A Project Report

on

A Deep Learning Approach to predict Appliance usage behaviour at a Household Level using RNN-LSTM Model

BY

ATHARVA WASADE

2022AAPS0469H

Under the supervision of

Dr. Sudha Radhika

SUBMITTED IN FULFILLMENT OF THE REQUIREMENTS OF

ECE F266 Study Oriented Project Course

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI

HYDERABAD CAMPUS

(APRIL 2024)

CONTENTS

ACKNOWLEDGMENTS

I wish to convey my heartfelt appreciation to the individuals and institutions acknowledged below for their invaluable contributions to this research endeavour:

I extend my deepest thanks to **Dr. Sudha Radhika**, my esteemed academic advisor, for their unwavering availability and expert guidance throughout this research journey. Their profound knowledge and extensive experience in this field have greatly propelled our work forward. I also thank **Mrs. Ajitha Tirumuru** for her valuable guidance throughout the completion of the research.

My profound gratitude goes to **BITS Pilani, Hyderabad Campus**, for providing the necessary support and resources that facilitated both on-site and remote work on this project. Their assistance has been indispensable in achieving our goals.

I am grateful to all my friends whose direct or indirect contributions have impacted this project positively. Your encouragement and support have significantly influenced my dedication and the overall quality of our efforts.

In conclusion, I want to express sincere thanks to the aforementioned individuals and institutions for their steadfast support, which has been instrumental in the completion and success of this study.

Birla Institute of Technology and Science-Pilani,

Hyderabad Campus

Certificate

This is to certify that the project report entitled "A Deep Learning Approach to predict Appliance usage behaviour at a Household Level using RNN-LSTM Model" submitted by Mr. Atharva Wasade (ID No. 2022AAPS0469H) in fulfillment of the requirements of the course ECE F266 Study oriented Project Course, embodies the work done by him under my supervision and guidance.

Date: 30/04/24

(Dr. SUDHA RADHIKA)

BITS- Pilani, Hyderabad Campus

ABSTRACT

Machine Learning for usage behaviour with Appliance-Level Data

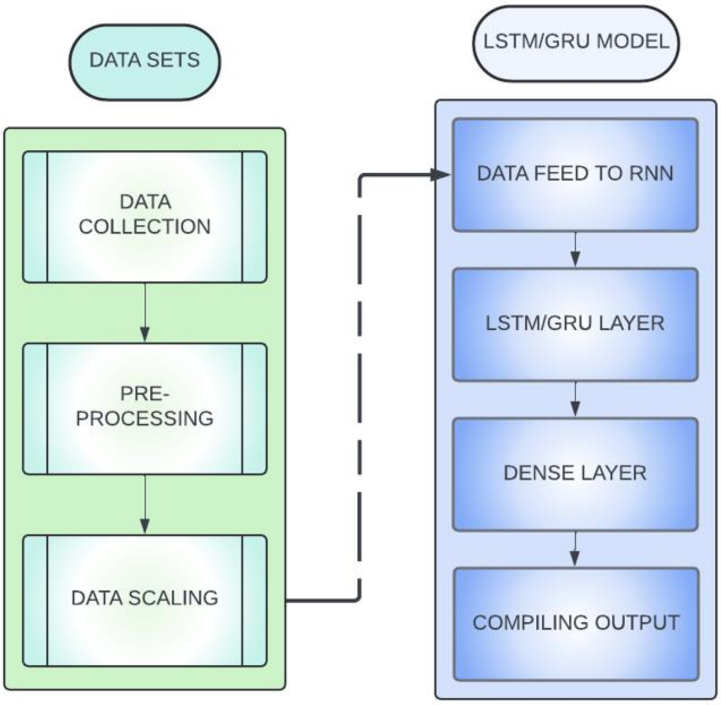
This project investigates the application of machine learning for prediction of user behaviour in the residential sector. It focuses on predicting appliance usage behaviour throughout the day, leveraging granular appliance-wise usage data. By analysing historical consumption patterns of individual appliances, the project aims to train machine learning models that can identify features and relationships and form reliable prediction for home appliance usage. After our initial idea of obtaining load data through a survey which did not yield a desired dataset. We then moved on to a better idea of obtaining open source data of appliance wise electricity consumption. We then applied several ML models such as Linear Regression, Support Vector Machine(SVM), Decision Tree, Random Forest. Most of these trivial ML models failed. After an extensive literature review we found for such a prediction a DL(Deep Learning) model called LSTM(Long Short-Term Memory) is commonly used. We have used this model while altering its parameters such as number of neurons, layers, epochs and batches and it has succeeded by giving an acceptable accuracy of approximately 85%. The analysis results show that the deep learning method has advantages in the prediction of household appliance energy consumption.

INTRODUCTION

There is an ever increasing demand for electricity in the residential sector. The rate of usage is increasing exponentially which leads to the need for load management. A majority of the electricity usage comes from the appliance in a household. The electricity consumption of a house is mainly related to the type and quantity of household appliances. A house energy management system, or HEMS, is basically a smart brain for your home's electricity. It tracks your energy use, helps you control devices, and can even optimize your consumption to save you money and reduce your environmental impact. For effective load management of any type, the first thing needed is an accurate production of load usage(appliance level) at any given time. Different Machine learning models must be applied to see which one suits the data and type of prediction the best. For the prediction of household appliance energy consumption, the typical research in traditional machine learning methods is to use multiple regression, neural network, support vector machine and other methods to establish related models to predict the energy consumption of household appliances. To test several ML models we would need data. After obtaining an open source data set, we started to apply ML models. While most Machine Learning models failed in giving a high and acceptable accuracy we moved on to Deep Learning models. After an extensive literature review on DL models used for prediction of similar prediction patterns, we applied LSTM(Long Short-Term Memory) Model. It is a type of recurrent neural network (RNN) designed to address the vanishing gradient problem, enabling better learning of long-term dependencies. It utilizes memory cells with gates to selectively store, update, and retrieve information over sequences, making it effective for tasks requiring understanding of temporal patterns, such as natural language processing and time series analysis. Long short-term memory is commonly used in deep learning methods, it has great advantages in time series simulation and often achieves good results. For the prediction of household appliance energy consumption, the typical research in traditional machine learning methods is to use multiple regression, neural network, support vector machine and other methods to establish related models to predict the energy consumption of household appliances.

METHODOLOGIES

FLOWCHART



1) DATA ACQUISITION AND PROCESSING

Method 1 - Through survey -

Initially, we tried to collect data for home appliance usage throughout the day via circulation of a survey on a Google form.

But it had the following problems -

1. We collected hourly samples of data which is not very accurate for predicting and makes it impractical to use.

2. We couldn't gather a large enough data to apply proper machine learning models on it.

3. There was missing information for some appliances and some extra for the other which made it hard to put up proper alternative data for those points.

So, we moved on to online resources and found an article by the University of Strathclyde. They had data taken the following data

- Data collection platform recorded data at 6-8 second intervals for a period of 2 years across 20 houses.
- Aggregate + 9 Individual Appliance Monitors
- Environmental Sensors (Light, Movement, Temperature)

There were approximately 1 million data points for each household. We applied the different models on different combinations of 100 thousand data points.

This 100 thousand was split as 0.3:0.7 ratio into 30 and 70 thousand datapoints into testing and training datasets respectively. The preprocessing involved data scaling and also the non-zero load value datapoints were made its Boolean equivalent to show its on and off status. This was done due to the very short period of measurement of load usage of 5-6 seconds per interval.

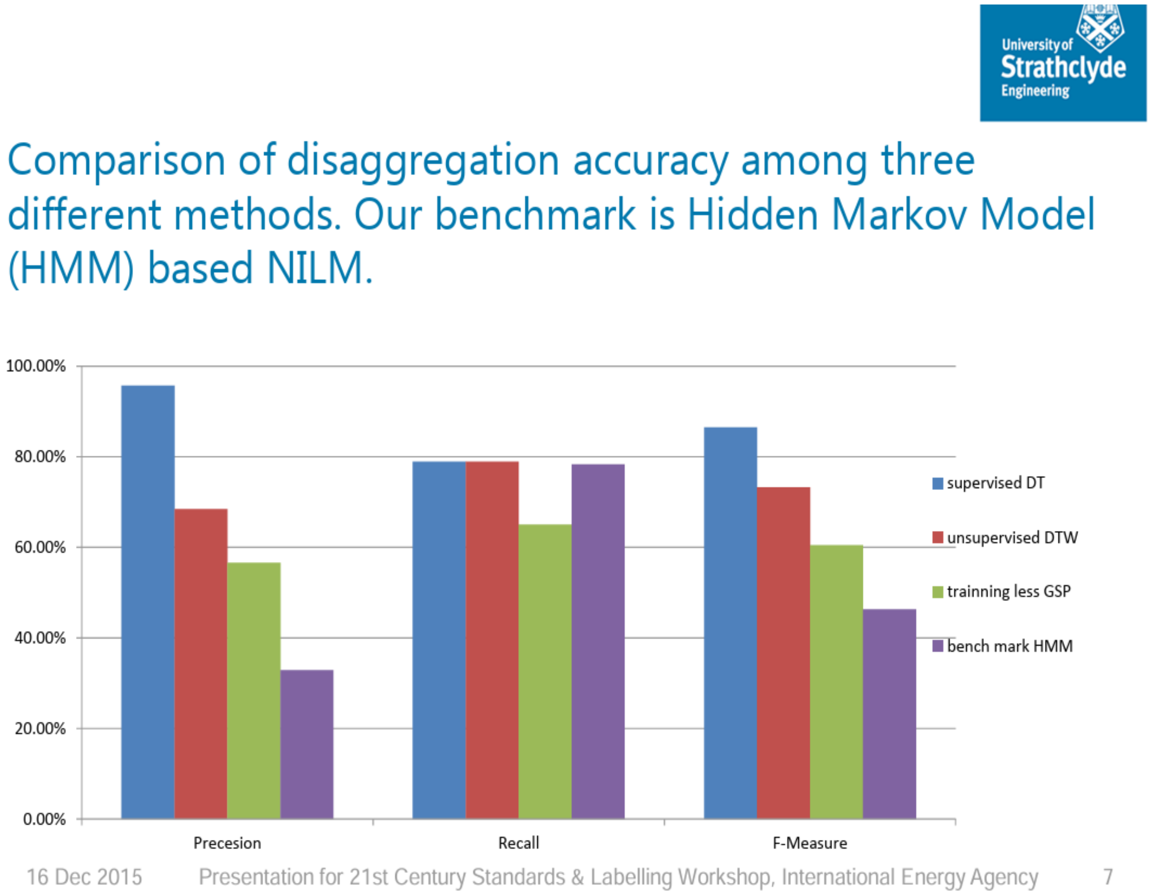
The time in Hours:Minutes:Seconds was converted to its corresponding value in seconds. The following figure shows a sample dataset used.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Time	Unix	Aggregate	Fan	Fridge	Appliance	Appliance	Appliance	Appliance	Computer	Television	Heater	Time_in_sec	time1
2	13:06:17	1.38E+09	523	74	0	69	0	0	0	0	0	1	47177	47177
3	13:06:31	1.38E+09	526	75	0	69	0	0	0	0	0	1	47191	47191
4	13:06:46	1.38E+09	540	74	0	68	0	0	0	0	0	1	47206	47206
5	13:07:01	1.38E+09	532	74	0	68	0	0	0	0	0	1	47221	47221
6	13:07:15	1.38E+09	540	74	0	69	0	0	0	0	0	1	47235	47235
7	13:07:18	1.38E+09	539	74	0	69	0	0	0	0	0	1	47238	47238
8	13:07:30	1.38E+09	537	74	0	69	0	0	0	0	0	1	47250	47250
9	13:07:32	1.38E+09	537	74	0	69	0	0	0	0	0	1	47252	47252
10	13:07:44	1.38E+09	548	74	0	69	0	0	0	0	0	1	47264	47264

2) Machine Learning Algorithms

Logistic regression model failed after giving Rsquare value of 0.6836.

```
Out[32]: ▾ LogisticRegression
         LogisticRegression()

In [33]: model.score(X_test.values.reshape(-1,1), y_test)

Out[33]: 0.6836
```



Based upon the above article[2] we ran our data through Supervised Decision Tree and Random forest. We got R-

square value of 0.7746 for Random Forest and 0.77456 for Decision Tree.

Table 1: Different ML Algorithms' Accuracy for Short Term Load Forecasting

Algorithms	MAPE	Algorithms	MAPE
C-Shape Clustering, LSTM networks, and Xgboost [5]	1.9%	Multi-scale convolutions (MS-CNN) [14]	0.98%
Bi-directional Long Short-term Memory (Bi-LSTM) Neural Network, Attention Mechanism (AM), and Rolling Update (RU) [9]	1.03%	(EMDHR-SVR-BPNN) [15]	0.04%
Generalized Regression Neural Network (GRNN) [6]	2.41%	Multi-temporal-spatial-scale Temporal Convolutional Network (MTCN) [16]	1.89%
Multivariable Linear Regression (MLR) [10]	2.88 %		
A hybrid ANN-based with modified Error Distribution Estimated (mEDE) in day-ahead load-forecasting (DALF) model [7]	1.24%	Stacking Fusion model [17]	0.88%
Bidirectional Recurrent Neural Network (Bi-RNN) and Deep Belief Network (DBN) [8]	1.95%	(STLF-IGEP_ALR) Improved Gene Expression Programming and Abnormal Load Recognition [18]	0.638%
Adaptive weight allocation strategy AWAS [11]	1.62%	Weighted k-nearest neighbor [19]	0.136%
FA fusion forecasting approach [12]	0.028%	Deep Neural Networks (DNN) [20]	2.08%
Fuzzy time series (FTS) [13]	1.32%	Parallel LSTM-CNN Network (PLCNet) [21]	1.48%

From the article[1], the MAPE for deep learning algorithms like Neural Networks is least.

After a literature review we found that the models to try for is GRU and LSTM.

3) Deep Learning Algorithms

1 – LSTM Model

LSTM, a type of recurrent neural network (RNN), is designed to overcome the vanishing gradient problem in traditional RNNs, enabling better learning of long-term dependencies in sequential data. It incorporates memory cells that can store information over long periods, selectively forgetting or updating information based on input signals. This allows LSTMs to capture temporal patterns and dependencies in data more effectively. With its ability to retain information over extended sequences, LSTMs have become a cornerstone in various applications such as natural language processing, speech recognition, time series forecasting, and more.

Our Model -

The parameters in an LSTM are epochs, layers and neurons. In our model we used 50 neurons in the hidden layer and 1 neuron in the output layer. The model uses 500 epochs and each batch has a size of 10. Activation parameter given is sigmoid. The optimizer used is Adam. The appliances whose data upon which we tested the LSTM model are a Fan, Television, Heater, Light. In our experimental and LabView setup we have three categories of loads which are critical, non-critical and programmable. Light and Fan fall under critical loads, Heater under programmable and Television under non-critical. We used 4 different datasets of the said appliances. The output parameter to check is the Accuracy of the model. Our result is as follows-

Appliance	Accuracy
Light	83.99%
Fan	83.84%
Television	85.6%
Heater	99.94%

We have an aggregate Accuracy of around 85% of the first three appliances and 99% for the Heater. The reason it is 99% for the Heater is because the model is overfitting due to the lack of change of user behaviour of the particular appliance. This also shows the model is quite accurate. For an LSTM model the ideal accuracy to show its functionality is 85-90% [5][6]

Since we have got an accuracy of 85% approximately this LSTM model with our set parameters for this type of data, this is successful model to predict user patterns of electrical appliances.

2 – GRU Model

The GRU (Gated Recurrent Unit) is a type of recurrent neural network (RNN) architecture designed to overcome some limitations of traditional RNNs. It utilizes gating mechanisms to selectively update and reset information in its memory cells, enabling better learning of long-term dependencies. GRUs have fewer parameters than LSTMs, making them computationally more efficient while still being effective in capturing temporal patterns. They are widely used in sequence modelling tasks such as time series forecasting.

Our Model

The parameters we used are 32 neurons in hidden layer 1, 16 neurons in hidden layer 2 and 1 neuron in output layer. 100 epochs with batch size 10. Activation parameter given is sigmoid. The optimizer used is Adam.

Our Result

For Fan appliance we got an accuracy of 83.66%. This is similar to the LSTM Model.

RESULTS AND DISCUSSION

Random Forest and Decision Tree Code Snippets

Random forest:

In [8]:
```
from sklearn.model_selection import train_test_split
```

In [12]:
```
X_train, X_test, y_train, y_test = train_test_split(y1,y, test_size=0.2)
```

In [11]:
```
from sklearn.ensemble import RandomForestClassifier
model1 = RandomForestClassifier(n_estimators=100)
model1.fit(X_train.values.reshape(-1,1), y_train)
```

```
C:\Users\athar\anaconda3\Lib\site-packages\sklearn\base.py:1151: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  return fit_method(estimator, *args, **kwargs)
```

Out[11]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [12]:
```
model1.score(X_test.values.reshape(-1,1), y_test)
```

Out[12]:
```
0.7746703860000477
```

Decision Tree:

In [12]:
```
from sklearn import tree
model = tree.DecisionTreeClassifier()
```

In [13]:
```
model.fit(X_train.values.reshape(-1,1), y_train)
```

Out[13]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [14]:
```
model.score(X_test.values.reshape(-1,1), y_test)
```

Out[14]:
```
0.774560713349069
```

LSTM Code Snippets Appliance wise

Fan -

```
In [215]: model.fit(X_train, y_train, epochs=5)

Epoch 1/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 2s 577us/step - accuracy: 0.8379 - loss: 12.2434
Epoch 2/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 1s 574us/step - accuracy: 0.8370 - loss: 12.3069
Epoch 3/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 1s 574us/step - accuracy: 0.8346 - loss: 12.4656
Epoch 4/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 1s 567us/step - accuracy: 0.8373 - loss: 12.2479
Epoch 5/5
2188/2188 ━━━━━━━━━━━━━━━━━━━━ 1s 574us/step - accuracy: 0.8372 - loss: 12.3251

Out[215]: <keras.src.callbacks.history.History at 0x18c40559690>

In [216]: model.evaluate(X_test, y_test)

938/938 ━━━━━━━━━━━━━━━━━━━━ 1s 535us/step - accuracy: 0.8397 - loss: 12.1681

Out[216]: [12.258438110351562, 0.8384000062942505]
```

Television -

```
In [26]: model.fit(X_train, y_train, epochs=5)

Epoch 1/5
219/219 ━━━━━━━━━━━━━━━━━━━━ 0s 605us/step - accuracy: 0.8484 - loss: 4.6231
Epoch 2/5
219/219 ━━━━━━━━━━━━━━━━━━━━ 0s 593us/step - accuracy: 0.8455 - loss: 4.7078
Epoch 3/5
219/219 ━━━━━━━━━━━━━━━━━━━━ 0s 590us/step - accuracy: 0.8494 - loss: 4.6452
Epoch 4/5
219/219 ━━━━━━━━━━━━━━━━━━━━ 0s 609us/step - accuracy: 0.8464 - loss: 4.7744
Epoch 5/5
219/219 ━━━━━━━━━━━━━━━━━━━━ 0s 599us/step - accuracy: 0.8547 - loss: 4.4289

Out[26]: <keras.src.callbacks.history.History at 0x1f6008ac390>

In [27]: model.evaluate(X_test, y_test)

94/94 ━━━━━━━━━━━━━━━━━━━━ 0s 624us/step - accuracy: 0.8552 - loss: 4.4840

Out[27]: [4.44758415222168, 0.8560000061988831]

In [ ]:
```

Heater -

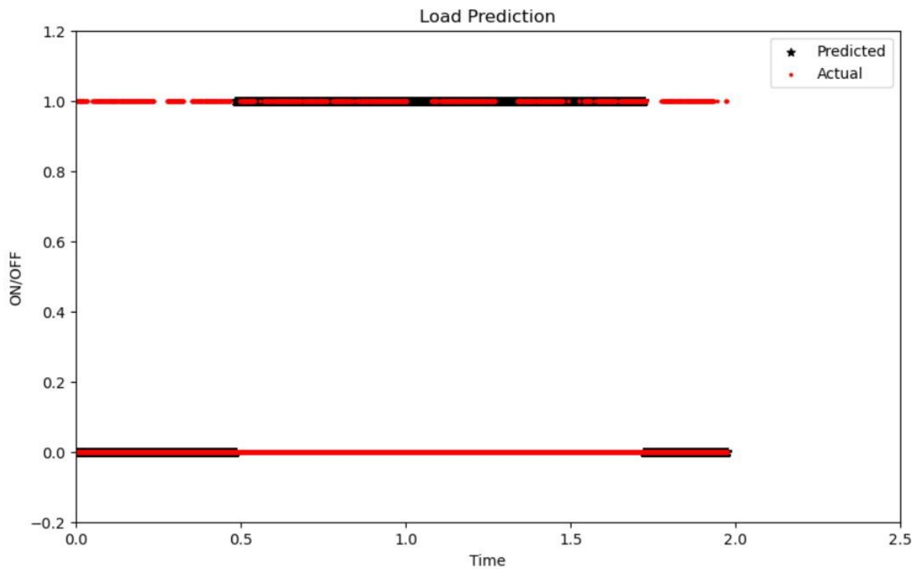
```
In [100]: from keras.layers import Dense
          from keras.models import Sequential

          model=Sequential()

          model.add(Dense(1,activation="softmax",input_shape=(1,)))
          #model.add(Dense(1,activation="softmax"))


          model.compile(optimizer='sgd',
                        loss='mae',
                        metrics=['accuracy'])

C:\Users\athar\anaconda3\Lib\site-packages\keras\src\layers\core\dense.py:85: UserWarning: Do not pass an `input_shape`/`input_
dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model i
nstead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)

In [101]: model.fit(X_train, y_train)

2500/2500 ━━━━━━━━━━━━━━━━━━━━ 1s 511us/step - accuracy: 0.9994 - loss: 5.6849e-04

Out[101]: <keras.src.callbacks.history.History at 0x18b3ab8dcd0>

In [102]: model.evaluate(X_test, y_test)

625/625 ━━━━━━━━━━━━━━━━━━━━ 0s 506us/step - accuracy: 0.9994 - loss: 5.8450e-04

Out[102]: [0.0005499999970197678, 0.9994500279426575]
```

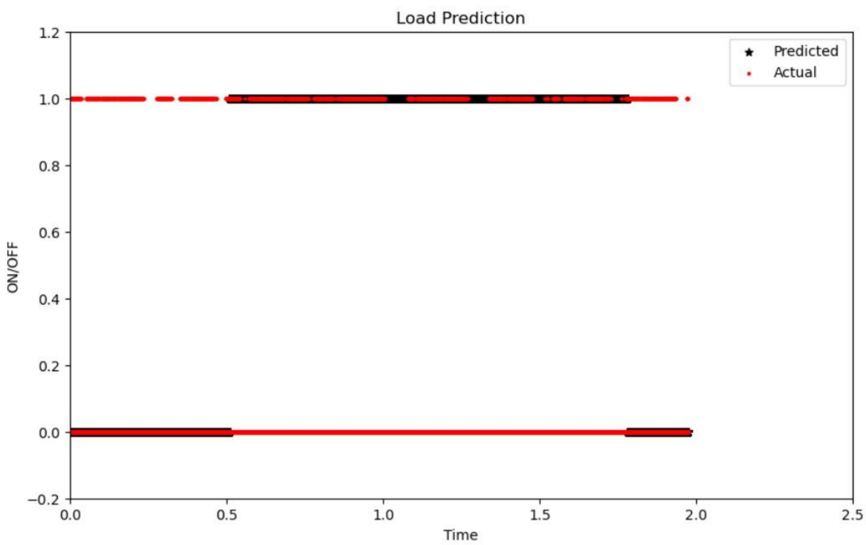
Light –

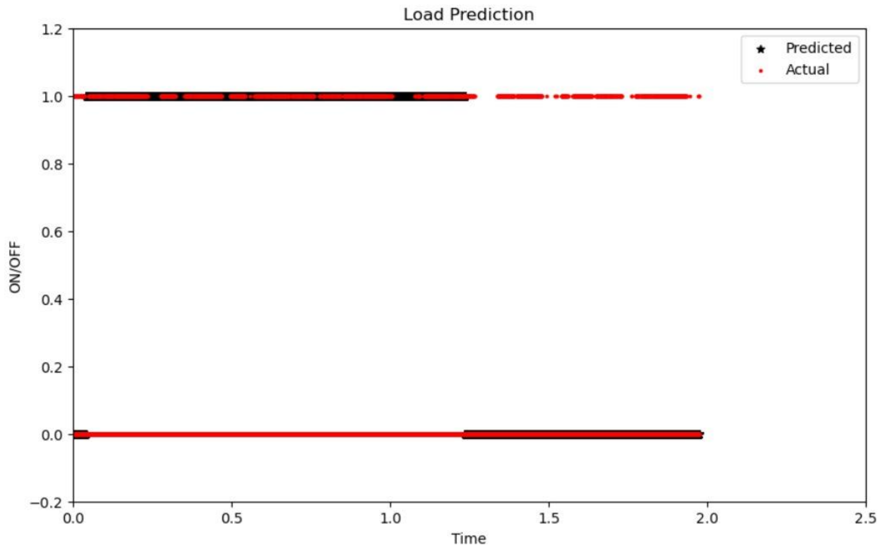
```
In [33]: model.fit(X_train, y_train, epochs=50,batch_size = 10)
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8349 - loss: 0.1378
         Epoch 26/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8357 - loss: 0.1373
         Epoch 27/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8366 - loss: 0.1368
         Epoch 28/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8397 - loss: 0.1346
         Epoch 29/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8387 - loss: 0.1353
         Epoch 30/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8349 - loss: 0.1378
         Epoch 31/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8367 - loss: 0.1366
         Epoch 32/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8353 - loss: 0.1376
         Epoch 33/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8341 - loss: 0.1384
         Epoch 34/50
         7000/7000 ━━━━━━━━━━━━━━━━━━━━ 10s 1ms/step - accuracy: 0.8339 - loss: 0.1385
         Epoch 35/50

In [34]: model.evaluate(X_test, y_test)
         938/938 ━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step - accuracy: 0.8390 - loss: 0.1350
Out[34]: [0.1344260573387146, 0.8399666547775269]
```

Output/Input graph of LSTM Model for fan, light and television appliance –

1 and 0 depicts on and off status







The resultant accuracy obtained for each of the models used on Fan appliance are as follows –

Model	Result
Logistic regression	68.36% (R-square score)
Random Forest	77.46% (R-square score)
Decision Tree	77.45% (R-square score)
Support Vector Machine	83.7% (R-square score)
LSTM	83.84% (Accuracy)
GRU	83.66% (Accuracy)

CONCLUSION

In conclusion, this study explored the efficacy of various machine learning and deep learning models for appliance energy prediction. While traditional models like regression, SVM, Random Forest, and Decision Tree achieved moderate accuracy (R-squared: 0.6-0.8), it was deemed insufficient for real-world applications. Conversely, deep learning models, particularly GRU and LSTM, demonstrated superior performance, reaching an accuracy of approximately 0.85. LSTM emerged as the most effective model for our specific dataset. However, it exhibited overfitting tendencies with the heater appliance due to its predictable usage patterns. This research highlights the potential of LSTM models for accurate appliance energy prediction, requiring careful parameter tuning to mitigate overfitting for specific appliance types. To conclude this paper, we have found that an LSTM is an accurate model through references and we have altered it parameters to get an accuracy of 0.85. Future research could explore techniques to mitigate overfitting and further enhance the model's generalizability.

References

1. Home Energy Management Machine Learning Prediction Algorithms: A Review Ohoud Almughram , Bassam Zafar , Sami Ben Slama3.

2. Analytical Tools for Understanding Appliance Usage Patterns and the Potential for Energy Savings David Murray, Lina Stankovic

3. HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving

4. State-of-the-Art Artificial Intelligence Techniques for Distributed Smart Grids: A Review

5. Arabic Opinion Mining Using Combined CNN - LSTM Models

6.The accuracy of the LSTM model for predicting the S&P 500 index and the difference between prediction and backtesting

7. Prediction model of household appliance energy consumption based on machine learning