# Energy Demand Prediction of Residential Buildings using Deep Learning

Seyed Milad Parvaneh
*Email address:* parvaneh@stanford.edu

**Abstract**

Energy consumption around the globe has been rising for many decades. A significant portion of this consumption occurs in residential buildings. Developing reliable methods to understand and predict energy use is essential in the global effort to become more sustainable. Many cities across the U.S. have mandatory energy benchmarking programs requiring large buildings to track and report their energy use. These openly available datasets have encouraged many researchers to study energy use and develop energy use prediction models. In this project, I use one of these available datasets and employ Artificial Neural Networks in the form of Deep Learning to predict building energy use in the City of Chicago. It is observed that these prediction models provide better results compared with typical baseline models such as those developed by Ordinary Least Squares regression. However, the amount by which better prediction accuracies are achieved is not significant.

## Motivation

Residential buildings accounted for 16% of the total energy use in the United States (U.S.) in 2019 [U.S. EIA]. Given that the global population is forecasted to increase at least until the end of the century, a proper understanding of building energy usage patterns can help us reliably predict energy demand in growing urban areas. This can then equip decision-makers with more reliable tools to properly dedicate resources for infrastructure in the energy sector, which would facilitate the path towards sustainable urban infrastructures.

Over the past decade, many cities across the U.S. started energy benchmarking programs requiring owners of non-residential and large multifamily buildings to track their energy consumption and annually report the data to municipal departments. Thanks to these freely available datasets, Machine Learning (ML) and its promising advances during the past decade have shown to provide a helpful toolkit in analyzing and estimating urban energy use. In this project, I use an energy consumption dataset for residential buildings in the City of Chicago and employ Deep Learning to develop models that aim to output Energy Use Intensity (EUI). Evaluation measures are then employed to compare and analyze the performance of the proposed models.

## Related Work

In order to predict building energy demand, one approach is the data-driven modeling in which ML techniques are used to train algorithms from available energy usage data and, then, the trained models are used to estimate energy use. This approach has shown to be promising when large-scale modeling is of interest and has been the focus of many research projects [Wei et al., 2018; Sun et al., 2020].

Within the data-driven approach, some supervised ML methods have gained more attention in estimating building energy usage. As of early 2017, Artificial Neural Networks (ANN) and Support Vector Machines (SVM) seemed to have gained the most interest in research studies by utilization ratios of 47% and 25% [Amasyali and El-Gohary, 2018]. Other commonly used methods are decision trees, such as Random Forest (RF) and Boosting Trees (BT), Multiple Linear Regression (MLR), and Ordinary Least Squares regression (OLS) among others. It is known that ANN and SVM provide better prediction accuracies, but at the cost of requiring many parameters and potentially becoming computationally expensive. Other algorithms require less computing power but sometimes at the expense of accuracy.

## Datasets and Features

In this project, I use an energy consumption dataset for residential buildings in the City of Chicago, CM-RBC 2017 [1], provided by the Complex and Sustainable Urban Networks (CSUN) lab at the University of Illinois. The dataset is the outcome of merging three major sources of data on multifamily residential buildings in Chicago: building attributes and energy data from the City of Chicago Energy Benchmarking database [Chicago Energy Benchmarking], census sociodemographic data from the American Community Survey (ACS) [U.S. Census Bureau], and buildings' envelope geometry and footprint data from the Chicago Building Footprint dataset [Chicago Building Footprint].

Table 1 summarizes each variable's statistical measures to provide a preliminary sense of the datasets' overall dispersion. Among the building features provided, three are most important: Gross Floor Area, Year Built, and Energy Star Score. These features are reported in the literature to be among the most significant contributing factors in energy consumption prediction of residential buildings [Abbasabadi et al., 2019]. Energy Star Score is a 1-to-100 rating that assesses a property's overall energy performance based on national data. A score of 50 represents the national median.

The target variable is site EUI. Site EUI is a property's site energy use divided by its gross floor area. Site energy use is the annual amount of all the energy consumed by the property on-site, as reported on utility bills. Data points with EUI greater than 1000 kBtu/ft$^2$, most likely outliers, are eliminated. Some of the other features provided are Building Height, Occupancy, and Household Size, among others. Further descriptions for each feature can be found at dataset references.

Table 1: Statistical measures of dataset variables.

| Variables | Mean ± SD | Variables | Mean ± SD |
|---|---|---|---|
| Gross Floor Area (ft$^2$) | 207760 ± 210269 | Single Occupied | 0.47 ± 0.17 |
| Year Built | 1965.62 ± 34.16 | Population w/o Diploma | 0.07 ± 0.09 |
| Energy Star Score | 55.94 ± 30.50 | Unemployed | 0.06 ± 0.06 |
| Log Site EUI (kBtu/ft$^2$) | 1.86 ± 0.19 | Crowded Housing | 0.02 ± 0.03 |
| Male Population | 0.48 ± 0.06 | Occupancy | 0.88 ± 0.09 |
| Population Under 18 | 0.10 ± 0.07 | Poverty | 0.17 ± 0.12 |
| Population Over 65 | 0.14 ± 0.11 | Labor Force | 0.69 ± 0.14 |
| Renter Occupied | 0.61 ± 0.21 | Number of Units | 29.13 ± 93.23 |
| Household Size | 2.43 ± 2.25 | Compactness | 12.39 ± 9.74 |
| Income ($) | 74217 ± 42464 | Building Height (ft) | 76.81 ± 122.77 |

## Methods and Evaluation Metrics

Two ML methods are employed in order to train prediction models. These methods are OLS and ANN. Dataset is split into two sets: one training/validation set (training only for OLS) consisting of 80% of the data points used for fitting the models and a test set with the remaining 20% used for model evaluation. As for validation, a 5-fold cross validation (CV) technique is adopted. Mean squared error is chosen as the CV score criterion. Hyperparameter tuning for each model is performed using both grid search and randomized approach. For all learning algorithms, the open-source `scikit-learn` [2] library is used.

### Ordinary Least Squares

Estimating energy consumption is considered to be a regression problem. Generalized Linear Models seem to be a proper first candidate to tackle such problems. To start, I utilize OLS to form a baseline. OLS tries to minimize the sum of the squares of the differences between the target variables and those

[1] https://csun.uic.edu/datasets.html
[2] https://scikit-learn.org/

predicted by the learning algorithm. Geometrically speaking, this is the sum of the squared distances of the variables mentioned above when measured vertically. The cost function that is aimed to be minimized has the form

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2 \tag{1}$$

where $\theta$ represents the parameters to be learned, $x^{(i)}$'s are input features, $y^{(i)}$'s are target variables, and $h_\theta$ denotes the hypothesis function (representing the predicted values).

### Artificial Neural Networks

Artificial Neural Networks (ANN) is the next learning algorithm employed. A typical ANN, inspired by a biological brain, consists of layers and neurons (nodes) within those layers. Each node within layer $l^{(i)}$ receives input data from some/all nodes of layer $l^{(i-1)}$. After processing the data, an output is sent to some/all nodes of layer $l^{(i+1)}$. Processing input data is mainly forming a weighted sum of the inputs plus a bias term, which is then passed through an activation function to from the output. As ANN is exposed to the training dataset, the weights are iteratively adjusted to have the cost function getting minimized. The cost function (for the regression problem) takes the form

$$J = \frac{1}{2}||y_{\text{predicted}} - y_{\text{actual}}||_2^2 + \frac{\alpha}{2}||W||_2^2 \tag{2}$$

where $\frac{\alpha}{2}||W||_2^2$ is an L2-regularization term. The equation above is borrowed from `scikit-learn` [3] library, which implies that the algorithm uses the square error loss function. The reader is referred to the corresponding reference for definitions of parameters and further descriptions.

### Evaluation Metrics

To evaluate the results qualitatively, scatter plots are provided that compare the measured target variable versus the predicted values. In terms of evaluating the results quantitatively, Coefficient of Variation (CoV), Mean Absolute Percentage Error (MAPE), and Root Mean Square Error (RMSE) are commonly-used metrics in the literature [Amasyali and El-Gohary, 2018]. Here, I use MAPE, Mean Squared Error (MSE), and $R^2$.

## Experiments, Results, and Discussion

The two learning algorithms described above are used to train prediction models. OLS is primarily used as a simple linear regression model in order to provide a baseline. This helps with getting a sense of the dataset and realizing the possible improvements that can be made. It also provides a benchmark by which we can evaluate the performance of the other prediction models.

ANN, or more precisely Multi-Layer Perceptron (MLP) regressor as labeled in the `scikit-learn` library, has the following hyperparameters: *alpha*, *hidden_layer_sizes*, *learning_rate*, and *learning_rate_init*. First, *lbfgs* is used as the solver which is an optimizer in the family of quasi-Newton methods and can converge faster and perform better for small datasets. For this solver, the hyperparameters *learning_rate* and *learning_rate_init* are not used. The remaining parameters are tuned as follows: *alpha* works as a regularization parameter. For tuning *alpha*, a range between $1.0 \times 10^{-4}$ and $1.0 \times 10^{+1}$ is considered. A logarithmic uniform distribution is employed to sample random *alpha*'s from the given range. For *hidden_layer_sizes*, different network architectures are tried out (see Table 2). Having three hidden layer of $(20, 7, 5)$ neurons seems to provide the best results.

Second, stochastic gradient descent (*sgd*) is used as the solver for which hyperparameters *learning_rate* and *learning_rate_init* need to be tuned in addition to the parameters explained above. *learn-*

[3] https://scikit-learn.org/stable/modules/neural_networks_supervised.html

Table 2: Network architectures.

| | | |
|---|---|---|
| (20, ) | (20, 10, 5) | (40, 40, 40) |
| (20, 20) | (20, 20, 5) | (40, 20, 10) |
| (20, 20, 20) | (20, 7, 5) | (40, 20, 5) |
| (20, 20, 20, 20) | (20, 7, 3) | (40, 40, 5) |
| (20, 20, 20, 20, 20) | (20, 5, 1) | (40, 60, 20) |

*ing_rate* controls the scheduling for weight updates. Two cases of *constant* and *adaptive* are examined. *learning_rate_init* represents the initial learning rate, and controls the step-size by which the weights are updated. Two ranges are considered: one between $1.0 \times 10^{-6}$ and $1.0 \times 10^{-1}$ in which a logarithmic uniform distribution is employed to sample random values, and the other between $1.0 \times 10^{-1}$ and 1.0 that uses a uniform distribution to sample. Also, for both solvers, the *relu* function is employed for activation. Multi-layer Perceptron is sensitive to feature scaling, so the data is scaled (i.e., standardized to have mean 0 and variance 1).

Figure 1 shows the scatter plots of the measured target variable versus the predicted values for each learning algorithm. The solid pink line at 45 degrees depicts the zero error line along which measured and predicted Log EUIs are equal. The dashed pink lines represent ±25% error threshold. Because the Log EUI of most buildings lies within the range of 1.5 to 2.5 kBtu/ft$^2$, the x-axis limits are set to 1.5 to 2.5 kBtu/ft$^2$ for clarity. From a qualitative point of view, by comparing the scatter plots, we see that ANN has better performance than OLS since more data points lie within the marked error range. Table 3 shows the evaluation measures, including Mean CV Score, MAPE, MSE, and $R^2$. CV score is calculated for ANN performed on the training dataset and provided as the average score of five CV folds. MAPE, MSE, and $R^2$ are calculated for predictions on the test dataset. Looking at the metric values, ANN seems to bring better performance. However, the amounts by which better evaluation scores are achieved do not seem to be significant.

Among the various combination of hyperparameters tried, two scenarios are selected to plot the average CV scores (MSE), which are shown in figure 2. Figure 2a is a line chart in which each line represents an ANN architecture. For this scenario, the *lbfgs* solver is used. These architectures are

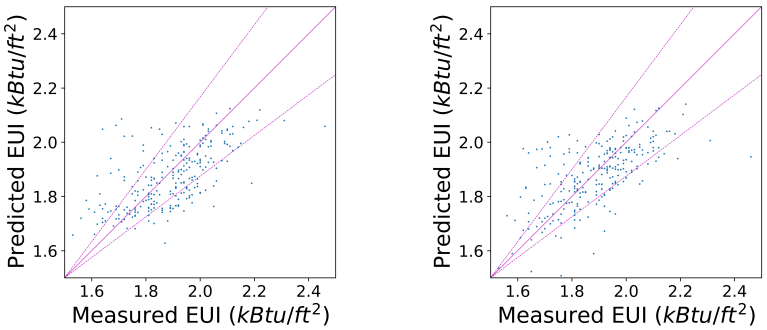


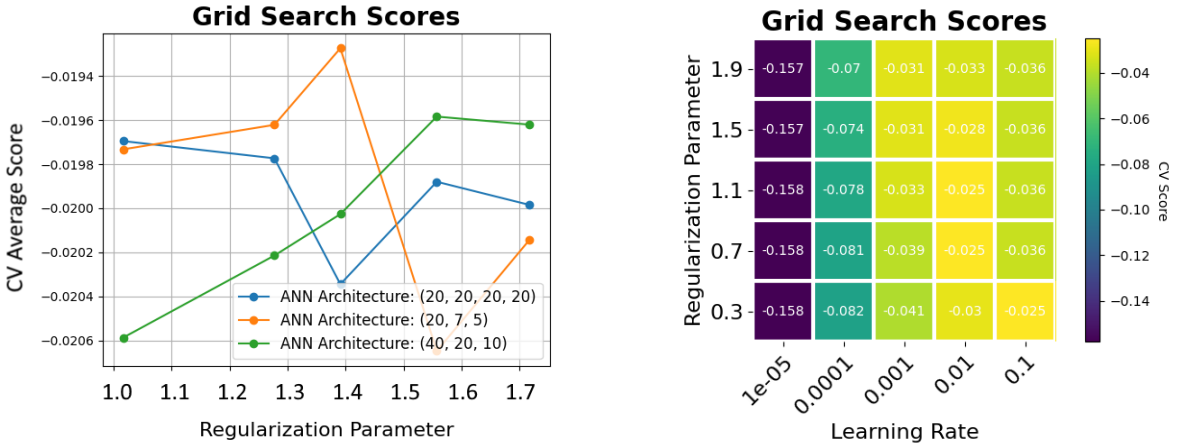

Figure 1: Scatter plot of the measured target variable vs. the predicted value for (a) OLS, and (b) ANN.

Table 3: Evaluation metrics for the prediction models.

| Metrics | OLS | ANN |
|---|---|---|
| Mean CV Score (MSE) | n/a | 0.019 |
| MAPE | 6.346 | 6.042 |
| MSE | 0.028 | 0.026 |
| $R^2$ | 0.308 | 0.353 |

selected based on their top performance among the set of architectures they are compared with. Average CV scores are then plotted for different values of *alpha*. It can be observed that $(20, 7, 5)$ with *alpha* $\simeq 1.4$ captures the best score. However, as *alpha* increases, it starts to underperform the other architectures. $(40, 20, 10)$ seems to have an overall rising score as *alpha* gets larger.

Figure 2b is a heatmap chart that shows CV scores for ANN structure $(20, 7, 5)$ using different *learning_rate_init* and *alpha* values. For this plot, the *sgd* solver is employed using *adaptive* learning rate. First thing to notice is the similar coloring pattern within each given column. It seems that changing *alpha* does not affect the scores especially when relatively small learning rates are adopted. Also, *sgd* does not seem to converge to better solutions compared with *lbfgs* solver.




(a) (b)

Figure 2: (a) Grid search average CV scores (MSE) for different ANN structures and regularization parameter values, and (b) Grid search scores for ANN structure of $(20, 7, 5)$ using different values for *learning_rate_init* and *alpha*.

## Conclusion and Future Work

In this project, two ML algorithms, including OLS and ANN, were employed to train prediction models in order to estimate energy consumption in residential buildings in the City of Chicago. First, a baseline was made using a simple linear regression model, OLS. This helps with getting a sense of the dataset and realizing the possible improvements that can be made. Then, ANN was used in the form of deep learning to introduce more nonlinearity and train more sophisticated prediction models.

The performance of the ML methods was compared qualitatively using measured variable vs. predicted value plots. Then, several evaluation metrics, including MAPE, MSE, and $R^2$, were employed to measure how each algorithm performs. Also a randomized approach for tuning hyperparameters was carried out that helped with improving the prediction model performance. Overall, I was able to use ANN to train a prediction model that brings better prediction accuracy compared with a typical baseline model such as one developed by OLS regression. However, the amount by which better prediction accuracies are achieved is not significant. This can be due to the noisy nature of the problem at hand which can be revealed by plotting some of the main dataset input features. Also, due to time constraints, not all ANN architectures were explored here which leaves some room for improvement.

For future work, one idea is to extend the project to cover other types of buildings. This would probably need to analyze each building type independently because consumption patterns seem highly dependent on building types and their application. Also, similar datasets are likely available from other metropolitan areas, e.g., New York City and San Francisco. Having the trained model, one may try those datasets to observe the proposed model performance and discuss possible sources of prediction failure that were not considered, e.g., weather conditions, energy cost, and sociocultural differences.

# References


N. Abbasabadi, M. Ashayeri, R. Azari, B. Stephens, and M. Heidarinejad. An integrated data-driven framework for urban energy use modeling (UEUM). *Applied Energy*, 253:113550, 2019.

K. Amasyali and N. M. El-Gohary. A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews*, 81:1192–1205, 2018.

Y. Sun, F. Haghighat, and B. C. Fung. A review of the-state-of-the-art in data-driven approaches for building energy prediction. *Energy and Buildings*, page 110022, 2020.

Chicago Building Footprint. URL https://data.cityofchicago.org/Buildings/Building-Footprints-current-/hz9b-7nh8?category=Buildings&view_name=Building-Footprints-current-. Accessed: 2021-04-18.

Chicago Energy Benchmarking. URL https://data.cityofchicago.org/Environment-Sustainable-Development/Chicago-Energy-Benchmarking-2017-Data-Reported-in-/j2ev-2azp. Accessed: 2021-04-18.

U.S. Census Bureau. URL https://www.census.gov/acs/www/data/data-tables-and-tools/data-profiles/2017/. Accessed: 2021-04-18.

U.S. EIA. URL https://www.eia.gov/totalenergy/data/monthly/pdf/flow/css_2019_energy.pdf. Accessed: 2021-04-18.

Y. Wei, X. Zhang, Y. Shi, L. Xia, S. Pan, J. Wu, M. Han, and X. Zhao. A review of data-driven approaches for prediction and classification of building energy consumption. *Renewable and Sustainable Energy Reviews*, 82:1027–1047, 2018.