Quebec Energy Consumption Prediction

Philippe Joly – 261205161
philippe.joly3@mail.mcgill.ca

2024-03-16

Project Statement

The goal of this project is to use past energy usage data supplied by Hydro-Québec [2] to estimate future needs in energy. To predict energy demand, energy consumption data since 2019 [2], coupled with weather data [1] will be used as features for machine learning models of various architectures, approaching the problem from both a non-sequential and time series approach. This project attempts to predict something complex like energy usage, using common predictions like weather.

Data Processing

The weather and population data were scraped from *Climate Canada* [1] and *Statistics Canada* [**?**] respectively. Population, GDP, and Household Disposable Income, were abstracted as these metrics did not fluctuate significantly during the observed period (2019-2022). Weather data was taken from four weather stations: Montreal, Quebec, Sherbrooke, and Gatineau. The weather value was then calculated as a weighted average of these hourly values based on the regions' population. The data was then concatenated through `csv` manipulation and posted to *Kaggle*. At training, data was normalized. Furthermore, for some models, the temporal data like hour, day, and month were converted to sinusoidal waves to capture their inherent cyclical nature.

Methodology

- Splits: The training, validation, and test sets were generated following a 90/5/5% split using *SkLearn*.
- Training: The training was completed on various batch sizes with the mean squared error loss using the *ADAM* optimizer.
- Performance Metrics: The performance of each algorithm will then be compared using the mean absolute error (MAE) and coefficient of determination (R^2) metrics.

Standard Models

- Support Vector Regression (SVR): The SVR model is implemented using the *SkLearn* library [4]. Using a cross-validation Bayesian search, optimal parameters were found for the model. The model was then further trained using the following.
- Deep Neural Network (DNN) [1]: The DNN is implemented using *PyTorch* framework [3]. The model is constructed with an arbitrary architecture of 6 hidden layers of 5 fully-connected nodes each.
- Convolutional Neural Network (CNN): Not yet implemented

Sequential Models

- Long Short-Term Memory Network (LSTM) [2]: The LSTM model is composed of a depth of 3 layers with 150 hidden units. This is connected to a fully connected layer resulting in the prediction. It is implemented using *PyTorch* framework [3].
- Recurrent Neural Network (RNN) [3]: The RNN model is composed of a depth of 4 layers with 150 hidden units. This is connected to a fully connected layer resulting in the prediction. It is implemented using *PyTorch* framework [3].
- Gated Recurrent Units (GRU) [4]: The RNN model is composed of a depth of 5 layers with 150 hidden units. This is connected to a fully connected layer resulting in the prediction. It is implemented using *PyTorch* framework [3].
- Sequential Convolutional Neural Network (SCNN) [5]: The SCNN model is composed of 2 convolutional layer followed by three fully-connected linear layers. It is implemented using *PyTorch* framework [3].
- Hybrid Architecture: The goal is to implement a hybrid architecture, combining convolution layers before a more common model for time series such as LSTM or GRU.

Preliminary Results

Models	Mean Absolute Error (MAE)	R^2
SVR	568.183	0.982
DNN	719.876	0.965
LSTM	302.864	0.953
RNN	254.381	0.964
GRU	194.613	0.981
SCNN	167.645	0.985

Table 1: Validation Scores for All Implemented Models

Next Steps

Implementing a non-sequential convolutional neural netweork as well as a hybrid time series model would be ideal if time allows. The implemented models can be further tuned. Also it is time to start creating a web integration tool to demonstrate the models' capacities.

References

[1] Environment and Natural Resources Canada. Past weather and climate historical data, 2024. URL: https://climate.weather.gc.ca/historical_data/search_historic_data_e.html.

[2] Hydro-Québec. Electricity demand in québec, 2024. URL: https://www.hydroquebec.com/documents-data/open-data/electricity-demand-quebec/.

[3] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. *CoRR*, abs/1912.01703, 2019. URL: http://arxiv.org/abs/1912.01703, arXiv:1912.01703.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Appendix: Training Loss

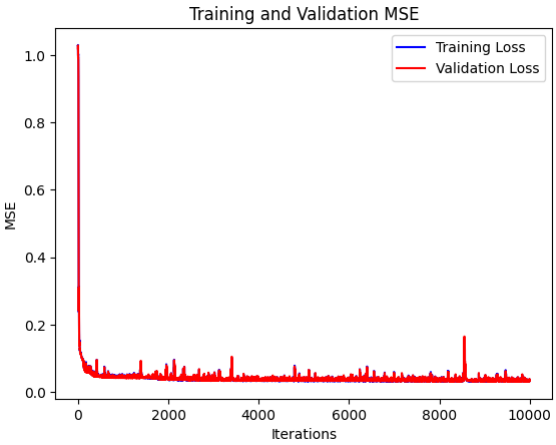

Figure 1: DNN training and Validation Loss

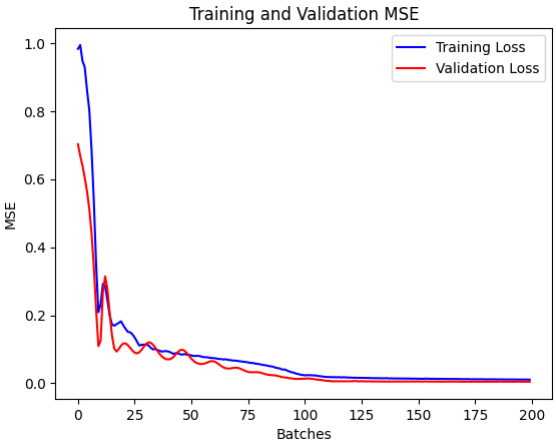

Figure 2: LSTM training and Validation Loss

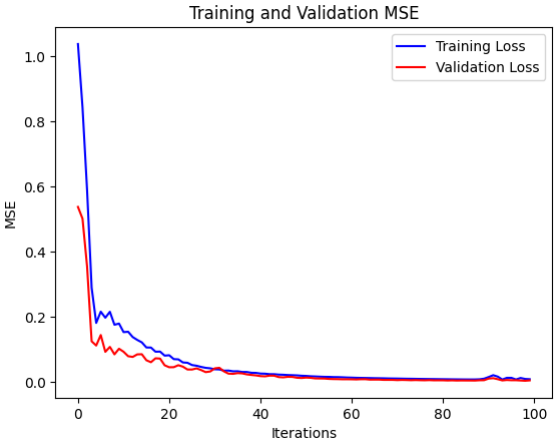

Figure 3: RNN training and Validation Loss

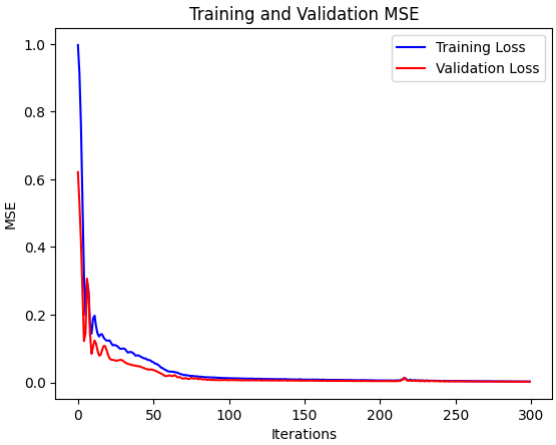

Figure 4: GRU training and Validation Loss

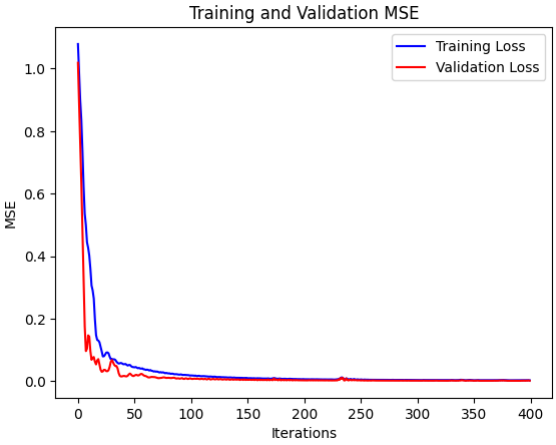

Figure 5: SCNN training and Validation Loss