energies

MDPI

Article

A Procedure for Automating Energy Analyses in the BIM Context Exploiting Artificial Neural Networks and Transfer Learning Technique

Mikhail Demianenko * and Carlo Iapige De Gaetani *

Department of Civil and Environmental Engineering, Politecnico di Milano, 20133 Milan, Italy
* Correspondence: mikhail.demianenko@mail.polimi.it (M.D.); carloiapige.degaetani@polimi.it (C.I.D.G.)

Abstract: One of the main benefits of Building Information Modelling is the capability of improving the decision-making process thanks performing what-if tests on digital twins of the building to be realized. Pairing BIM models to Building Energy Models allows designers to determine in advance the energy consumption of the building, improving sustainability of the construction. The challenge is to consider as many elements involved in the energy balance as possible and shuffling their parameters within a certain range. In this work, the automatic creation of a relevant set of design options to be analyzed for searching the optimum has been carried out. Firstly, the usual workflow that would be applied manually has been automatically followed by running scripts and codes, depending just on the initial setup given by the user. Although the procedure is very resource consuming, the main advancement relies in the reduction of the manual intervention and the possibility of creating large datasets of design options, avoiding gross errors. Secondly, Artificial Neural Networks and Transfer Learning techniques are applied to speed up the process of dataset creation. With such approach, the same dataset has been created, with about 30% of initial data and without significant loss of accuracy.

Keywords: BIM; design optioonering; energy analyses; process automation; Artificial Neural Networks; transfer learning

Citation: Demianenko, M.; De Gaetani, C.I. A Procedure for Automating Energy Analyses in the BIM Context Exploiting Artificial Neural Networks and Transfer Learning Technique. *Energies* **2021**, *14*, 2956. https://doi.org/10.3390/en14102956

Academic Editor: Frede Blaabjerg

Received: 20 April 2021
Accepted: 18 May 2021
Published: 20 May 2021

1. Introduction

Building Information Modeling (BIM) has been used to transform the Architectural Engineering Construction (AEC) industry or rather the way buildings are designed, constructed, maintained operational and even dismissed or renewed. In the whole life cycle of a building, BIM technology is also being used to provide accurate, timely and relevant information, dramatically improving the effectiveness of asset management [1]. By shifting efforts in the early stages of a project, BIM enables designers to model structures before they are built with great impact on decision-making of all involved stakeholders. In fact, by exploiting the BIM approach and related technologies, architects, engineers, and building owners can explore multiple options when planning for or manage facilities, infrastructure, and environment. By creating a virtual model and applying different techniques, materials, and designs, prior to the beginning of construction, reliable predictions can be made about cost, functionality, stability and more [2].

Today, most stakeholders understand the implications and possibilities enabled by using BIM as a tool to facilitate information flow and support better decision-making throughout the building lifecycle. Among all the aspects, sustainability has raised the profile of building lifecycle management [3]. The importance of long-term sustainable construction in our current social climate is indisputable. BIM technology is a tool that is being used to help make the AEC industry more economically and environmentally sustainable [4].

In this context, digital twins of buildings are tied to Building Energy Models (BEM) allowing designers to perform what-if tests to determine the energy consumption of the building, helping reduce waste and overall building costs. When BEM tool is integrated within the BIM application and workflow, energy analyses can occur as early as the concept stage, allowing for more informed design decisions [5]. Energy modeling tools use a range of parameters to estimate the energy performance of a building. Some of this information, e.g., building envelope, zones, rooms, structure, equipment, control scenarios, can often come directly from the BIM system. However, for comprehensive energy modeling, a more detailed level of information on energy characteristics is needed [6]. BIM model is made up of various categories of 'objects', which include each unit of equipment in the building. Objects can include different levels of detail, often provided by the equipment manufacturer, which needs to be in a format the specific BIM system can understand [7]. For effective energy modeling, the designer needs to include the right amount of information about each piece of equipment or component, including at the very least: power consumption, heating and cooling capacity (for HVAC equipment), thermal characteristics and behavior (for passive systems).

The challenge is to consider as many elements involved in the energy balance as possible and shuffling their parameters in a plausible and meaningful range [8]. The permutation of all considered design components and energy properties realize a set of combinations among which the optimal solution must be searched. This optimization problem can then be solved under different perspectives (e.g., minimum cost, best performance and many others) [9–11]. Further complicating things is the fact that BIM design typically uses multiple tools, each with a different focus, and separate BEM tools for energy modeling. This requires data to be transferred between the BIM and BEM models, which is more difficult and time-consuming when the tools are not integrated [12,13] or used through a platform that manages them jointly [14]. Standards exist that enable the extraction and sharing of data between BIM and BEM tools, including the Industry Foundation Classes (IFC) and Green Building XML (gbXML). However, these require a middleware conversion utility, and there is room for potential errors along the way or missing data during the conversion process [15–19].

In terms of environmental sustainability, many factors are involved when computing the energetic balance of a building, also interacting one to each other differently. Several studies analyze this aspect, exploiting BIM technology and its capability of virtualizing the design and construction process, therefore being able to perform preliminary analysis in a more exhaustive way. In [20], BIM is used to perform a parametric analysis aimed at designing the building performance parameters based on climate conditions towards energy efficiency. They studied various design standards (exterior walls material, roofs material and a set of window-to-wall ratios) in combination with building location. The researcher found through results around 15% improvement in the energy consumption due to change design options such as window-to-wall ratio, disregarding the location. Piselli et al. [21] analyzed the impact of energy plant replacement for the retrofit of existing buildings. Ground source heat pump system and existing gas boiler were compared and results showed up to 73% heating energy need reduction and 69% CO_2 emission savings while maintaining the same operation and comfort conditions for the occupants. Taha et al. [22] performed daylight and photovoltaic panel performance analyses on a modelled two-floor educational facility. They quantified also the energy saving led by design alternatives in terms of kWh/year, experimenting the use of BIM technology for preliminary design assessments. In [23], De Gaetani et al. investigated the interdependency of main building components in terms of both costs and energy consumption. They found that depending on the expected life cycle of the building, the priority of investments on more performing technologies should be accurately analyzed, being that not obvious. Mohelníková et al. [24] analyzed several options for complex retrofits focused on building envelopes and their window/shading systems together with the installation of efficient technical systems for HVAC systems. Their analyses on heating energy demands in existing

old buildings showed the importance of renovations for their energy efficiency. On the other hands, the thermal and daylight evaluation results showed that renovation improvements could be sometimes counter-productive from the indoor comfort point of view.

The aforementioned researches are based on the generation of several BIM models by differently combining factors influencing the energy analysis, carried out subsequently the BIM modeling process. The objective was not finding the optimum in absolute terms but identifying factors more affecting the energy balance or comparing different design options. The bottle-neck of such comparisons were the limited amount of investigable alternatives, due to operational and practical reasons. Among all the possible solutions there are the automation of the whole process of modeling and analysis of the building or the estimate of intermediate options so to densify the set of design alternatives. In this context, approaches using BIM include work by [25] who created a generative design system on top of Autodesk Revit [26] that manipulates window sizes and invokes the Autodesk Green Building Studio API [27] to determine the resulting energy-analysis metric. In a subsequent work [28], the authors used Autodesk Dynamo [29], a visual-programming tool, to solve a similar problem, namely that of discrete window-size optimization for reducing daylight usage and energy consumption. In [30], a framework for optimization BPOpt (BIM Performance Optimization) based on Dynamo was created, which breaks the generative design into fives phases: decision variables (input), initial random population (initial setup), fitness functions (evaluation), generation loop (decision making), and writing to CSV File (output). Depending on the problem the user plans to optimize, they could alter the input parameters and develop the appropriate fitness functions, thereby making the problem seemingly independent of the type of generative design.

However, recent advances in Information Technology have led to a new breed of computer-based tools. Artificial Neural Networks (ANNs) are one of the best and most widely utilized tools in the development of prediction models in different fields and BIM exploitation for energy analysis is one of these. Alshibani and Alshamrani [31] described the development, testing, and validation of a conceptual system to assist architects in selecting the optimum alternative design that minimizes the cost of energy consumption of residential buildings in Saudi Arabia. The proposed system incorporated BIM and ANN-based models to predict energy cost. Different ANN models with different characteristics were tested and built using real data on energy consumption collected from six cities across the eastern province of the country. In [32], Ma et al. evaluated indoor personal thermal comfort for a comfortable and green thermal environment. They proposed a BIM-ANN based system for this purpose. The system included an ANN predictive model considering three environment parameters (air temperature, air humidity, and wind speed around the person), three human state parameters (human metabolism rate, clothing thermal resistance, and the body position) and four body parameters (gender, age, height, and weight) as inputs. In [33], the research focused on developing robust ANNs for use as surrogate models for simulation by using data generated from the Simulation-Based Multi-Objective Optimization (SBMO) model developed in a previous research [34]

The outcome of this study showed that the proposed ANN models could efficiently predict the total energy consumption, life-cycle cost and life-cycle assessment for the whole building renovation scenarios considering the building envelope, HVAC, and lighting systems.

In this work, a framework for automatic creation of large datasets where searching the optimal combination of a set of given parameters of a BIM model is proposed, facing also the problem of how to speed up the whole procedure. Such approach has been implemented in two main steps. Firstly exploiting the possibility of automating the model generation and analysis given by interfacing the involved software packages with scripts and codes that replicate the human intervention avoiding gross errors and allows for managing the complete procedure from an initial setup. Secondly, Artificial Neural Networks and Transfer Learning technique has been applied, aiming at speeding up the

dataset creation and increasing the parameters range resolution on the basis of a subset of available design alternatives.

2. Methodology

In this work, the automatic creation of a relevant set of design options to be analyzed for searching the optimum was carried out in two main steps. In the first step, the usual workflow that would be applied manually was followed by running scripts and codes that perform each stage (i.e., parameters setup, model creation, energy analysis, results evaluation) sequentially and depending just on the initial setup given by the user. This allowed a quick creation of design alternatives by greatly reducing the manual intervention and consequently the possibility of gross errors and loss of time. At the end of this first step a great amount of data could already be obtained but the process was time-consuming and intermediate solutions should be obtained by repeating the scripts with different setups. This aspect could be faced by exploiting the results of this first step as input for predicting (instead of computing) the desired intermediate solutions. In the second step of the proposed approach, ANNs were applied to increase the resolution of parameters range and Transfer Learning technique was applied aiming at allowing the possibility of evaluating options that were not considered in the initial setup. The following subsections focus on the description of these two main steps that are the framework of the proposed approach.

2.1. Automatic Design Options Creation

With the aim to build a framework that automates design options creation for energy analysis, a relevant dataset had to be built as a first step. At the first stage, for energy analysis, the following parameters of a building were accounted for: window-to-wall ratio (WWR), shape of the building, rotation in plan, thermal characteristics of elements such as floor, walls and roof. Each parameter had a certain range of variations, thus for a complete analysis, each combination was considered. The general workflow of this step is presented in Figure 1:

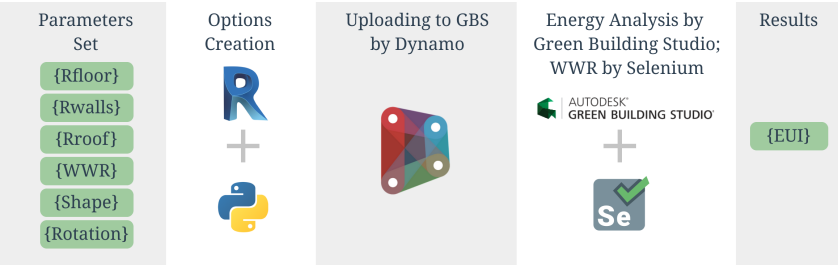

Figure 1. Automation of design options creation workflow.

After having determined the range of parameters of interest at the first stage, each combination of these parameters was applied to an Autodesk Revit model one by one. This was done using Revit API and Python programming language: they together made it possible to interact with the authoring software through code. As combinations were applied, current state of the Revit model is saved in gbXML file format, where each gbXML file described the Revit model with a specific applied set of parameters from an energetic point of view. Therefore, the output of the second stage was a collection of gbXML files, with each file being a design option.

At the third stage, to conduct energy analyses of this collection of data, these gbXML files were uploaded to Autodesk Green Building Studio using Dynamo, which is a part of the out of the box Revit and is a visual programming tool. The Dynamo package used was "Energy Analysis for Dynamo" [35], which requires a list of gbXML files created earlier, and a Green Building Studio project ID, specified in the ProjectID node on Figure 2. The ID can be found using GetProjectsList node from the package, given that a Green Building Studio

project has already been created. Other than that, for Revit to be able to recognize the user, one must have logged in in the Revit environment with valid Autodesk credentials.

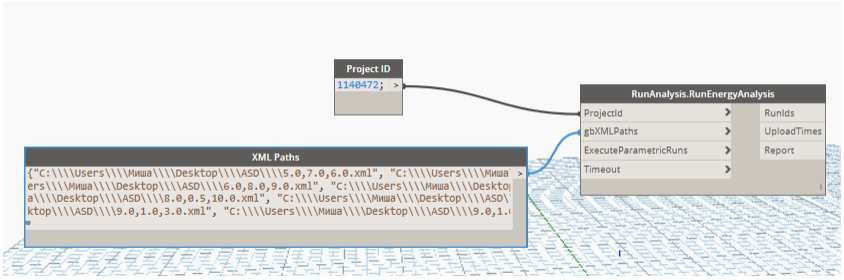

Figure 2. Dynamo for Revit node linking the gbXML files to the Green Building Studio project Id.

After this Dynamo script was run, each file was uploaded to Green Building Studio, and the energy analyses started automatically using DOE-2 engine. This concluded the third stage of the workflow.

Upon completion of the analyses, additional design alternatives based on previous design options were created. The additional design alternatives accounted for different WWR ratios and they were accounted for by creating so-called "alternative runs" using browser automation technique (i.e., Selenium Python package). GBS offers a wide range of parameters to change in alternative runs; however in this case, advantage was taken only of assigning different WWR ratios, which added the last level complexity to the analysis. It is important to note, that windows added by GBS had a default R value of 0.5 $\frac{W}{m^2K}$. Analyses of added alternatives started automatically, and when completed, the results were downloaded from the website using built-in function of GBS that allowed us to download reports in Excel format. The results represented a set of Energy Use Intensity (EUI) values for each design alternative. EUI estimated how good or bad a model performed from energy consumption standpoint and was measured in $\frac{kWh}{m^2year}$. The obtained EUI values were the output for any given configuration considered and could be used as input to train an ANN that could then predict additional EUI values for not considered configurations.

2.2. Design Options Prediction through ANN

The body of an ANN can be thought of as a series of matrices with gradually decreasing size. To get a prediction of EUI, the input vector was multiplied by the first matrix, giving a vector that was in turn multiplied by the next matrix. This series of operations eventually led to a single number, which was the EUI prediction. Conceptually, just described ANN could be represented as in Figure 3:

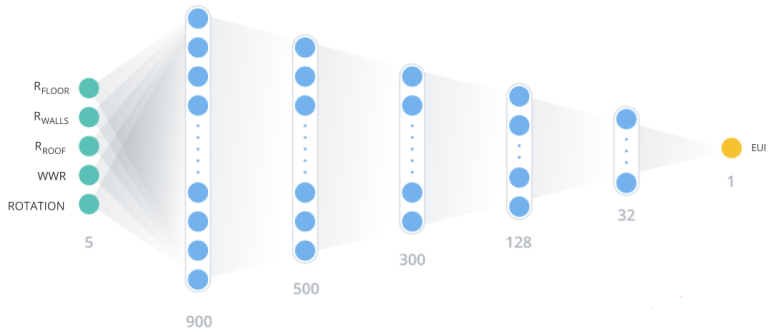

Figure 3. Conceptualization of the implemented ANN architecture for EUI prediction on the basis of the five considered input layers.

Initially randomly generated values of the matrices got updated based on partial derivative of so-called loss function used to assess predictions after each iteration described above. This ensured that the prediction came closer to the ground truth value after each iteration. This process was called training of a neural network, where each neuron performed a weighted sum of the x values representing the input vector. By tuning the weights of the neuron w, each of of them was capable of fitting well only data with linear correlation, but if they were stuck atop each other, they formed a powerful system capable of fitting quite complex data, which was indeed this case.

Usually, out of a dataset, a big amount of available data was used for training, and small fractions—for validating, i.e., some data points were left out of training process intentionally to check how good or bad the predictions based on part of the data were. However, if an ANN trained for a similar task was available, then so-called transfer learning could be applied. Transfer learning is a technique allowing to use already trained neural network and retrain it for a new dataset. This is feasible because an already trained ANN 'knows' how each input influences the outcome, therefore it knows general patterns. Therefore, if a new dataset represents about the same distribution as the one for which the existing ANN is trained, then with minor shifts of weights it is possible to make the ANN fit the new data. The main benefits of using this technique is that it takes less time and data to train an ANN built on other ANNs. In fact, being able to apply transfer learning has been the key consideration behind this particular choice of ML technique. Based on this reasoning, workflow depicted on Figure 4 emerges:

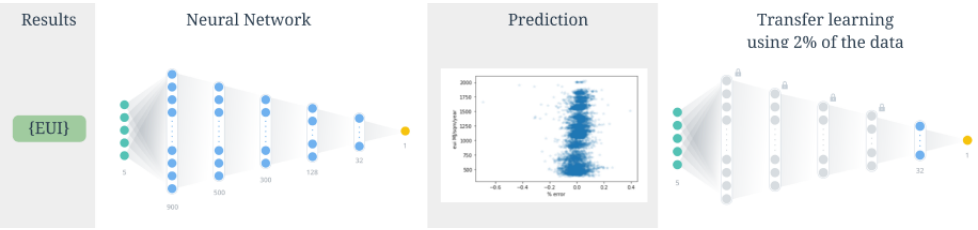

Figure 4. Workflow for properly exploiting Transfer Learning technique. A smaller amount of input data is used for predicting values on the basis of previously trained similar Neural Networks.

According to the workflow, first, an ANN was trained on a set of data, and the ANN was reused to be a foundation of other ANN to take advantage of Transfer Learning and estimate time savings and predictions accuracy.

3. Case Study

This section is devoted to a practical implementation of the workflow previously described. The considered parameters around which design optioneering revolved were those reflected in Table 1.

Table 1. Design parameters.

Parameter	Range of Values
R_{Floor}	0.5–1–2–3–4–5–6–7–8–9–10–12.5 $\frac{m^2K}{W}$
R_{Walls}	0.5–1–2–3–4–5–6–7–8–9–10–12.5 $\frac{m^2K}{W}$
R_{Roof}	0.5–1–2–3–4–5–6–7–8–9–10–12.5 $\frac{m^2K}{W}$
Rotation	0°–45°
Shape	Box–Prism (same Box wall area)–Prism (same Box volume)
WWR	0%–50%–95%

Regarding the shape parameter, three different geometries of the Revit model were analyzed: a box shape and two hexagonal prisms reflecting the wall surface and the volume

of the box reference shape. We called them Box A, Prism B and Prism C, respectively. The reason for having two hexagonal models was the following: when evaluating an influence of a parameter on a certain outcome, it is a good practice to change the parameter not influencing other ones, but in case of different shape it is not possible (EUI refers to unitary floor surfaces), so all variations conjugated with a shape change are accounted for.

As seen in Table 2, all options had the same floor area to exclude influence of this factor.

Table 2. Characteristics of the three considered model geometries.

Model Type	Floor Area, m^2	Volume, m^3	External Wall Surface, m^2
Box A	94	310	144
Prism B	94	337	144
Prism C	94	310	155

From Table 1 it is seen that number of combinations was 12·12·12·3·2·3 = 31104. This large number of combinations required automation of their creation, which was achieved using Revit Python Shell for Revit.

All Revit models were created using a build-in Revit HVAC system definition; namely, Residential 14 SEER. It was kept constant throughout the study.

3.1. Creation of the Initial Dataset

As first step, thermal resistances of the materials of which the elements were made of were subject to iterative change. Thermal resistance of an element is $R = \frac{C}{h}$, where C is the thermal conductivity of the material measured in $\frac{W}{mK}$ and h is the element thickness. Therefore, knowing thickness of an element, it was possible to set desirable thermal resistance, changing the thermal conductivity value, which was accessible in Revit through the Material Browser and could be found as attribute of the material the element was made of. Revit API was used to iteratively change the value of thermal conductivity of each individual material, going through each combination of thermal resistances. The value of thermal resistance that had to be set on each iteration was calculated, h being known, as well as the desired R. To find each relevant material's thermal asset, which is a Revit object holding all thermal characteristics of materials, the script in Appendix A as Listing 1 was run in Revit Python Shell with a Revit model opened.

Upon completion, those objects could be used to assign thermal resistances and subsequently convert each Revit model state to a gbXML representation using Revit API. The code in Appendix A as Listing 2 did that.

After each cycle of the loop, a rotation of 45° was applied to the current model to take into account of orientation of the model.

At this point, 10368 gbXML models had been created. The last factor, WWR, was accounted for on Green Building Studio website after all gbXML files were loaded onto the GBS using Dynamo node (see Figure 2) where all paths came from parsing the directory containing just obtained gbXML files, as per Python code in Appendix A as Listing 3.

Launch of this node marked the start of uploading process, after which energy analyses of the design options started automatically using DOE-2 engine implemented in GBS. Green Building Studio was then used to assign different WWRs for walls in North, South, West and East directions. This was usually done by manually creating an alternative run with the desired WWR but considering the number of alternative runs to be done, decision was made to resort to browser automation technique. Thus, after having collected URLs of all main ran into a single txt file, Listing 4 in Appendix A was implemented to assign all WWRs.

After this, 31104 design alternatives were created and analyzed. In fact, at the end of this stage GBS dashboard contained six projects with 5184 runs each. A summary of how the dashboard looked like is presented in Table 3:

Table 3. Summary of the GBS dashboard at the end of the automatic analyses.

Project Id	Shape / Orientation	No. of Runs
1	Box A/0°	5184
2	Prism B/0°	5184
3	Prism C/0°	5184
4	Box A/45°	5184
5	Prism B/45°	5184
6	Prism C/45°	5184

The results of the analyses can be downloaded for further analyses aiming at finding the optimal configuration or assessing which, among the considered parameters, influenced the final result more. This would be out of the scope of the presented work but with such amount of design alternatives to be compared, a preliminary assessment can be done by plotting them. This can be done by fixing three of the six considered parameters under investigation so to be able to plot the EUI values obtained as function of the remaining three parameters free to vary in their assigned range. An example is provided in Figure 5.

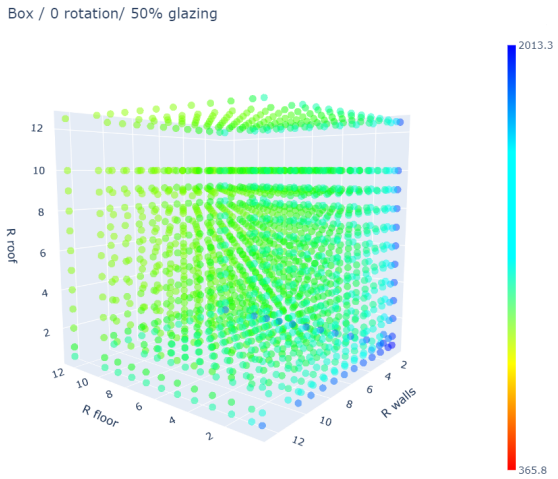

Figure 5. Example of visual inspection of automatic GBS results. The provided example shows the impact of R_{Floor}, R_{Roof} and R_{Walls} in the case of Box A model shape with 0° of rotation and 50% of WWR.

The cube in Figure 5 is identified by 1728 points where each point represents a single combination of thermal resistance of walls, roof, and floor of a Revit model. In particular, it refers to the Box A model results with 50% WWR and 0° rotation. Points are coloured on the basis of the correspondent obtained EUI and the EUI scale beside the cube is common for all the cubes obtainable with Box A setup. As a preliminary and qualitative assessment, such a plot revealed that among all the alternatives of the Box A model, energy demand had a range of about 365-2010 EUI. This meant that the worst configuration demands about 5.5 more energy if compared with the best one. In the provided example, no red points were present (lowest EUI of the scalebar), immediately leading to state that the best configuration could not be found with 50% WWR and 0° rotation for Box A model. Further, although it was obvious that opposite results were obtained along the edges of the cubes with lowest/highest values of R_{Floor}, R_{Roof} and R_{Walls}, by navigating within the cube the designer could easily check the impact of tuning the thickness (therefore the R value) of one or two components. Although the presented qualitative analysis was out of the scope

of the presented research, it described one of the possible practical applications exploiting a large number of available processed design alternatives.

3.2. Neural Networks Implementation for Dataset Resolution Increase

Conducting such a set of experiments was a very time-consuming procedure. Moreover, the data were quite complex for these five parameters already. If we should refine the above procedure, the number of factors will grow, making the data even more complex. However, with neural networks it was possible to achieve the same results spending less time and computational power and to get a scalable framework for design optioneering.

Main principles of ANNs were explained in the previous section. The implemented ANN had the architecture shown in Figure 3, and it was built using Keras framework [36]. The green circles represent the input factors expressed in numbers, and the yellow circle is the EUI value in output. Each blue circle is a single neuron. The ANN had five hidden layers with 900, 500, 300, 128 and 32 neurons respectively.

Such an architecture was a product of a trial and error process. From one side, an ANN should be complex enough to capture accurately all trends present in the data, so that it does not underfit. From the other side, an ANN should not be very complex to prevent overfitting. Therefore, an ANN model should be balanced between the two. This could be achieved in several ways, for example: manual brute force selection, uniform or random grid search available in Scikit learn Python package. Loss function, described further, served as an indication of the ANN overfitting or underfitting the data.

The loss function used to compare N predictions $\hat{y}$ with the ground-truth y was Root Mean Squared Error ($RMSE$): it was sensitive to outliers in the data, which were not present in the dataset, since it was prepared synthetically.

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

Along with RMSE, Mean Absolute Error (MAE) was used as metrics, only to see the discrepancy between RMSE and MAE as it was a good indicator of different magnitude among the N predictions.

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{2}$$

Additionally, the constant term of the last neuron was set to mean EUI value of the available training set to accelerate learning during first training loops [37], and all other weights were initialized as per Xavier [38]. The Adam optimizer [39] was used to adjust gradient descent paths, so that convergence happened smoother and faster. The learning rate of the optimizer, being the rate at which weights were updated with respect to gradients, followed a linear descent as training progressed:

$$LearningRate = 3e-5 * \frac{2500 - epoch}{2550} \tag{3}$$

It was important to lower the learning rate as the predictions came closer to the convergence point in order not to overshoot the target. Finally, all except the last neurons' outputs were fed through Selu activation function. The activation function was applied to a neuron output, so that the "activated" output was an input to the next layer. Activation added non-linearity in the process and makes the learning possible. The last neuron had a linear activation function. The Selu function was as follows [40]:

$$SELU(x) = \lambda \begin{cases} x & \text{if x} > 0 \\ \alpha e^x - \alpha & \text{if x} \leq 0 \end{cases} \tag{4}$$

where:

$$\lambda \approx 1.67, \alpha \approx 1.05$$

Following the methodology and architecture described above, the NN was trained using data of the Box A Revit model: 93.75% for training, and 6.25% for validation. After training was completed, the ANN knew main patterns in the data, and how inputs influenced the output. With this just trained ANN, a use could be made of the rest of the data obtained from Prism B and Prism C models to put in evidence low amount of effort needed to predict their EUI. For this, Transfer Learning (TL) technique was applied: in just trained NN, four largest layers were frozen (their weights were set as non-trainable), and the rest of the weights were retrained using only 6.25% of available data for training, as shown in Figure 6.

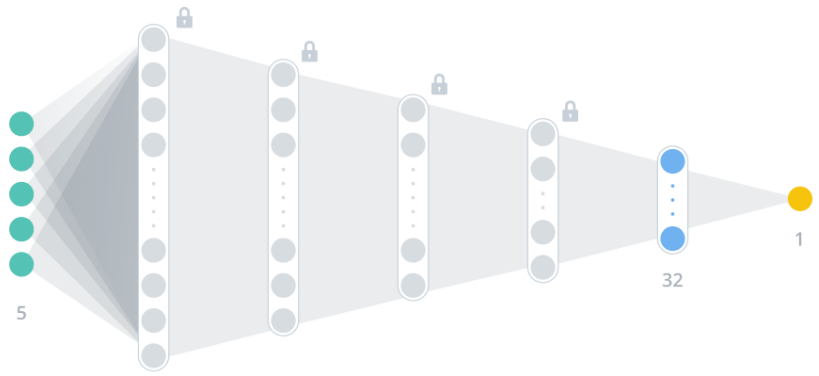

Figure 6. Architecture of the ANN when Transfer Learning is applied.

The main benefit of this approach was its gain of speed and lower about of data needed: only a small fraction of the weights was updated, and thus, less data were needed when using pre-trained network. Lower level features determined general patterns in the data, and they could remain frozen given that the data on which it was retrained came from about the same distribution.

4. Results and Discussion

Predictions after training the first ANN with the Box A data are displayed in Figure 7 while their accuracy is reported in Table 4.

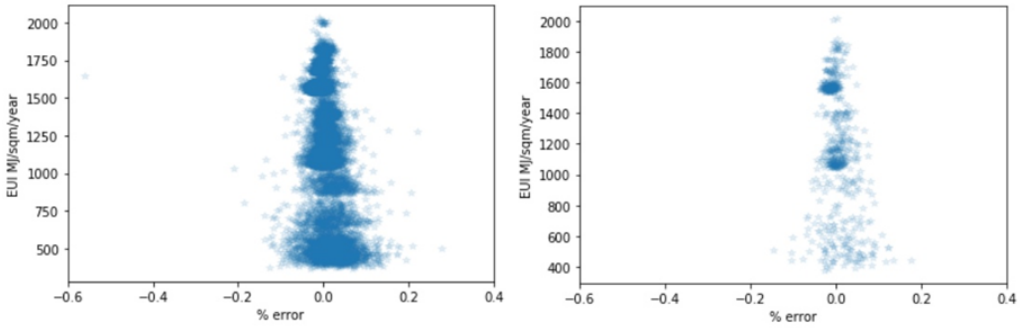

Figure 7. Predicted EUI values for training set on the left side and validation set on the right side. Each point represents an EUI value for a certain combination of the input parameters.

By making use of the 10368 data points obtained from the Box Revit model, 93.75% went for training, 6.25% for validation. The ANN was able to predict the real value with RMSE = 0.5479 EUI for training set and RMSE = 0.7202 EUI for validation (data the ANN had not seen). RMSE and MAE were of the same order of magnitude. This in turn put in

evidence that non-linear patterns of the data could be captured along the whole range of parameters considered.

Table 4. Box A data predictions accuracy.

	Training Set	Validation Set
RMSE [EUI]	0.5479	0.7202
MAE [EUI]	0.2711	0.3110

As per results of the transfer learning process, exploiting the ANN trained on the Box A model dataset and used for Prism B and Prism C models, obtained predictions are displayed in Figure 8 and summarized in Table 5.

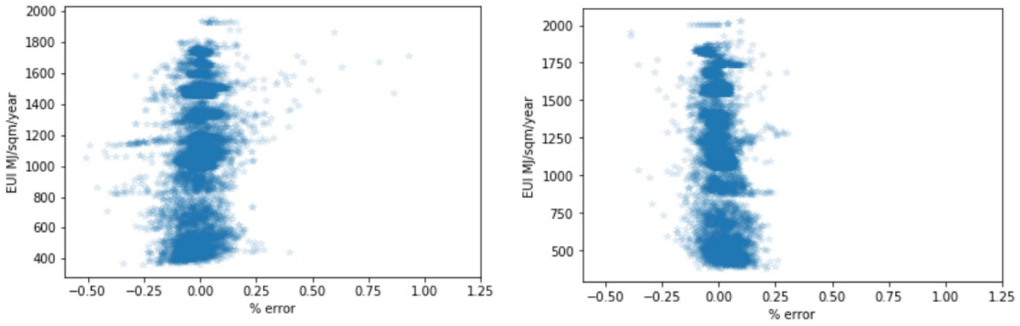

Figure 8. Predicted EUI values for Prism B model on the left side and Prism C model on the right side after applying transfer learning to the previously trained ANN based on Box A data. Each point represents an EUI value for a certain combination of the input parameters.

Table 5. Prism B and Prism C data predictions accuracy.

	Training Set	Validation Set
Prism B model		
RMSE [EUI]	0.7135	0.9493
MAE [EUI]	0.4065	0.5292
Prism C model		
RMSE [EUI]	0.5767	0.6897
MAE [EUI]	0.2983	0.3587

In order to apply transfer learning (i.e., retrain to match the new data) to the Prism B and Prism C datasets, the first four layers were then frozen (set as non-trainable). For this step only 6.25% of the available data were used for training. Additionally, in these cases, RMSE and MAE remained with the same order of magnitude. Prism C predictions got very similar results to the original Box A model ones for both the training and validation sets. Sligthly worse results were obtained for the Prism B model but considering the fact that the training took approximately 1–2 min compared to about 3 h for the first neural network, with the amount of training applied to those two models being exactly the same, the improvement in applying transfer learning technique was indisputable.

This point was further investigated testing the amount of Box A model data needed for applying transfer learning technique to Prism C model without loosing accuracy. For this purpose, a different amount of the available data was used for training: 5.5%, 4%, 2%, 1%, 0.5% and 0.25%. As one can see in Figure 9, the performance started degrading at about 1–2% of the available training dataset. This amount of points (approximately 150) could be

obtained several times faster than the original 10,368 data points, thus taking about 10–20 min for the whole procedure if automated.

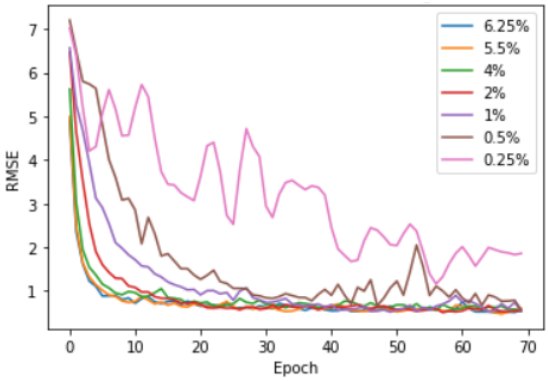

Figure 9. Ablation study results aiming at investigating the impact of reducing the size of the subset of data to be used for applying transfer learning technique without significant loss of accuracy.

It was clearly seen that 1–2% training/validation split was the breaking point, lower than which it was hard to achieve good performance, i.e., loss function did not reach the plateau reached with a higher amount of data. At the same time, there was no need to feed more data points as it did not improve the performance. However, prediction quality of the Neural Network for different Revit models was somewhat different (Tables 4 and 5), albeit fluctuations neutralized this effect to some extent. This implied that to train an ANN for different models would take different amount of training, but most importantly, that for different models the best approximation performance would differ. Summarizing, the neural network that was obtained was able with lower effort and time to output a large set of precise predictions of energy performance of a building in different configurations. For this, few hundreds of data points would be required (that can be obtained using the automation technique described in the previous sections) and just 1–2 min for training the original neural network.

5. Conclusions

In this work, the creation of the dataset where searching the optimal combination of a set of given parameters of a BIM model has been analyzed into two main steps.

First, a framework for automatic design options creation and analysys has been proposed, implemented and tested on a case study. Such framework exploits the possibility of replicating the usual manual workflow by means of scripts and codes. Revit API, Python programming language and the Dynamo package interacted with the BIM authoring software through code, that created and upload gbXML files to Green Building Studio for completing the design alternative setup and performing energy analyses. The test case considered several parameters (thermal characteristics of floors, walls, roofs, building orientation, windows-to-wall ratios), each of them with different range values, constituting 10368 design alternatives for three different building shapes (31104 options in total). Although the procedure is very time and resource consuming, the main advancement relies in the reduction of the manual intervention and the possibility of creating a very large dataset of design options, simply defining the initial desired set of parameters and avoiding gross errors. Such amount of data would have been quite difficult to be obtained manually.

The second part of the investigation aimed at speeding up the dataset creation and increasing the resolution of the considered parameters range. For this purpose, neural network and transfer learning technique were applied, tested and compared with the results obtained with the previous approach. First, a single building shape was considered and 93.75% of the 10368 previously analyzed options was used to train a neural network and predict the remaining 6.25%. Results showed a very good agreement in the validation dataset, leading to the conclusion that with such approach, the resolution of parameters

range can be increased without repeating the automatic computation process. More interesting have been the results of the application of transfer learning technique to the other two building shapes. By exploiting the neural network trained on the previous building shape, the other two datasets have been predicted by using only 6.25% of the available data, without significant loss of accuracy. This means that the complete initial dataset of 10368 options for a given shape can be reduced down to only 648. Ablation studies also showed that this percentage is redundant, and could be further reduced down to about 1% or 100–150 design options.

It is important to note, that these 100–150 design options come from a model with slightly different settings (i.e., different shape of the envelope), which makes it possible to take another model located, for instance, in a different climate zone and conduct the same type of analyses, which may require different amount of data to predict the EUI with high accuracy; however for that more testing of the framework is required.

The use of transfer learning achieved its target of speeding up the process: the time taken was 1 day to create the initial dataset for a model (10368 design options) and 3 hours for training the ANN; however the transfer learning part would take around 10–15 minutes to create only a fraction of that data and 2–3 minutes to re-train the ANN, which is a significant speed-up.

The framework presented serves as a proof of concept, and obviously does not account for all possible factors that influence the final EUI value. Additional input factors would be handled in one of the two ways. In case of moderate number of new factors, a new neuron should be added to the input layer for each factor, and the body of the ANN should be adjusted accordingly. However, in case of extreme number of new inputs, they would be grouped semantically to form a number of separate ANNs, output of which would be an input to the main ANN. In other words, the system would become an assembly of ANNs, where separation of concerns of each aspect is respected, and they only are mixed at the later stage in the main ANN.

Author Contributions: Conceptualization, M.D. and C.I.D.G.; methodology, M.D. and C.I.D.G.; software, M.D.; validation, M.D. and C.I.D.G.; formal analysis, M.D.; investigation, M.D.; resources, M.D.; data curation, M.D.; writing—original draft preparation, M.D. and C.I.D.G.; writing—review and editing, C.I.D.G.; visualization, M.D.; supervision, C.I.D.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Listing 1: Material Assets' Properties Detection

```
### Import ###
import clr
from Autodesk.Revit.DB import *
clr.AddReference('ProtoGeometry')
from Autodesk.DesignScript.Geometry import *
clr.AddReference("RevitServices")
import RevitServices
from RevitServices.Persistence import DocumentManager
from RevitServices.Transactions import TransactionManager
import Autodesk.Revit.Creation
clr.AddReference("RevitAPI")

### List of necessary Thermal Conductivity parameters ###
FloorR=[1, 0.5, 0.25, 0.166667, 0.125, 0.1, 0.083333,
```

```
            0.071429, 0.0625, 0.0555556, 0.05, 0.04]
WallsR=[0.6, 0.3, 0.15, 0.1, 0.075, 0.06, 0.05, 0.042857, 0.0375, 0.03333,
                                                0.03, 0.024]
RoofR=[1, 0.5, 0.25, 0.166667, 0.125, 0.1, 0.083333,
            0.071429, 0.0625, 0.0555556, 0.05, 0.04]
# List of materials in the project.
#'doc' refers to the current Revit document
mat=FilteredElementCollector(doc).OfCategory(BuiltInCategory.OST_Materials)
materials=[]
for m in mat:
        materials.append(m.Name)

### Assign required material property to a variable ###
for m in mat:
        if m.Name=='Rigid insulation':
                Wall=doc.GetElement(m.ThermalAssetId).
                                        LookupParameter("Thermal Conductivity")
        if m.Name=='Wood Sheathing, Chipboard':
                Roof=doc.GetElement(m.ThermalAssetId).
                                        LookupParameter("Thermal Conductivity")
        if m.Name=='Structure, Timber Joist/Rafter Layer':
                Floor=doc.GetElement(m.ThermalAssetId).
                                        LookupParameter("Thermal Conductivity")
```

Listing 2: Material Assets

```
### Setting out Energy Analysis parameters ###
opt=Analysis.EnergyAnalysisDetailModelOptions()
opt.EnergyModelType=Analysis.EnergyModelType.BuildingElement
opt.ExportMullions=False
opt.IncludeShadingSurfaces=False
opt.SimplifyCurtainSystems=True
opt.Tier=Analysis.EnergyAnalysisDetailModelTier.SecondLevelBoundaries

### Loop over all R combinations and create models ###
for i in FloorR:
        t=Transaction(doc,"R change")
        t.Start()
        Floor.Set(i/0.3048)
        t.Commit()
        t.Dispose()
        for j in WallsR:
                t=Transaction(doc,"R change")
                t.Start()
                Wall.Set(j/0.3048)
                t.Commit()
                t.Dispose()
                for k in RoofR:
                        t=Transaction(doc,"R change")
                        t.Start()
                        Roof.Set(k/0.3048)
                        t.Commit()
                        t.Dispose()
                        c=Transaction(doc,"Model Creation")
                        c.Start()
                        model=Analysis.EnergyAnalysisDetailModel.
                                                Create(doc, opt)
                        model.TransformModel()
                        GBopt=GBXMLExportOptions()
                        GBopt.ExportEnergyModelType=
                            ExportEnergyModelType.BuildingElement
                        doc.Export("C:\Users\User\Desktop\ASD",
                               str(round(0.5/i,1))+","+
                               str(round(0.3/j,1))+","+
                               str(round(0.5/k,1)),GBopt)
                        c.Commit()
                        c.Dispose()
```

Listing 3: Directory Parsing

```
import os, sys
path = "D:\\New folder (2)\\"
dirs = os.listdir(path)
paths=[]
for i in dirs:
    paths.append(path+i)
paths[0] #'D:\\New folder (2)\\0.5,0.5,0.5.xml'
```

Listing 4: WWRs Assignment

```
from selenium import webdriver
from selenium.webdriver.support.ui import Select
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
direct=str(input('Enter file name'+'txt'))
file=open(r'C:\\Users\\User\\Directories\\'+'direct','r')
lis=file.readlines()
for i in range(len(lis)):
    lis[i]=lis[i].strip()

driver=webdriver.Chrome('D:\\Python\\chromedriver')
driver.get('https://gbs.autodesk.com/GBS/')

for i in range(len(lis)):

    driver.get(str(lis[i]))
    try:
        WebDriverWait(driver, 13).until(EC.text_to_be_present_in_element

            ((By.CLASS_NAME,"gbs"),"No change"))
        WebDriverWait(driver, 10).until(EC.
            frame_to_be_available_and_switch_to_it
            ((By.ID,"MainContent_PinTab1")))
        WebDriverWait(driver,10).until(EC.
            frame_to_be_available_and_switch_to_it
            ((By.ID,"__tab_area3")))
           except:
        print('err'+str(lis[i])+str(50))
        continue
    time.sleep(3)

    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem103_title"]').click()
    driver.switch_to.frame('__tab_area3')
    try:
        obj=Select(driver.find_element_by_name
                                ('DataGrid1$ctl01$drpWindowtoWallRatio'))
        obj.select_by_visible_text('50%')
    except:
        print(str(lis[i])+str(50))
        continue
    driver.switch_to.default_content()

    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem104_title"]').click()
    driver.switch_to.frame('__tab_area4')
    try:
        obj=Select(driver.find_element_by_name
                                ('DataGrid1$ctl01$drpWindowtoWallRatio'))

        obj.select_by_visible_text('50%')
    except:
```

```
        print(str(lis[i])+str(50))
        continue
    driver.switch_to.default_content()
    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem105_title"]').click()
    driver.switch_to.frame('__tab_area5')
    try:
        obj=Select(driver.find_element_by_name
                                ('DataGrid1$ctl01$drpWindowtoWallRatio'))

        obj.select_by_visible_text('50%')
    except:
        print(str(lis[i])+str(50))
        continue
    driver.switch_to.default_content()

    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem106_title"]').click()
    driver.switch_to.frame('__tab_area6')
    try:
        obj=Select(driver.find_element_by_name
                                ('DataGrid1$ctl01$drpWindowtoWallRatio'))

        obj.select_by_visible_text('50%')
    except:
        print(str(lis[i])+str(50))
        continue
    driver.switch_to.default_content()

    driver.find_element_by_id('MainContent_btnAdd').click()

    try:
        WebDriverWait(driver, 13).until
                    (EC.text_to_be_present_in_element
                    ((By.CLASS_NAME,"gbs"),"No change"))
        WebDriverWait(driver, 10).until
                    (EC.frame_to_be_available_and_switch_to_it
                    ((By.ID,"MainContent_PinTab1")))
        WebDriverWait(driver, 10).
                    until(EC.frame_to_be_available_and_switch_to_it
                    ((By.ID,"__tab_area3")))

    except:
        print('err'+str(lis[i])+str(95))
        continue
    time.sleep(3)
    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem103_title"]').click()
    driver.switch_to.frame('__tab_area3')
    try:
        obj=Select(driver.find_element_by_name
                        ('DataGrid1$ctl01$drpWindowtoWallRatio'))

        obj.select_by_visible_text('95%')
    except:
        print(str(lis[i])+str(95))
        continue
    driver.switch_to.default_content()

    driver.switch_to.frame('MainContent_PinTab1')
    driver.find_element_by_xpath('//*[@id="tabitem104_title"]').click()
    driver.switch_to.frame('__tab_area4')
    try:
        obj=Select(driver.find_element_by_name
                        ('DataGrid1$ctl01$drpWindowtoWallRatio'))
        obj.select_by_visible_text('95%')
    except:
        print(str(lis[i])+str(95))
        continue
```

```
        driver.switch_to.default_content()

        driver.switch_to.frame('MainContent_PinTab1')
        driver.find_element_by_xpath('//*[@id="tabitem105_title"]').click()
        driver.switch_to.frame('__tab_area5')
        try:
            obj=Select(driver.find_element_by_name
                            ('DataGrid1$ctl01$drpWindowtoWallRatio'))
            obj.select_by_visible_text('95%')
        except:
            print(str(lis[i])+str(95))
            continue
        driver.switch_to.default_content()

        driver.switch_to.frame('MainContent_PinTab1')
        driver.find_element_by_xpath('//*[@id="tabitem106_title"]').click()
        driver.switch_to.frame('__tab_area6')
        try:
            obj=Select(driver.find_element_by_name
                            ('DataGrid1$ctl01$drpWindowtoWallRatio'))

            obj.select_by_visible_text('95%')
        except:
            print(str(lis[i])+str(95))
            continue
        driver.switch_to.default_content()

        driver.find_element_by_id('MainContent_btnAdd').click()
        time.sleep(2)

        driver.find_element_by_id('MainContent_btnSubmit').click()
        time.sleep(2)

file.close()
```

References

1. Abbasnejad, B.; Nepal, M.P.; Ahankoob, A.; Nasirian, A.; Drogemuller, R. Building Information Modelling (BIM) adoption and implementation enablers in AEC firms: A systematic literature review. *Archit. Eng. Des. Manag.* **2020**, 1–23. [CrossRef].
2. Nguyen, P.; Akhavian, R. Synergistic effect of integrated project delivery, lean construction, and building information modeling on project performance measures: A quantitative and qualitative analysis. *Adv. Civ. Eng.* **2019**, *2019*. [CrossRef].
3. Olawumi, T.O.; Chan, D.W. Identifying and prioritizing the benefits of integrating BIM and sustainability practices in construction projects: A Delphi survey of international experts. *Sustain. Cities Soc.* **2018**, *40*, 16–27. [CrossRef]
4. Khodeir, L.M.; Othman, R. Examining the interaction between lean and sustainability principles in the management process of AEC industry. *Ain Shams Eng. J.* **2018**, *9*, 1627–1634. [CrossRef]
5. Carvalho, J.; Bragança, L.; Mateus, R. Guidelines for analysing the building energy efficiency using BIM. In *IOP Conference Series: Earth and Environmental Science*; IOP Publishing: Bristol, UK, 2020; Volume 588, p. 022058.
6. Gerrish, T.; Ruikar, K.; Cook, M.; Johnson, M.; Phillip, M. Using BIM capabilities to improve existing building energy modelling practices. *Eng. Constr. Archit. Manag.* **2017**. [CrossRef]
7. Lim, C.W.; Yu, J.H.; Kim, C.D. Analysis for BIM object information compatibility problem classification among BIM softwares. In *Proceedings of the Korean Institute of Building Construction Conference*; The Korean Institute of Building Construction: Seoul, Korea, 2010; pp. 257–260.
8. Kim, J.B.; Jeong, W.; Clayton, M.J.; Haberl, J.S.; Yan, W. Developing a physical BIM library for building thermal energy simulation. *Autom. Constr.* **2015**, *50*, 16–28. [CrossRef]
9. Marzouk, M.; Azab, S.; Metawie, M. BIM-based approach for optimizing life cycle costs of sustainable buildings. *J. Clean. Prod.* **2018**, *188*, 217–226. [CrossRef]
10. Khanzadi, M.; Kaveh, A.; Moghaddam, M.R.; Pourbagheri, S.M. Optimization of building components with sustainability aspects in BIM environment. *Period. Polytech. Civ. Eng.* **2019**, *63*, 93–103. [CrossRef]
11. Shadram, F.; Mukkavaara, J. An integrated BIM-based framework for the optimization of the trade-off between embodied and operational energy. *Energy Build.* **2018**, *158*, 1189–1205. [CrossRef]
12. Bastos Porsani, G.; Del Valle de Lersundi, K.; Sánchez-Ostiz Gutiérrez, A.; Fernández Bandera, C. Interoperability between Building Information Modelling (BIM) and Building Energy Model (BEM). *Appl. Sci.* **2021**, *11*, 2167. [CrossRef]
13. Hijazi, M.; Kensek, K.; Konis, K. Bridging the gap: Supporting data transparency from BIM to BEM. In *Proceedings of the Architectural Research Centers Consortium 2015 Conference*; Perkins and Will: Chicago, IL, USA, 2015; pp. 149–166.

14. Kamel, E.; Memari, A.M. Automated building energy modeling and assessment tool (ABEMAT). *Energy* **2018**, *147*, 15–24. [CrossRef]
15. Mastino, C.C.; Baccoli, R.; Frattolillo, A.; Marini, M.; Di Bella, A.; Da Pos, V. *The Building Information Model and the IFC Standard: Analysis the Characteristics for the Acoustic and Energy Simulation of Buildings*; Free University of Bolzen: Bolzano, Italy, 2017; p. 479.
16. Ramaji, I.J.; Messner, J.I.; Mostavi, E. IFC-based BIM-to-BEM model transformation. *J. Comput. Civ. Eng.* **2020**, *34*, 04020005. [CrossRef]
17. Dimitriou, V.; Firth, S.K.; Hassan, T.M.; Fouchal, F. BIM enabled building energy modelling: Development and verification of a GBXML to IDF conversion method. In *Proceedings of the 3rd IBPSA-England Conference BSO*; IBPSA: Newcastle, UK, 2016; p. 1126.
18. Malhotra, A.; Frisch, J.; van Treeck, C. *Technical Report: Literature Review Concerning IFC, GbXML and CityGML Data Models for Energy Performance Simulation*; Universitätsbibliothek der RWTH Aachen: Aachen, Germany, 2019.
19. Kamel, E.; Memari, A.M. Review of BIM's application in energy simulation: Tools, issues, and solutions. *Autom. Constr.* **2019**, *97*, 164–180. [CrossRef]
20. Najjar, M.K.; Tam, V.W.; Di Gregorio, L.T.; Evangelista, A.C.J.; Hammad, A.W.; Haddad, A. Integrating parametric analysis with building information modeling to improve energy performance of construction projects. *Energies* **2019**, *12*, 1515. [CrossRef]
21. Piselli, C.; Romanelli, J.; Di Grazia, M.; Gavagni, A.; Moretti, E.; Nicolini, A.; Cotana, F.; Strangis, F.; Witte, H.J.; Pisello, A.L. An integrated HBIM simulation approach for energy retrofit of historical buildings implemented in a case study of a medieval fortress in Italy. *Energies* **2020**, *13*, 2601. [CrossRef]
22. Taha, F.F.; Hatem, W.A.; Jasim, N.A. Effectivity of BIM technology in using green energy strategies for construction projects. *Asian J. Civ. Eng.* **2020**, *21*, 995–1003. [CrossRef]
23. De Gaetani, C.I.; Macchi, A.; Perri, P. Joint Analysis of Cost and Energy Savings for Preliminary Design Alternative Assessment. *Sustainability* **2020**, *12*, 7507. [CrossRef]
24. Mohelníková, J.; Novotnỳ, M.; Mocová, P. Evaluation of School Building Energy Performance and Classroom Indoor Environment. *Energies* **2020**, *13*, 2489. [CrossRef]
25. Rahmani Asl, M.; Zarrinmehr, S.; Yan, W. Towards BIM-based parametric building energy performance optimization. In *Proceedings of the 33rd Annual Conference of the Association for Computer Aided Design in Architecture*; CUMINCAD: Cambridge, MA, USA, 2013; pp. 101–108.
26. Autodesk Revit. 2020. Available online: https://www.autodesk.com/products/revit/overview?term=1-YEAR (accessed on 14 April 2021).
27. Autodesk Green Building Studio. Available online: https://knowledge.autodesk.com/search-result/caas/CloudHelp/cloudhelp/ENU/BPA-GBSWebService/files/GUID-9BD1341A-89D4-4A19-B4A9-6FDFF419A985-htm.html (accessed on 14 April 2021).
28. Asl, M.R.; Bergin, M.; Menter, A.; Yan, W. BIM-based parametric building energy performance multi-objective optimization. In *Proceedings of the 32nd eCAADe Conference*; CUMINCAD: Newcastle, UK, 2014; pp. 455–464.
29. Autodesk Dynamo. Available online: https://knowledge.autodesk.com/it/support/revit-products/learn-explore/caas/CloudHelp/cloudhelp/2018/ITA/Revit-Customize/files/GUID-F45641B0-830B-4FF8-A75C-693846E3513B-htm.html (accessed on 14 April 2021).
30. Asl, M.R.; Zarrinmehr, S.; Bergin, M.; Yan, W. BPOpt: A framework for BIM-based performance optimization. *Energy Build.* **2015**, *108*, 401–412.
31. Alshibani, A.; Alshamrani, O.S. ANN/BIM-based model for predicting the energy cost of residential buildings in Saudi Arabia. *J. Taibah Univ. Sci.* **2017**, *11*, 1317–1329.
32. Ma, G.; Liu, Y.; Shang, S. A building information model (BIM) and artificial neural network (ANN) based system for personal thermal comfort evaluation and energy efficient design of interior space. *Sustainability* **2019**, *11*, 4972. [CrossRef]
33. Sharif, S.A.; Hammad, A. Developing surrogate ANN for selecting near-optimal building energy renovation methods considering energy consumption, LCC and LCA. *J. Build. Eng.* **2019**, *25*, 100790.
34. Sharif, S.A.; Hammad, A. Simulation-based multi-objective optimization of institutional building renovation considering energy consumption, life-cycle cost and life-cycle assessment. *J. Build. Eng.* **2019**, *21*, 429–445.
35. Energy Analysis for Dynamo. Available online: http://core.thorntontomasetti.com/energy-analysis-for-dynamo/ (accessed on 14 April 2021).
36. Keras: The Python Deep Learning API. Available online: https://keras.io/ (accessed on 14 April 2021).
37. A Recipe for Training Neural Networks. Available online: http://karpathy.github.io/2019/04/25/recipe/ (accessed on 14 April 2021).
38. Xavier, G.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*; JMLR Workshop and Conference Proceedings: Sardinia, Italy, 2010; pp. 249–256.
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.
40. Pedamonti, D. Comparison of non-linear activation functions for deep neural networks on MNIST classification task. *arXiv* **2018**, arXiv:1804.02763.