Energy Disaggregation using Neural Networks

Purboday Ghosh, Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, USA

Abstract

Deep Learning has gained a lot of prominence in the past decade in solving complex prediction-based problems. This paper deals with the Energy Disaggregation problem in Power Systems Engineering and examines a solution approach using Neural Networks. A deep neural net model is proposed and its performance is evaluated.

1. Introduction

The use of deep machine learning approaches has yielded extremely promising results in fields such as image classification, Natural Language Processing (NLP) and anomaly detection. This has motivated Industry and Academia to explore learning-based solutions in other sectors as well, including the Energy Industry. Machine learning has been applied to the problem of load forecasting and generation prediction in Demand-Supply Management for Smart Grids. It has also been used for line-fault prediction in self-healing grids.

Another area that has witnessed a recent spurt in the application of learning-based predictive techniques is the problem of **Energy Disaggregation**. It encompasses the set of computational techniques using which individual appliance energy usage metrics can be estimated from the aggregate energy consumption information. The aggregate information is usually in the form of smart meter readings sampled periodically for a house. Then the disaggregation algorithm tries to quantify how much each appliance within the house, say, a refrigerator, or a washing machine accounts for in the total power consumed. It belongs to the class of problems known as **Non-Intrusive Load Monitoring (NILM)**. A use case for such algorithms is the production of itemized energy bills. An example of an energy disaggregation application is shown in Figure 1. The benefit of providing such information is in encouraging energy consumption awareness which when combined with real-time feedback to households inspire positive behavioural change and engage households toward sustainable energy consumption. It can also help the suppliers and policy makers to evaluate the efficacy of their implementation strategies and plan for the future.

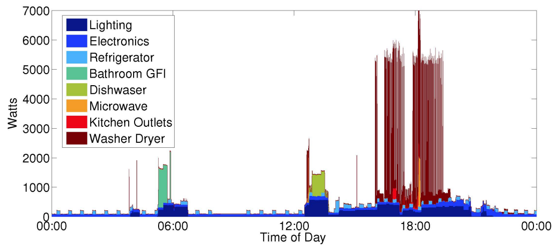

Figure 1. A Disaggregation Example (LIDS)

In this paper, a Deep Neural Network model is proposed to estimate the appliance energy consumption from the aggregate consumption data for a residential load. The organization of the paper is as follows: Section 2 describes the disaggregation problem and the steps involved in the process. Section 3 describes the data set that has been used in the paper for training and testing the algorithm. Section 5 presents the results of training the model and testing it on the data provided. Section 6 puts this work in context with other solution methods that have been explored in this field and finally Section 7 provides some closing remarks.

2. Problem Description

The problem of energy disaggregation can be mathematically formulated as a decomposition problem. Let $X = \{X_1, X_2..., X_T\}$ denote the sequence of aggregate power consumption obtained from a smart meter constituting N active appliances in chronological order for time $t = \{1, 2..., T\}$. The task of an NILM algorithm is to, given this aggregate data, infer the individual contributions of each appliance given by ${y_t}^i$ where $i \in \{1, 2..., T\}$, such that, at any point in time t,

$$X_t = \sum_{i=1}^{N} {y_t}^i + n(t) \quad (1)$$

where $n(t)$ represents any unaccounted appliances and measurement noise.

NILM algorithms consist of the following steps:

1. **Data Acquisition**: In the first step, load measurement

data is collected from equipment such as smart meters at an adequate rate from which appliance consumption patterns can be extracted. The aggregate readings can be sampled at different rates. High-frequency is when sampling rate is in a range of 10MHz to 100MHz for the quantity whose electrical characteristics is to be determined. Power meters for this range are often custom-built and expensive due to sophisticated hardware. Smart meters belong in the low sampling rate of the power signal which is less than 1 Hz.

2. **Appliance Feature Extraction**: In this step, the individual appliance signatures are identified using some set of features such as power, current, voltage etc. Different devices have their unique power consumption patterns which are referred to as their signatures. These signatures can be transient or steady-state. Transient features are short-term fluctuations that occur during appliance state transitions. These events usually relate to turning on or turning off appliances and require high frequency hardware to be detected. On the other hand, steady-state features include active power, reactive power, current, voltage etc which can be observed at much lower frequencies.

3. **Inference and Learning**: In this phase, the model uses the extracted appliance signatures to *learn* the model parameters and *infer* the individual appliance power contributions. The learning algorithms can be *supervised* or *unsupervised*. Supervised learning algorithms require a training phase in which the model is calibrated using both the aggregate data as well as the appliance data before being deployed. Unsupervised learning does not require pre-training and require only the aggregate data to learn the model parameters.

This paper utilizes Neural Networks and readily available measurement data to develop a supervised learning model. Thus, the task defined for the deep model is to perform **Regression**.

3. Dataset Description

For the purpose of training and evaluation, the data that was used was taken from **The Reference Energy Disaggregation Data Set (REDD)**(Kolter & Johnson, 2011), an open-source data set created for disaggregation research. It consists of several weeks of power data for six different homes, and high-frequency current/voltage data for the main power supply of two of these homes. The organization of the data is as follows:

1. **Low Frequency Data**: It contains average power readings for both the two power mains and the individual circuits of the house, logged at a frequency of 1 Hz for the mains and 1/3 Hz for the circuits. Data fo the mains as well as each appliance is stored in a separate ”channel_i.dat” file, with 1 and 2 for the two mains and the rest for others. A ”labels.dat” file lists the mapping between the channel and the load type.

2. **High Frequency Data**: It contains AC waveform data for the power mains and a single phase of the voltage for the homes. The data consists of three files, ”current_1.dat” and ”current_2.dat” for the current wave forms and ”voltage.dat” for the voltage waveform.

   The data storage is in the form of a time-series where the first column is the UTC time-stamp and the second column is the actual reading.

For this paper, low frequency data was chosen for the mains and the dishwasher unit (channel_6) for house 1 for training and testing.

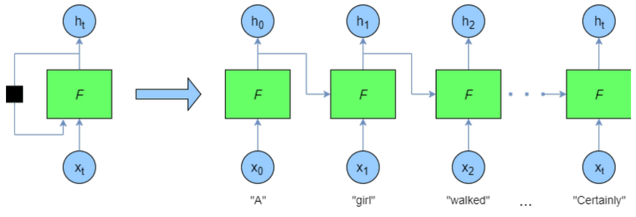

Figure 2. Unrolling a RNN

4. Model Architecture

This section discusses the Neural Network model architecture selected for the disaggregation task. Since the data has a strong temporal co-relation a Recursive Neural Network (RNN) architecture was chosen.

4.1. Background of RNN and LSTM

A RNN is a type of neural network where each layer relies on its previous state along with the input value to determine its output. Thus, it is highly suited to perform processing on sequential data where each subsequent value in the sequence is dependent on the previous value. The hidden layer output is fed back onto itself at the next time step. The network is ”unrolled” as the data sequence is processed, as shown in Figure 2.

To overcome the problems of vanishing and exploding gradients in long RNNs, the Long Short Term Memory (LSTM) model was introduced. An LSTM cell contains ””gates” that control the effect of the previous input and it replaces the multiplication of states and inputs by addition. The arrangement of a LSTM block is shown in Figure 3.

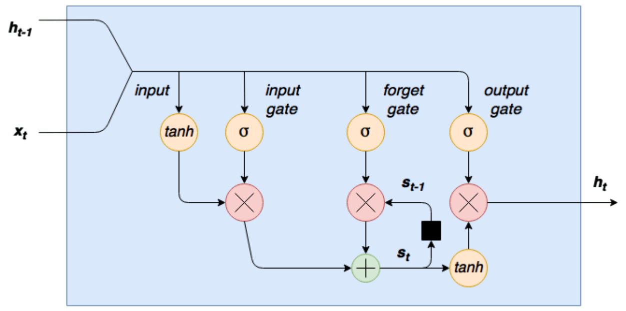

Figure 3. A LSTM cell

In traditional RNNs, information flow occurs in one direction only, i.e. from the past to the future. In Bidirectional RNN (BRNN), the output layer can get information from both the past as well as future states simultaneously. In this architecture, one hidden layer neurons are connected in the forward time direction while in the other they are connected in the backward time direction. This allows each neuron to have more information available to it.

4.2. Layer Description

The network consists of six layers with two LSTM layers, two fully connected layers and one convolutional layer. The input data consists of seven features namely month, week, hour, minute, second, mains1 power, mains2 powerwhich produces a single value at the output, which is the predicted contribution of the dishwasher unit. The details are as follows:

1. Input layer of length 7×1.
2. 1D convolutional layer with kernel size=4, stride=1, number of filters=16 and activation function=RelU.
3. Bidirectional LSTM layer with N=128.
4. Bidirectional LSTM layer with N=256.
5. Fully connected layer with N=128 and activation function=RelU.
6. Output layer with N=1 and activation function = Sigmoid.

5. Training and Evaluation

The model described in Section 4 was trained on a training set and its performance was evaluated using a test data set. The *Keras* Deep Learning package for Python was used to implement the model. The steps involved in the process were as follows.

5.1. Data Pre-processing

In the first stage, the REDD data set was imported. As discussed previously, for this paper, the low frequency data for house 1 was used and the model was trained for the dishwasher load. Thus, channel_1, channel_2 and channel_6 were loaded. Next, all three data frames were joined based on the time-stamp column. This was done to eliminate any empty rows in the data set. From the time-stamp, the month, week, hour, minute and second values were extracted. These were used as input features as the power consumption of a house has a strong relation with the time of the day and the season of the year. Next, the data was decomposed into the input X (time features and two mains readings) and the labels y(dishwasher readings). Thus the input data had a dimension of $num\_samples \times 7$ while the output was a single value.

Following the extraction, each column of the input data was normalized by subtracting the mean of each column and hen dividing by its standard deviation.

The input data was reshaped to form a three-dimensional array of shape ($num\_samples$ $\times 7 \times 1$). Similarly, the labels were reshaped to ($num\_samples$ $\times 1$).

The data set was partitioned into training and testing sets. Due to computational resource constraints, a training size of 20000 samples was used and a testing size of 2000 samples was used. Also, since the data is time-dependent, the testing sequence was selected to fall immediately after the training sequence. The starting point of the training window was selected randomly. Thus, the model learned using a block of 20000 samples and predicted for the following 2000 samples in the testing phase.

5.2. Model Creation

The model was created as per the specifications in Section 4. Dropout regularization was added after the convolutional layer and the two fully-connected layers with rates of 25%, 50% and 50% respectively. *Adam* optimization algorithm was also incorporated to take into account the effect of momentum and adaptive learning rates.

The performance metric used for training and evaluating the model was the **Mean Squared Error (MSE)**. The number of epochs over which the training was carried out was chosen as 20 and the batch size was selected to be 256.

5.3. Results

The learning curve is shown in Figure 4. It plots the loss function over the length of epochs. From the figure it can be seen that the loss value decays fairly quickly and is maintained at a low value indicating that the gradient descent process is executed correctly. The final value of the training

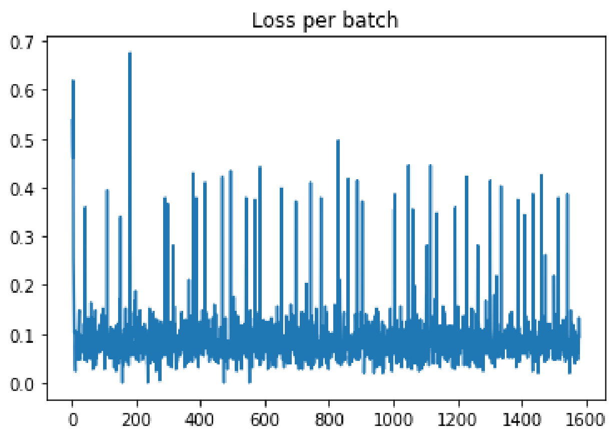

Figure 4. Learning Curve

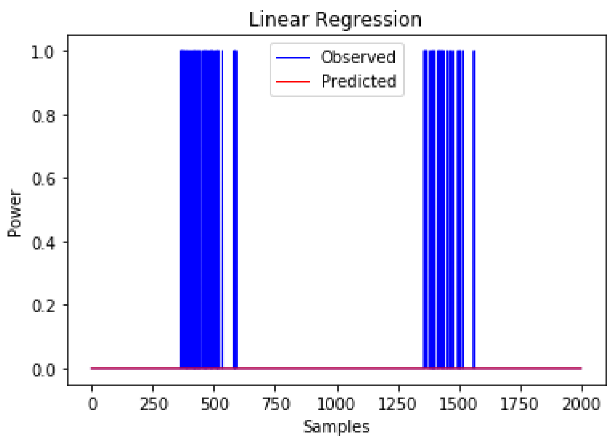

Figure 5. Prediction performance for the dishwasher

loss is 0.0899 and that of the validation loss is 0.0400. The loss curve has some noise associated and is not perfectly smooth. This is the effect of adding dropout which is a regularization scheme used to prevent over fitting.

The prediction accuracy is depicted in Figure 5. The predicted value is indicated by the red line and the actual value is denoted by the red line. It can be seen that the model is able to trace the observed values for the most part. However, since the observation values are extremely sparse, the model predicts a large number of the values to be zeros, causing a bias towards zero values.

This problem can be overcome if the model is trained on a larger window of data. For example, if the data for the first three weeks is used for training the model and that for the fourth week is used for testing, then the model is most likely to perform much better at the prediction task. However, such large volumes of data require higher computational resources which were not available for the scope of this paper.

6. Related Work

Several supervised as well as unsupervised NILM algorithms have been introduced and implemented over the years, with varied results. Some of the popular approaches are listed below:

6.1. Hidden Markov Models

A **Markov chain** is a probabilistic model of a system where the current state depends only on the previous state. A Markov chain assumes that at any point of time the state of the system is completely observable, if that is not so, then the associated model is known as a **Hidden Markov Model (HMM)**. An HMM model is specified as a triplet:

$$\lambda = (\Pi, A, B) \tag{2}$$

where Π represents the initial state probabilities, A represents the *Transition Matrix* where A_{ij} = P(next state is j | current state is i), B is the *Emission Matrix* where $B_{k,i}$ = P(value k is observed | state is i).

In the context of energy usage, the states of the system could be binary On or Off while the observations could be the power consumption values. HMM based models are mainly useful for state estimation, i.e. determining which devices are On and Off based on the meter readings (Kim et al., 2011). There are several algorithms in place, such as the *Viterbi* algorithm that can iteratively predict the sequence of states given a model and an observation sequence. (Kolter & Jaakkola, 2012) uses an extension of the basic HMM model called Factorial HMM (FHMM) that achieves better performance.

Despite the fact that HMM-based NILM approaches have been widely used in energy disaggregation they re- quire an expert knowledge to set a-priori values for each appliance state. Their performance are thus limited by how well the generated models approximate appliance true usage.

6.2. Graph Signal Processing

Graph Signal Processing (GSP) is a means of expressing information flow among entities in a network using nodes and edges in the form of a graph (Sandryhaila & Moura, 2014). These networks can include social media networks, traffic networks or a power grid. GSP represents a data set using a graph signal defined by a set of nodes and a weighted adjacency matrix. GSPs are a powerful approach to solve learning problems.

Given a set of aggregate power measurements X, Let $G =$

$\{V, A\}$ be a graph where V is the set of vertices for each $x_i \in X$ and A is the weighted Adjacency Matrix. In the context of energy disaggregation, each vertex can represent the variation in the aggregate power signal between two two successive samples.

GSP based approaches are most suited for supervised classification problems (Zhao et al., 2015). They are quite sensitive to noise and fluctuations of data, which are quite common in electrical devices (Zhao et al., 2016).

Neural network base prediction schemes have been shown to provide good results provided adequate amount of data is available. The training process is dependent on availability of large amounts of data covering different conditions which the model can learn to produce generalized results. Several kinds of architecture have been proposed. (Kelly & Knottenbelt, 2015) perform a comparative analysis between three different types of networks for energy disaggregation, namely a RNN LSTM model, an Auto Encoder model and a CNN model and evaluate them using specialized metrics.

7. Conclusion

In this paper a Deep Neural Network based architecture was introduced and implemented to address the energy disaggregation problem for home appliances. The model was trained using real world data and the performance was verified using the Mean Squared Error. The results show that the model was able to learn its parameters based on the training data and was able to achieve reasonable performance milestones. The prediction can be further improved by using higher computational resources to train on a larger volume of data. These possibilities can be investigated in future works.

Acknowledgement

The author would like to thank the instructor Dr. Xenofon Koutsoukos and the TA Dustin Zhou for their guidance and feedback throughout the duration of the course.

References

Kelly, J. and Knottenbelt, W. Neural nilm: Deep neural networks applied to energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pp. 55–64. ACM, 2015.

Kim, H., Marwah, M., Arlitt, M., Lyon, G., and Han, J. Unsupervised disaggregation of low frequency power measurements. In *Proceedings of the 2011 SIAM international conference on data mining*, pp. 747–758. SIAM, 2011.

Kolter, J. Z. and Jaakkola, T. Approximate inference in additive factorial hmms with application to energy disaggregation. In *Artificial intelligence and statistics*, pp. 1472–1482, 2012.

Kolter, J. Z. and Johnson, M. J. Redd: A public data set for energy disaggregation research. In *Workshop on Data Mining Applications in Sustainability (SIGKDD), San Diego, CA*, volume 25, pp. 59–62, 2011.

LIDS, M. Power disaggregation. URL `https://lids.mit.edu/research/research-highlights/power-disaggregation`. Accessed: 2019-04-19.

Sandryhaila, A. and Moura, J. M. Big data analysis with signal processing on graphs. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014.

Zhao, B., Stankovic, L., and Stankovic, V. Blind non-intrusive appliance load monitoring using graph-based signal processing. In *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 68–72. IEEE, 2015.

Zhao, B., Stankovic, L., and Stankovic, V. On a training-less solution for non-intrusive appliance load monitoring using graph signal processing. *IEEE Access*, 4:1784–1799, 2016.