Electric Load Disaggregation at Varying Timescales

Prepared for: EE4951W Senior Design

Ben Moorlach, Brandon Gorter, Zach Xiong, Jon Skarda, Zeb Zimmer
University of Minnesota
Electrical and Computer Engineering
Minneapolis, MN

Advised by Professor Giannakis, Professor Dhople, and (PhD) Manish Singh

December 22nd 2022

Ben Moorlach, Brandon Gorter, Jon Skarda, Zach Xiong and Zeb Zimmer
2634 Blaisdell Ave S. Apt 203
Minneapolis, Minnesota 55408

Undergraduate Students of Computer and Electrical Engineering
University of Minnesota College of Science and Engineering
117 Pleasant St SE, Minneapolis, MN 55455

To Whom it May Concern,

Within the attached project report, we will outline the processes and outputs of our Senior Design Engineering project. The reader will gain limited background knowledge of electric load disaggregation, an understanding of our application of electric load disaggregation theory and technology, and our final impressions of our project outputs as they relate to the current state of this field of engineering.

Our process has involved extensive research into the background and current state of electric load disaggregation technology. In this write-up, we provide hardware recommendations and reports on our use of said hardware in tandem with two common disaggregation algorithms. The ultimate goal of our project was to determine optimal timescales of input waveforms in regards to computational demands and output waveform granularity. Included in this report, is a summary of the data we collected and our impressions, related to our research goal, of this data.

We appreciate the opportunity to have worked on this project while being supported by University of Minnesota staff and resources. Please do not hesitate to contact project group members with any questions or recommendations for future work.

Sincerely,

Ben Moorlach, Brandon Gorter, Jon Skarda, Zach Xiong and Zeb Zimmer

Table of Contents

Abstract

An *Energy Signal* (i.e. of a household) describes a measured consumption of energy over time that can be represented as a waveform. *Electric Load Disaggregation* (ELD) is the process of discerning the energy signals of individual entities (i.e. household electric appliances) from their aggregate, or combined, energy signal over an arbitrary sampling period. The benefits of applying such a capability towards collecting mass energy-consumption data could include helping consumers in identifying energy-wasting habits, allowing power companies to perform automatic shut-offs to prevent waste and power grid failures, and informing forecasting of future energy needs, to name a few. Our project was an attempt to implement a device and complimentary software capable of performing ELD using a consumer-grade smart power meter and algorithms adopted from the *NILMTK* Python library. Our project goals were to collect our own real world data, disaggregate our own data in order to evaluate the performance of the algorithms at varying timescales, and finally, interpret the results and present our findings. We concluded that 30 or 60 second timescales were optimal in regards to granularity of output power waveforms and computational burden.

Problem Definition

Since its proposal in 1992 by George Hart, electric load disaggregation has been a highly researched topic. The concept is simple: extract appliance-level electric power usage data by looking only at the aggregate waveform, such as the load on an entire home's power meter. This is done algorithmically, through various supervised and unsupervised machine learning methods. The more robust algorithms are commonly supervised models, however, unsupervised models, while not normally as accurate, have the advantage of being less intrusive to homeowners. Learning-based electric load disaggregation is often referred to as NILM in the research community, and it stands for Non-Intrusive Load Monitoring. The non-intrusive nature of NILM is due to the fact that appliance-level meters do not need to be installed in order to derive their usage. This provides benefits to homeowners and energy providers who may gain energy and cost savings from the data, but this very same idea raises privacy concerns. It is within the scope of our work on this subject to assess at which timescales data may be collected in order to accurately reconstruct appliance-level data. This information will help optimize the deployment of NILM algorithms and provide a threshold at which the homeowner's privacy is protected.

The three main goals of this project, as defined by our advisors, were as follows: 1) collect real load data to disaggregate per-appliance consumption; 2) analyze performance and tradeoffs at varying timescales; and 3) use meta and transfer learning to adapt to unseen devices. Goals 1 and 2 were critical to the success of this project, while goal 3 was a stretch goal meant to be completed if and only if we have time after completing the first two.

To achieve these goals, our design was stated to include: 1) supervised and unsupervised load disaggregation algorithms developed using contemporary machine learning tools. 2) a comparison of the performance of load disaggregation algorithms at varying timescales using ground-truth data. 3) a demonstration of the use of an algorithm in real-time using seen and unseen appliances at the design show.

Concept Design

The concept of electric load disaggregation has been around for over three decades. The recent inclusion of machine learning in the space increased research attention and subsequently the number of scholarly papers focusing on attempted solutions. The idea generation process for our team involved reading 11 of these papers that related to electric load disaggregation to understand the work being done and how these other teams created their research environments. The 7 of those papers that ended up contributing to our final design were:

1. "Nonintrusive Appliance Load Monitoring"[1], George W. Hart. This paper gave us lots of information about the being of NILM and eventually we went on to use the proposed algorithm: AS-NALM algorithm, in our final design.
2. "Unsupervised disaggregation of appliances using aggregated consumption data"[2], H. Goncalves. This paper gave us insight to how accurate we could hope the design could be. There is extensive information in this paper about how

certain appliances make unsupervised algorithms, like the AS-NALM algorithm, perform poorly.

3. “Energy Disaggregation via Discriminative Sparse Coding”[3], J. Zico Kolter, Siddarth Batra, and Andrew Y. Ng. The main takeaway was that a very low resolution time scale as low as 15 minutes to an hour can be effective in classifying what devices are on but will not give accurate individual waveforms.

4. “Neural NILM: Deep neural networks applied to energy disaggregation”[4], Jack Kelly and William Knottenbelt. These researchers took a very novel approach by taking an image of the waveform and doing visual analysis using machine learning. The main takeaway for us was that sampling window size needs to capture the majority of appliance activations in training. This was a key concept that we used throughout the design process.

5. “Sequence-to-point learning with neural networks for non-intrusive load monitoring”[5], Chaoyun Zhang et al. We used this paper as inspiration for using a neural network to disaggregate. We eventually spent considerable time working with a recurrent neural network with a similar set up.

6. “Low-complexity energy disaggregation using appliance load modeling”[6], Hana Altrabalsi et al. This paper gave us ideas about more efficient ways to do disaggregation. We did not end up going with the ideas used in the paper because we ended up not necessarily needing an efficient solution.

7. “Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014 (AMPds2)”[7], Stephen Makonin et al. These researchers released, alongside the paper, a public dataset for use in electric load disaggregation. We used this dataset extensively in our testing of various algorithms.

The team took what was gained from these research papers and took an attempt on our own design. We needed to collect data too, which is separate from the goals of any of the papers, so the use of an ACUVIM II power meter was employed. Later on in the design process when the ACUVIM II was failing to reach our design goals an Emporia Smart Plug was used.

Design Description

In order to test the effectiveness of using machine learning to disaggregate an energy load signal at different timescales, we need to collect our own data. We researched different methods to collect this data and iterated through different devices, and eventually landed on the EMPORIA smart plug. This smart plug was able to collect data about the devices which were plugged into it, and transmitted that data over wifi to a smartphone app. We then exported that data in CSV format to our laptops so we could use it.

Once we had access to the CSV files of our data, we used python scripts we produced to artificially extend the length of the dataset, alter the timesteps of the CSV files, and create aggregate CSV files which contain all of the appliance power usage summed together. After these scripts were run, we condensed all of the files into a single file in the format of a .h5 file. This .h5 file contained all of the individual appliance level ground truths, as well as the aggregate data. It also needed to contain metadata which is required by NILMTK, a Python toolkit we used which is designed to make dealing with datasets used for the energy disaggregation task easy.

Once our data was in the .h5 file format, we used NILMTK to split the dataset into train ranges and test ranges by inputting the windows of the dates we wanted to specify into each section. Our dataset contained data for 96 hours, so we used the first 72 hours to train our models and the last 24 hours to test our models. We trained supervised and unsupervised models for each of our timescales: 1 second, 8 seconds, 15 seconds, 30 seconds, and 60 seconds. After we had our models trained, we used the data we specified to be test data to test our models, compared the results to the ground truth data, and evaluated the performance of the model. The block diagram of this design is shown below in *Figure 1*.

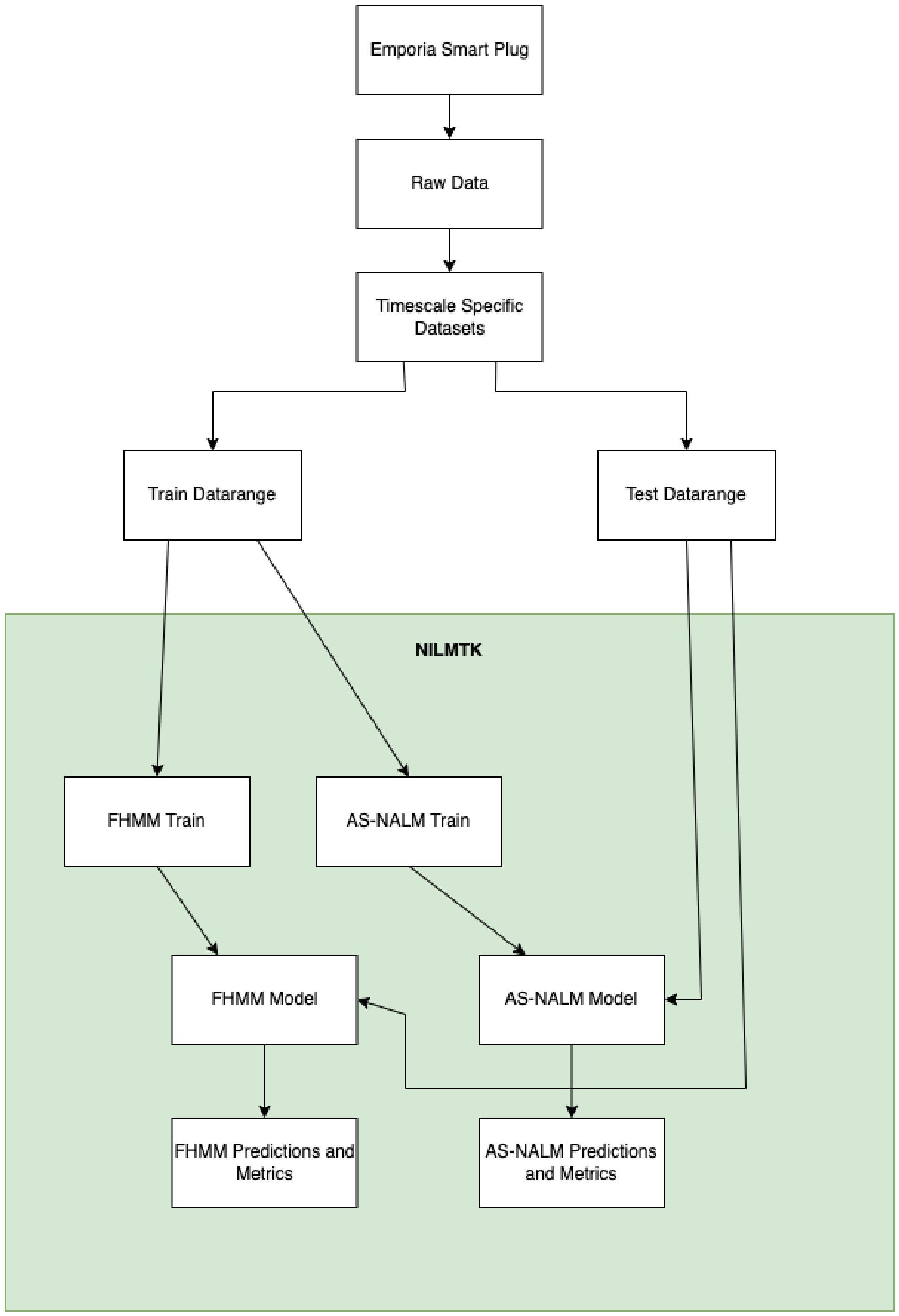


Figure 1: A hardware-software-integration block diagram of our design.

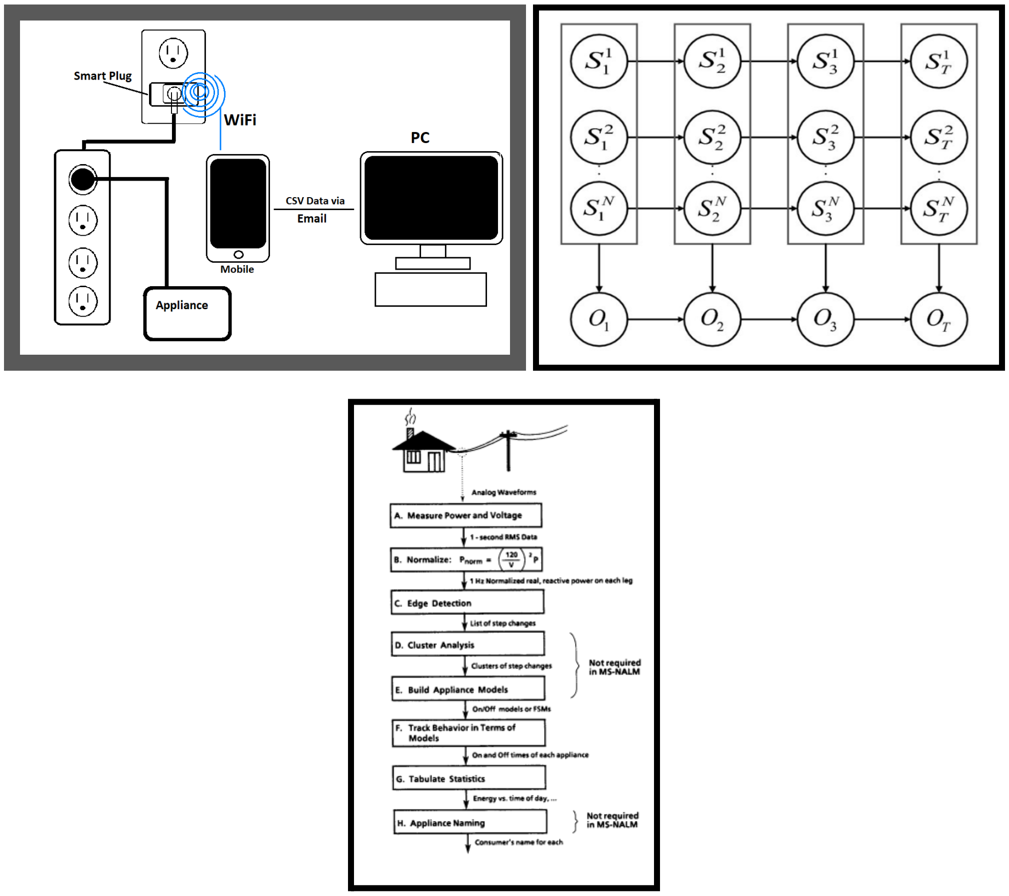

Smart Plug
WiFi
PC
CSV Data via Email
Mobile
Appliance
S_1^1 S_2^1 S_3^1 S_T^1
S_1^2 S_2^2 S_3^2 S_T^2
S_1^N S_2^N S_3^N S_T^N
O_1 O_2 O_3 O_T
Analog Waveforms
A. Measure Power and Voltage
1 - second RMS Data
B. Normalize: $P_{norm} = \left(\frac{120}{V}\right)^2 P$
1 Hz Normalized real, reactive power on each leg
C. Edge Detection
List of step changes
D. Cluster Analysis
Clusters of step changes
E. Build Appliance Models
Not required in MS-NALM
On/Off models or FSMs
F. Track Behavior in Terms of Models
On and Off times of each appliance
G. Tabulate Statistics
Energy vs. time of day, ...
H. Appliance Naming
Not required in MS-NALM
Consumer's name for each

Figure 2. ***(Top Left)*** **Functional Hardware Diagram of Project Output.** ***(Top Right)*** **FHMM algorithm data-training diagram. S objects represent appliance states, O objects represent observe or aggregate states.** ***(Bottom)*** **Block diagram of Hart model algorithm.**

Design Evaluation

In order to evaluate our design, we went through an initial stage of both software and hardware testing before arriving at a more developed design with refined auxiliary python scripts and an alternative power meter. This second design iteration allowed us to generate actual performance metrics at varying timescales.

For software, we started by attempting to disaggregate public datasets from UK-DALE and AMPds with our newly implemented algorithms from NILMTK and a self built neural network algorithm on Tensorflow. The UK-DALE (UK Domestic

Appliance-Level Electricity) dataset had 54 separate appliance meters for a single building and AMPds (Almanac of Minutely Power dataset) had 21 meters [7]. Both stored data in the form of an .h5 file which was exactly the form that our algorithms used. We found that the Hart85 algorithm from NILMTK successfully disaggregated AMPds data, the neural network on Tensorflow successfully disaggregated UK-DALE data, and the FHMM algorithm from NILMTK worked marginally with either dataset. This is due to the fact that the FHMM model was better developed for smaller appliance datasets, so the dozens of ground truth power meters that were aggregated in UK-DALE and AMPds may have posed an issue. Note that, since we would be collecting our own real data on fewer appliances, we would continue to evaluate all three algorithms with data we've generated ourselves later. However, this first stage proved the possibility of using these machine learning algorithms.

With hardware, we initially attempted to start data collection with an Acuvim II Series Advanced Energy Power Meter (*Figure 4*). While this power meter had many features that could allow it to generate powerful and vast data sets, there were existential issues we encountered. Although we had wired the setup correctly and validated it with the given circuit diagrams in the device manual, there were abnormal power readings returned by the meter. Current readings (via a current transformer) and voltage references fluctuated between incorrect values when turning a known load on and off. However, this could be attributed to errors within the current transformer.

With the assistance of Dr. Manish Singh, we were able to isolate a concrete indicator: the power factor ratio. The PF ratio of our load could not have been affected by our current transformer, unlike the direct current readings and apparent power (kVA). The power factor identifies a load as being more resistive or more reactive, with a purely resistive load taking on a power factor of 1. When the load, a toaster with a purely resistive heating element, is turned on, the power factor abnormally takes on a value close to zero (*Figure 3*). This is opposite of what we would expect, and indicated that the Acuvim II had issues beyond the scope of our project.

As an alternative to the Acuvim II, we purchased the Emporia Smart Outlet (*Figure 4*), which, as mentioned in the previous section, was able to correctly read and transmit .csv data over wifi for further processing.

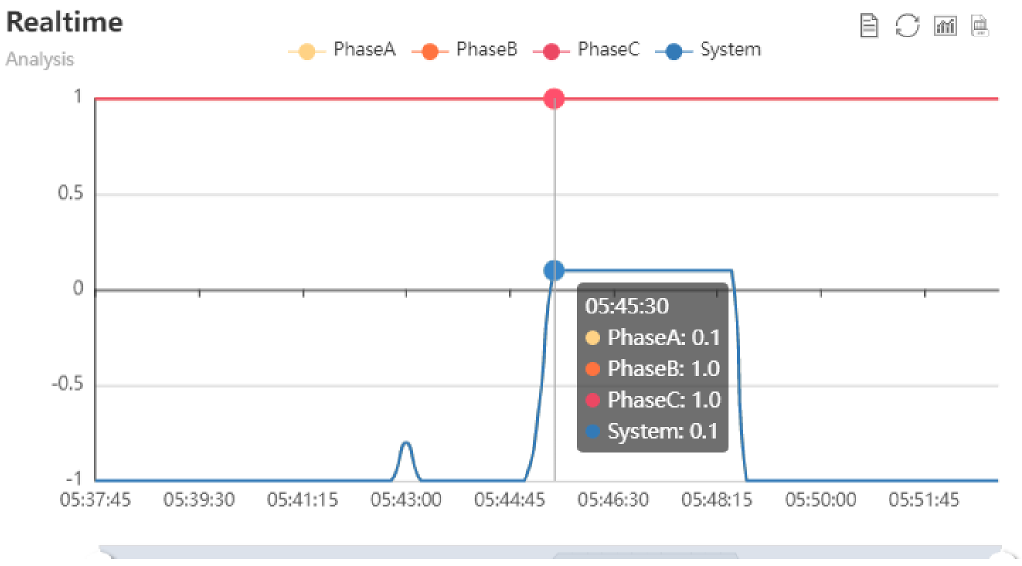


Figure 3: Power Factor Value of Phase A/System Line. Toaster is on between 5:44:45 and 5:48:15.

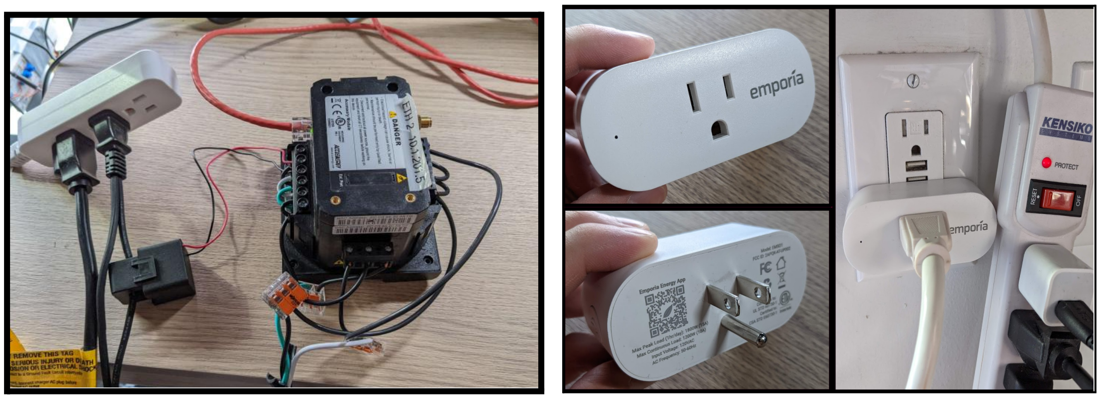


Figure 4: Power Meter Setup
(Left) **Acuvim II Series Power Meter**
(*Right)* EMPORIA Power Meter.

This allowed us to begin collecting data for six different appliances with varying usage durations. Most datasets for each appliance were collected for around 30 minutes, and some appliances were turned on and off multiple times in order to collect power changes and capture the waveform. One exception to this was the fridge appliance, which we collected multiple hours of data for overnight in order to accurately capture the device's power cycling behavior.

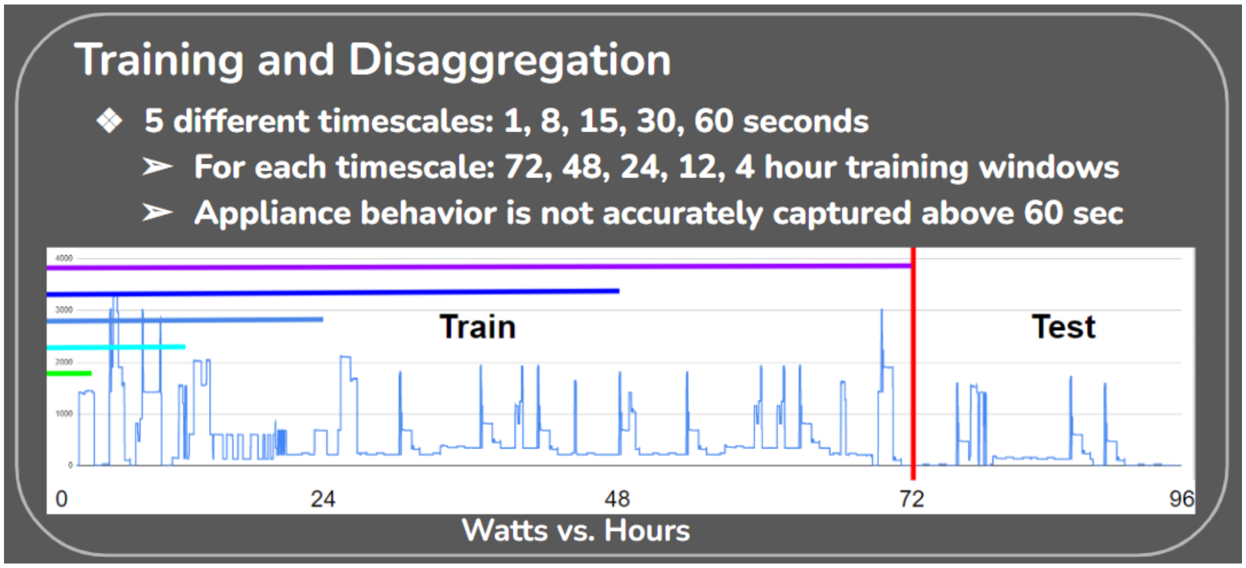
After collecting this 30 minute data, we then isolated waveforms found from these data sets and copied and stitched them together into much longer signals. The first signal we constructed was a 12 hour data set created from hypothetical usage of the six appliances from 8am to 8pm, with justifications explained in *Figure 5*.

Justification for Constructed Data Hypotheticals:
12 Hours, from 8am to 8pm

1. Fridge
    a. Running constantly for all 12 hours
2. Kettle
    a. Running a couple times for breakfast - 8:30am
    b. Running a couple times for lunch - 12:30pm
    c. Running a couple times for evening - 7:00pm
3. Microwave
    a. Running a couple times for breakfast, lunch, dinner - 8:30am, 12:30pm, 6:00pm
    b. Afternoon snack - 3:30pm
    c. Brunch snack - 10:30am
4. Toaster
    a. Running a couple times for breakfast - 8:30am
    b. Running a couple times for lunch - 12:30pm
5. Hair Dryer
    a. Running a couple times for 8:15am
    b. Running a couple times for 7:30pm
6. Slow Cooker
    a. Running for a few hours from 9 to 12pm
    b. Running for a few hours from 4 to 6pm

Figure 5: Constructed Data Justifications for Hypothetical Usage

This 12 hour dataset however, was not long enough to provide accurate algorithm training and testing. We then created a 96 hour version of the dataset using similar methods as before, and used the 12 hour hypothetical set within the last 24 hours that we wanted to use for testing the models and running a performance analysis. We trained the model on various lengths of time, with these “training windows” ranging from 4 to 72 hours long. This allowed us to verify that we could use more hours of data without overfitting the models. Additionally, we developed python scripts that could automatically calculate a moving average in order to vary the timescales from the range of 1 to 60 seconds. An example of the aggregate waveform at the 1 second timescale is shown in *Figure 6.*



Training and Disaggregation

- 5 different timescales: 1, 8, 15, 30, 60 seconds
  - For each timescale: 72, 48, 24, 12, 4 hour training windows
  - Appliance behavior is not accurately captured above 60 sec

Figure 6: Constructed Data Set for both Training and Testing the Algorithms.

From disaggregating the last 24 hour test section of the 96 hour signal, we could generate performance metrics in the form of an F1 score that is based on a precision and recall comparison between the algorithm predicted appliance usage and the actual individual ground truths. We discovered that 72 hour training data was not overfitting the data and so used that window for every test. Furthermore, of the three algorithms we planned on testing, the neural network developed on TensorFlow was unable to work properly with the data that we had collected. However, the FHMM model was able to correctly work with our small number of appliance data, so we were able to draw results from this algorithm as well as the Hart85 algorithm (*Figure 7*).

As shown, both models had a spread of F1 score accuracy over timescale and appliance type. We noticed an overall decrease in accuracy as the timescale was increased; however, there were anomalous increases in accuracy at the 30 second timescale. Finally, we also noticed a clear correlation between decreased F1 scores and appliances that had a more random, sparse and high power draw usage than appliances that had waveforms which were more regular. For instance, the power cycling of the fridge, depicted on the right in *Figure 7*, had some of the highest F1 scores, while more intermittently used appliances like the toaster and hairdryer did significantly worse.

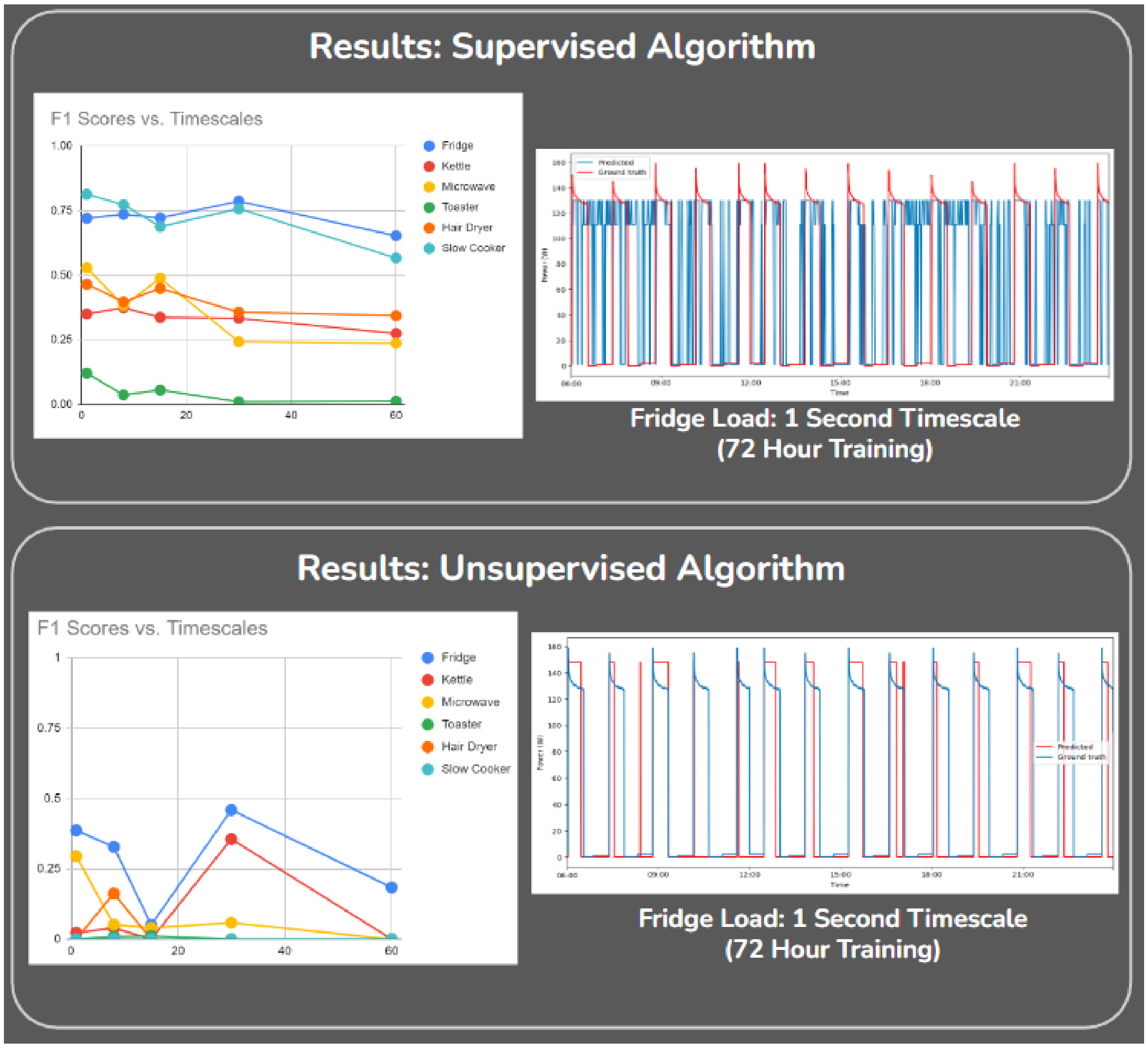


Figure 7: Performance results of supervised and unsupervised models over varying timescales.

Conclusions and Recommendations

Our final design, which consists of the supervised and unsupervised algorithms, along with a custom appliance load dataset and the resulting performance metrics, completes the first two goals of our project. Unfortunately, there was not enough time to achieve the third stretch goal of implementing meta and transfer learning to our algorithms.

In terms of cost, all of the code and NILM algorithms we leveraged were free, open-source material, with the only expenditure of this project being the smart plugs, which we purchased at roughly $10 each. Earlier on in the design process, we were attempting to use a more sophisticated power meter, the ACUVIM II Series, which had the potential to supply real-time data directly to our algorithms without the overhead of a smartphone app, as with the EMPORIA smart plugs. This turned out to be a dead end, since it required a current transformer, which we could not get to work and had to scrap. Troubleshooting problems with a power meter was not worth the convenience of real-time data. This also meant, however, that a live demo of our disaggregation algorithm working on appliances in real-time would no longer be feasible.

Obtaining a working power meter was not our only struggle. Our initial timeline had us obtaining a NILM model working on publically available datasets by the end of September. Due to the incredibly steep learning curve of NILMTK and all of the background research we needed to do to understand the NILM field, this was not accomplished until mid-October. This also delayed our data collection, since we needed to know what type of inputs our algorithm would take before then. By mid-October our algorithms were decided: we would use the Hart 85 (H85) unsupervised algorithm and a Recurrent Neural Network (RNN) supervised algorithm. We decided that the ACUVIM II power meter we had access to would be ideal. After a few more weeks, in early November, we had the RNN working on the UK-DALE dataset, but the ACUVIM II power meter was still not working, after trying two different current transformers. It was decided then that we would switch to the EMPORIA smart plugs and sacrifice our live demo. The next two weeks consisted of lots of data collection and one final big change. Once we had our own data collected and properly formatted, we discovered that the

RNN did not perform well on our data, likely due to a very specific formatting issue which we did not have time to debug. We then made the last-minute switch to the FHMM supervised model for performance analysis.

Due to the artificial nature of our data, these algorithms did not perform as well as expected; however, there were enough differences between sample rates and time windows to extract somewhat useful conclusions. The performance results are summarized above in *Figure 7*. There are two useful takeaways we obtained from this data. First is that, as shown by the prediction graph for the unsupervised model, this model was great at predicting waveforms with a periodic nature, such as the fridge. Also, for both the supervised and unsupervised models, there is a slight downward trend in performance with increasing timescale, but it is hardly significant. This led us to conclude that there is no significant performance advantage to changing the timescale for NILM machine learning models. It would be best both in terms of the granularity of the output power waveforms and in terms of the computational burden of high sampling to choose a 30 or 60 second timescale for load disaggregation.

Despite the many challenges we faced, and the less than stellar results we achieved, we would still consider this project a mild success. Looking back on this project and all of our work, it was a monumental task to begin with. Not one of us had any significant background in machine learning, so the fact that we were able to utilize these algorithms to produce decent results on public datasets and on our own data, is an accomplishment on its own.

If we had more time, we would make a couple of significant changes. First is to spend the time and effort to implement a newer NILM algorithm from more recent research. This was not within the scope of our work this time around, as it would have required a lot of time and knowledge in machine learning to accomplish. While there are numerous different algorithms floating around in NILM research, finding usable code for those algorithms is extremely difficult. Secondly, we would buy many more smart plugs, since they are so cheap, and collect data over a much longer period of time. This would help us obtain more data to train on and reduce the artificial nature of the data. As with all machine learning endeavors, the more data, the better.

References

[1] G. W. Hart, "Nonintrusive appliance load monitoring," in Proceedings of the IEEE, vol. 80, no. 12, pp. 1870-1891, Dec. 1992, doi: 10.1109/5.192069.

[2] H. Gonçalves, A. Ocneanu, and M. Bergés, “Unsupervised disaggregation of appliances using aggregated consumption data,” 2011.

[3] J. Z. Kolter, S. Batra, and A. Y. Ng, “Energy Disaggregation via Discriminative Sparse Coding,” in Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1, 2010, pp. 1153–1161.

[4] J. Kelly and W. Knottenbelt, “Neural NILM: Deep Neural Networks Applied to Energy Disaggregation,” Nov. 2015. doi: 10.1145/2821650.2821672.

[5] C. Zhang, M. Zhong, Z. Wang, N. Goddard, and C. Sutton, “Sequence-to-Point Learning with Neural Networks for Non-Intrusive Load Monitoring,” 2018.

[6] Hana Altrabalsi, Vladimir Stankovic, Jing Liao, Lina Stankovic. Low-complexity energy disaggregation using appliance load modelling. AIMS Energy, 2016, 4(1): 1-21. doi: 10.3934/energy.2016.1.1

[7] Makonin, S., Ellert, B., Bajić, I. et al. Electricity, water, and natural gas consumption of a residential house in Canada from 2012 to 2014. Sci Data 3, 160037 (2016). https://doi.org/10.1038/sdata.2016.37