Time limitation: To determine the optimal building design solution to support energy-efficient design, the existing BPS can be time-consuming, leading to architects abandoning the BPS processes to adhere to project deadlines.
- Simulation preparation and storage: It is often difficult to carry out large-scale BPS due to the need to prepare numerous files of a complex structure. Most BPS engines require files to be organized in specific way and uniquely named, making this process almost impossible to complete in a manual manner. Finally, storing the files on a local hard drive can become troublesome due to the large size of the files.
• Data analytics and visualization: The amount of data obtained from carrying out BPS of parametric models is usually very large. Hence, it is almost impossible to manually visualize the results, given that BPS results of each design configuration are stored in individual files. Additionally, the BPS results are often stored in a raw format, and thus additional computations are needed to convert the data into a format that the non-technical user can interpret in the context of the built environment performance of the examined design configuration.

3. Methodology

The geometric model
For the purpose of this paper, a small parametric model of a 4x5m office room was used to carry out illuminance and daylight autonomy simulations. The office location was set to New York City. The model geometric model contains only one window oriented in different ways (see Table 1). Ten different parametric variables were assigned to the model to produce thousands of configurations of the office (Table 1). Some parameters, such as the ceiling height and the lightshelf depth, are geometrically related; other parameters, such as the ceiling reflectance values and transmittance values, are related to the optical properties of the construction material. Each parameter contains a set of different values, with total of 25 different values that lead to 5,120 unique design configurations.

Table 1. A total of 25 model variables used to create various design configurations of the office room.

| Variable             | Configuration | CT |
|----------------------|---------------|----|
| Room Hight           | 3m, 4m        | 2  |
| Glazing Ratio        | 50%, 60%      | 2  |
| Lightshelf Depth     | 0.5m, 1m      | 2  |
| Lightshelf Location  | Top of the window, shifted | 2 |
| Walls Reflectance    | 50%, 60%, 70%, 80%, 90% | 5 |
| Lightshelf reflectance| 80%, 90%     | 2  |
| Ceiling Reflectance  | 80%, 90%      | 2  |
| Floor reflectance    | 50%, 70%      | 2  |
| Glazing transmittance| 30%, 50%      | 2  |
| Window orientation   | North, South, East, West | 4 |
| Total number of room configurations | 5,120 |

The daylighting simulations
In order to use an ML algorithm to predict the daylighting performance of the various room configurations, it was necessary to simulate the daylighting performance of some configurations to use as an ML training set. Therefore, the performance simulations of 506 random room configurations were carried out. Additionally, to examine the accuracy of the proposed ML algorithm to predict one output and multi-output values, two types of daylighting simulations were carried out: point-on-time illuminance simulations and annual daylight autonomy (DA) simulations.

3.1.1. One-output point-on-time simulations
Point-in-time daylighting simulations were carried out for the randomized 506 room configurations. These simulations were used to calculate the average illuminance value of the illuminance levels at
144 sensor points in the office room, and the simulations’ time and date were set to winter solstice at 12:00.

3.1.2. Multiple-outputs DA simulations
Annual daylighting simulations were used to calculated daylight autonomy (DA) values over a grid of 144 sensor points (multiple outputs, 144).

3.2 High Performance Computing
An initial illuminance and DA simulation process for one configuration was performed on a mobile workstation equipped with an i7-7700HQ processor with a processing speed of 2.80 GHz (up to 3.80GHz with Turbo Boost technology) and 16 GB of RAM. The DA simulation took two minutes to complete, and the illuminance level simulation took 0.25 minutes to complete. DA simulations take more time than illuminance simulations do; this is mainly because the calculation procedure of DA determines the hourly illuminance level in all 144 analysis grid points in the test room for the entire year [11]. It was evident that running 5,120 configurations could be a complex and time-consuming task; therefore, the NN algorithm is crucial for speeding up such a process since the NN model uses only a subset of the simulations to predict the outcome. Two small subsets of 506 (illuminance, and DA) simulations of random room configurations were used to train the ML algorithm. Although NN allowed the use of a small subset, executing 506 illuminance and DA simulations would still be a time-consuming process, taking about two hours and 16 hours, respectively. Therefore, a HPC (a.k.a., supercomputer) environment was used to execute simulations. HPC facilitates the execution of various commands and processes in parallel on individual computing nodes that are part of a computing cluster. Various researchers confirmed that HPC provides an economical solution for executing large-scale computing processes [8–10]. Initial test simulations were performed on one HPC node. The illuminance calculations took about 10 seconds, and the DA simulation took about 65 seconds. Thus, it was clear that one node was much faster than performing the simulation on one computer, and by utilizing the parallel computing framework such as HPC, simulations could be significantly sped up.

In this study, the Extreme Science and Engineering Discovery Environment (XSEDE), which is a cloud-based HPC environment that serves scientists and researchers in the United States [12]. The technical specifications of the HPC system are listed in Table 2. The HPC system used for the simulations consists of 1944 computing nodes that can run tasks simultaneously. When compared to a workstation, a node has three times the processing power and two to three times the memory. To put that into perspective, each computing node has 24 processing cores and 128 GB of RAM; meanwhile, most modern workstations are equipped with 8 processing cores and between 32 and 64 GB of RAM.

### Table 2. Technical specifications of the XSEDE HPC cloud-based system.

| System component          | Configuration          |
|---------------------------|------------------------|
| Node count                | 1944                   |
| Processor type            | Intel Xeon E5-2680v3   |
| Sockets                   | 2                      |
| Cores per node            | 24                     |
| Clock speed               | 2.5 GHz                |
| Flop speed                | 960 GFlop/s            |
| Memory capacity per node  | 128 GB DDR4 DRAM       |
| Flash memory              | 320 GB SSD             |
| Memory bandwidth          | 120 GB/s               |
| STREAM Triad bandwidth    | 104 GB/s               |

Both subsets of 506 simulations were executed on the HPC environment using a method similar to the one introduced in 2019 by Labib and Baltazar [13], which consists of the following steps (see Figure 1):
Batch file preparation: To prepare the files required for calculating both illuminance and DA on the HPC environment, Honeybee was used to create two sets of 506 batch and scene files. The batch files contained Radiance commands that allow the simulation to be run at a later time on any machine if the machine has Radiance installed. The scene files contained information about the geometric model and its Radiance materials. The process of creating the batch and scene files was automated using Grasshopper sliders, which are automation tools that are native to the Grasshopper environment.

Automated bash file upload: The produced batch files were uploaded to the cloud based HPC environment using Python scripts that automate the upload process via FTPS, which is a secure file transfer protocol.

Formatting the batch files: The batch files resulting from Honeybee contained windows-based Radiance commands. However, the HPC environment is Linux-based. Therefore, it was necessary to convert the windows-based Radiance commands into Linux-based Radiance commands, which are executed later from bash files (compatible with Linux-based platforms). This process was completed by a custom python script that was written specifically for this purpose.

Grouping the Batch files: A custom python script facilitated the process of grouping the commands into 120 groups for execution on 120 computing nodes, which were utilized in parallel. The illuminance simulations of the subset were completed in roughly one minute (1.1 minutes), and the DA simulations took about 8.5 minutes.

Figure 1. The workflow used for executing both illuminance and DA simulations on the cloud based HPC environment. The blocks in the gray shaded area are processes that are executed on the HPC, while the other blocks contain processes that are executed on a local machine.

3.3 The NN model
A NN framework was applied to the datasets that resulted from both sets of 506 simulations. The proposed NN model consists of the following processes:

- Data Normalization: Both data samples contained only 506 simulations which were split between 404 training samples that are used to train the NN model, and 102 test samples that are used to validate the NN model. Each feature (e.g., the lightshelf height) in the dataset had a different scale. For instance, some values were proportions, which take values between 0 and 1; others take values between 50 and 80, others between 3 and 4, and so on. Therefore, a data
normalization technique was used to unify the scale of all the features in the dataset and to make sure that they are centred around zero.

- Developing the NN model: A small NN model was constructed with two intermediate layers, each with 64 units, and one layer that contained only one output unit. The model was compiled with the loss function mean squared error (MSE), which is the square of the difference between the predictions and the targets. To monitor the absolute value of the difference between the predictions and the targets, the mean absolute error (MAE) was calculated.

- Validating the ANN Approach with K-fold Cross validation: To evaluate the NN model while adjusting parameters, such as the number of epochs, the number of layers, and the number of units in each layer, the validation dataset is used to validate the model. However, the validation dataset in this case is very small. Therefore, the MAE and the MSE might change every time the model’s parameter are changed; this is caused by the choice of data points used for validation and training, which leads to high variance in the MAE and MSE scores, making the evaluation process of the model unreliable. To mitigate this problem, the author used the K-fold cross-validation method, which consists of splitting all available data into K-partitions and training the model on all partitions except one used for validation. Then the process was repeated by cycling through all the partitions. The validation score for the model used then was the average of the K validation scores obtained from all the cycles. For this work, K = 4 was used to apply the K-fold cross-validation method.

4. Results

4.1 The single-output illuminance simulations

Applying the proposed NN model to the illuminance simulations dataset, the calculated performance metrics MSE and MAE to dataset resulted from the point-on-time illuminance simulations with 130 epochs produced the best results before the model started to overfit. The MSE and MAE values were equal to 0.015 and 0.094, respectively. This means that the predicted average illuminance values are 94 points off the actual value (considering that the MAE was multiplied by 1,000 because the initial average illuminance values were divided by 1,000). The NN reduced the time required to examine more than 5,000 room configurations from about a couple of hours to a few minutes with an error margin of 0.94%. To validate the proposed NN model, the model was applied to nine different room configurations to examine the accuracy of the resulted predictions. shows the predicted average illuminance values compared to the simulated values that were produced by implementing the NN model. Figure 2 shows the distribution of the simulated data versus the predicted data.

4.2 The single-output illuminance simulations

Similar to the validation method used to examine the accuracy of the single-output model, the author applied the NN model to compare the predicted DA values to their respective simulated values. The MSE and MAE values were equal to 7.6 and 14.8, respectively. The predicted DA values were plotted against the actual values that are resulted from simulations (see Figure 4). Figure 3 shows the distribution of the simulated data versus the predicted data.
5. Conclusion

This paper examined the use of HPC to speed up the application of NN in the context of daylighting simulations. The proposed workflow was used to predict the daylighting performance of 5,120 different room configurations. The NN model was examined for two cases, to predict one output, the average illuminance value inside the room, and multiple outputs, or 144 DA values. Training data was populated by executing 506 simulations of random room configurations. Running daylighting simulations is time-consuming, and it was evident that running 506 simulations was not practical. Therefore, this author automated the execution of all 506 simulations on an HPC cluster that contains 120 computing nodes that facilitated the execution of simulations in parallel. The utilization of the HPC environment facilitated executing the simulations needed to obtain the training data in a time-efficient manner. The illuminance simulations were completed in roughly one minute (1.1 minutes), and the DA simulations took about 8.5 minutes. The execution of these simulations on a desktop computer would have normally taken two hours and 16 hours, respectively.

In case of the point-on-time average illuminance value, the NN model’s calculated loss functions: MSE was equal to 0.015 and MAE was equal to 0.094. The NN model showed highly accurate results, where the average illuminance values of the validated configurations were within 95 points of the actual value. Considering that the average illuminance values of the validated configurations ranged from around 2000 to 3000 lux, a 95-point range of error is negligible.

Similarly, the NN model was examined to predict multiple outputs: 144 DA values of the analysis grid points. The MSE and MAE values were equal to 7.6 and 14.8, respectively. The predicted DA values were plotted against the actual values that are resulted from simulations (see Figure 4).

In conclusion, the proposed NN model showed impressive results in predicting single and multiple outputs, although it was observed that the NN model showed higher accuracy in predicting single outputs. In addition to accuracy, coupling the HPC with NN increased time efficiency.
Figure 4. Daylight autonomy maps of one room configuration. Right, the NN-predicted DA map, Left, the simulated DA map.

Acknowledgement
This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562.

References
[1] IEA, Key World Energy Statistics 2014, Organisation for Economic Co-operation and Development, 2014. https://doi.org/10.1787/key_energ_stat-2014-en.
[2] K. Terzidis, Algorithmic architecture, Architectural Press, 2006.
[3] A.-T. Nguyen, S. Reiter, P. Rigo, A review on simulation-based optimization methods applied to building performance analysis, Appl. Energy. 113 (2014) 1043–1058. https://doi.org/10.1016/j.apenergy.2013.08.061.
[4] S.L. Wong, K.K.W. Wan, T.N.T. Lam, Artificial neural networks for energy analysis of office buildings with daylighting, Appl. Energy. 87 (2010) 551–557. https://doi.org/10.1016/j.apenergy.2009.06.028.
[5] H. Zhao, F. Magoulès, A review on the prediction of building energy consumption, Renew. Sustain. Energy Rev. 16 (2012) 3586–3592.
[6] S. Zhou, D. Liu, Prediction of Daylighting and Energy Performance Using Artificial Neural Network and Support Vector Machine, Am. J. Civ. Eng. Archit. Am. J. Civ. Eng. Archit. 3 (2015) 1–8. https://doi.org/10.12691/ajcea-3-3A-1.
[7] A.H. Neto, F.A.S. Fiorelli, Comparison between detailed model simulation and artificial neural network for forecasting building energy consumption, Energy Build. 40 (2008) 2169–2176.
[8] L. Pérez-Lombard, J. Ortiz, C. Pout, A review on buildings energy consumption information, Energy Build. 40 (2008) 394–398. https://doi.org/10.1016/j.enbuild.2007.03.007.
[9] D. Thain, T. Tannenbaum, M. Livny, Distributed computing in practice: the Condor experience, Concurr. Comput. Pract. Exp. 17 (2005) 323–356.
[10] Y. Zhai, M. Liu, J. Zhai, X. Ma, W. Chen, Cloud versus in-house cluster: evaluating Amazon cluster compute instances for running MPI applications, in: ACM, 2011: p. 11.
[11] C.F. Reinhart, J. Mardaljevic, Z. Rogers, Dynamic Daylight Performance Metrics for Sustainable Building Design, LEUKOS. 3 (2006) 7–31. https://doi.org/10.1582/LEUKOS.2006.03.01.001.
[12] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G.D. Peterson, XSEDE: accelerating scientific discovery, Comput. Sci. Eng. 16 (2014) 62–74.

[13] R. Labib, J.-C. Baltazar, Automated Execution of Large-Scale Daylighting and Glare Simulations in a Cloud-Based Parallel Computing Environment, in: 16th IBPSA Int. Conf., Rome, Italy, 2019: p. 7.