प्रगत संगणन विकास केंद्र
CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

PROJECT REPORT
STUDY OF BATTERY USAGE PATTERN USING AI

Submitted In Partial Fulfilment of the Requirement for the Award of

Post Graduate Diploma in Artificial Intelligence (PG-DAI)

Under the Guidance of

Mr. Shivam Pandey

(Project Guide)

Submitted By: -

Bhanu Thapliyal	Meet Agnihotri	Suman Gokhare	Vaishali Verma
PRN: 220920528010	PRN: 220920528022	PRN: 220920528050	PRN: 220920528054

TABLE OF CONTENTS

प्रगत संगणन विकास केंद्र
CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

CERTIFICATE

CDAC, B-30, Institutional Area, Sector-62
Noida (Uttar Pradesh)-201309

This is to certify that Report entitled "**Study of Battery usage pattern using AI**" which is submitted by Bhanu Thapliyal, Meet Agnihotri, Suman Gokhare, Vaishali Verma in partial fulfilment of the requirement for the award of **Post Graduate Diploma in Artificial Intelligence (PG-DAI)** to **CDAC, Noida** is a record of the candidates own work carried out by them under my supervision.

The documentation embodies results of original work, and studies are carried out by the student themselves and the contents of the report do not from the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Mr. Shivam Pandey

(Project Guide)

ACKNOWLEDGEMENT

We would like to express our best sense of gratitude and endeavour with respect to Mr. Shivam Pandey (Project Guide) CDAC, Noida for suggesting the problems scholarly guidance and expert supervision during the course of this project. Special thanks to Mr. Ravi Payal (Program Coordinator). We are very thankful to Mr. Shivam Pandey (Project Guide) the project guide for constant simulated discussion, encouraging new ideas about this project.

Bhanu Thapliyal	Meet Agnihotri	Suman Gokhare	Vaishali Verma
PRN: 220920528010	PRN: 220920528022	PRN: 220920528050	PRN: 220920528054

ABSTRACT

According to a recent UN report, unless drastic cuts in greenhouse gas production are implemented immediately, global warming is likely to exceed the Paris Agreement target of 1.5°C above the pre-industrial level within the next 10–15 years. Climate change has caused weather and climate extremes, such as heatwaves, heavy precipitation, droughts and more intense tropical cyclones, across every region of the globe. Switching from fossil fuels to electric as fast as possible is the need of the hour. But the relatively low energy density of current battery technology has been a bottleneck. As such we hope to make testing new technologies a little faster with our project.

In this project we examine the datasets obtained by performing various tests on the Turnigy Graphene 5000mAh 65C cell by McMaster University in Hamilton, Ontario. We look for relationships between the different variables and attempt to apply various machine learning algorithms on it.

INTRODUCTION

The global economy is rapidly transitioning to portable and storable electricity, which impacts every major technology sector, especially consumer electronics, clean energy and mobility/transportation. The electric vehicle (EV) market, in particular, has seen a significant uptick in sales in the past decade. According to BloombergNEF (https://about.bnef.com/electric-vehicle-outlook), by 2040 over half of all passenger vehicle sales will be electric. Of course, this will also directly impact battery sales, making battery reliability and lifetimes will more important as we transition away from gas-powered vehicles.

One new upcoming technology utilizes the ‘wonder material’, graphene.

What exactly is Graphene?

There’s a good chance you’ve heard about graphene in the media before. Every few years there are breathless predictions of how this wonder material will transform various technologies. Graphene is basically just carbon.

Graphene is a one-atom-thick crystalline lattice of graphite, which is essentially crystalline carbon. This sounds like something incredibly fancy, but you can make flakes of graphene with a pencil and some sticky tape. The Nobel prize for doing so, though, has already been awarded.

Graphene has several properties that make it very exciting as a potential part of future technology. It has high thermal and electrical conductivity. So if you want to move electricity or heat with high efficiency, it’s a promising choice.

Graphene also exhibits a high level of hardness and strength. It’s very flexible and elastic. It’s also transparent and can be used to generate electricity from sunlight.

Its thinness, however, also makes it very hard to mass produce (for now).

Lithium-Ion Batteries Have Problems Graphene Won’t

Lithium batteries are the most energy-dense battery you can find in consumer electronics. They make devices like smartphones, drones, and electric cars possible. However, lithium batteries are volatile and need extensive safety circuitry to keep them stable.

They also degrade with every recharge, there's a limit to how much power they can deliver at once, and they have to be charged slowly lest the battery overheat and explode.

Batteries enhanced with graphene can fix or mitigate many of these issues. Adding graphene to current lithium batteries can increase their capacity dramatically, help them charge quickly and safely, and make them last much longer before they need replacement. The battery used here for testing boasts about 'going harder for longer' on its product page.

It also tells us about these advantages over traditional LiPo batteries-

• Power density: 0.15-0.17kw/kg (5Ah-16Ah)

• Power density: 0.13-0.15kw/kg (1Ah-4.9Ah).

• Stable High pack voltage through duration of use.

• High discharge rate, giving more power under load.

• Internal impedance can reach as low as 1.2mO compared to that of 3mO of a standard LiPo.

• Greater thermal control, packs stay much cooler under extreme conditions

• Higher capacity during heavy discharge.

• Maintains higher pack capacity even after hundreds of cycles

• Fast charge capable, up to 15C on some batteries.

• Longer Cycle Life (reportedly in excess of 900 during testing).

THE DATASET

1-Battery Main Specifications

Chemistry	**LiPo**
Nominal Voltage	**3.7 V**
Charge	**4.2V, 50mA End-Current (CC-CV) Fast**
Discharge	**2.8V End Voltage, 20A MAX Continuous Current**
Nominal Capacity	**5 Ah**
Energy Density	**134 (Wh/Kg)**

Cell removed from *Turnigy Graphene 5000mAh 3S 65C LiPo Pack* below, dimensions 144 x 51 x 33 mm.

A new 5Ah Turnigy cell (Turnigy Graphene 5000mAh 65C cell) was tested in an 8 cu.ft. thermal chamber with a 75amp, 5 volt Digatron Firing Circuits Universal Battery Tester channel with a voltage and current accuracy of 0.1% of full scale.

A series of tests were performed at six different temperatures, and the battery was charged after each test at 1C rate to 4.2V, 5mA cut off for charge current, at battery temperature of 25degC.

<u>The tests were performed as follows:</u>

1. Four pulse discharge HPPC test (1, 2, 5, and 10C discharge and 1, 2, 5, and 10C charge,

with reduced values at lower temperatures) performed at 100, 95, 90, 80, 70..., 20, 15, 10, 5, 2.5 % SOC.

2. C/20 Discharge and Charge test.

3. 0.5C, 2C and two 1C discharge tests. The first 1C discharge test is performed before the UDDS cycle, and the second is performed before the Mix3 cycle.

4. Series of four drive cycles performed, in following order: UDDS, HWFET, LA92, US06.

5. A series of eight drive cycles (mix 1-8) consist of random mix of UDDS, HWFET, LA92, US06.

The drive cycle power profile is calculated for a single Turnigy cell in a compact electric vehicle.

6. The previous tests are repeated for ambient temperatures of 40degC, 25degC, 10degC, 0degC, -10degC, and -20degC, in that order. For tests with ambient temperature below 10degC, a reduced regen current limit is set to prevent premature aging of the cells. The drive cycle power profiles are repeated until 95% of the 1C discharge capacity at the respective temperature has been discharged from the cell.

7. A series of 16 charges (charge 1-16) store the cell charging before and after every drive cycle starting with an initial top up before doing the Cap_1C test.

Data columns:

Time (time in seconds)

TimeStamp (timestamp in MM/DD/YYYY HH:MM:SS AM format)

Voltage (measured cell terminal voltage, sense leads welded directly to battery terminal)

Current (measure current in amps)

Ah (measured amp-hours, with Ah counter typically reset after each charge, test, or drive cycle)

Wh (measured watt-hours, with Wh counter reset after each charge, test, or drive cycle)

Power (measure power in watts)

Battery_Temp_degC (battery case temperature, at middle of battery, in degrees Celsius measured with a AD592 +/-1degC accuracy temperature sensor)

Missing data/notes:

-20 degC: There is no drive cycles data or 1C, 2C capacity tests caused by the fact that the Turnigy behaved poorly at this temperature, the cell could also not do the HPPC test at this temperature.

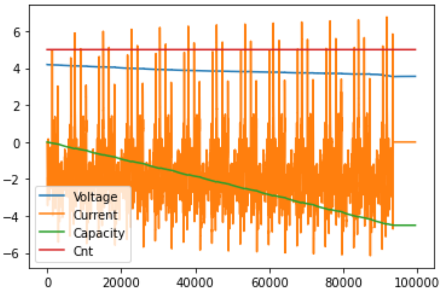
CODE SNIPPETS

In [3]:
```
data_1.info()
```

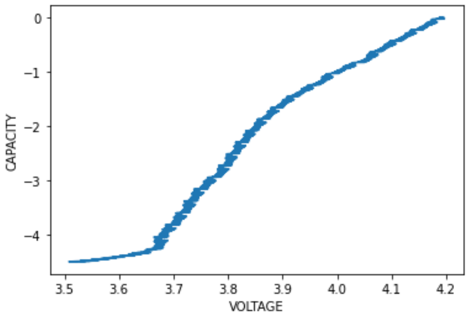
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99410 entries, 1 to 99410
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Time Stamp   99410 non-null  object 
 1   Step         99410 non-null  float64
 2   Status       99410 non-null  object 
 3   Prog Time    99410 non-null  object 
 4   Step Time    99410 non-null  object 
 5   Cycle        99410 non-null  float64
 6   Cycle Level  99410 non-null  float64
 7   Procedure    99410 non-null  object 
 8   Voltage      99410 non-null  object 
 9   Current      99410 non-null  object 
 10  Temperature  99410 non-null  object 
 11  Capacity     99410 non-null  object 
 12  WhAccu       99410 non-null  object 
 13  Cnt          99410 non-null  object 
dtypes: float64(3), object(11)
memory usage: 10.6+ MB
```

In [4]:
```
data_1[['Step','Cycle', 'Cycle Level', 'Voltage','Current','Temperature','Capacity','WhAccu','Cnt']] = data_1[['Step','Cycl

data_1['Prog Time'] = pd.to_datetime(data_1['Prog Time'], format='%M:%S.%f')
```

In [5]:
```
plt.plot(data_1['Voltage'].astype(float))
plt.plot(data_1['Current'].astype(float))
plt.plot(data_1['Capacity'].astype(float))
plt.plot(data_1['Cnt'].astype(float))
plt.legend(labels=['Voltage','Current','Capacity','Cnt'])
plt.show()
```



In [17]:
```
plt.plot(data_3['Voltage'], data_3['Capacity'])
plt.xlabel('VOLTAGE')
plt.ylabel('CAPACITY')
plt.show()
```



In [42]:
```
X = data.drop(['Temperature'], axis=1)
y = data['Temperature']

from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
y = encoder.fit_transform(y)

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.30,random_state=41)
print("X_train : ",X_train.shape, "y_train : ",y_train.shape, "X_test : ",X_test.shape, "y_test : ",y_test.shape)
```

```
X_train :  (417531, 10) y_train :  (417531,) X_test :  (178942, 10) y_test :  (178942,)
```

In [43]:
```
from sklearn.linear_model import LinearRegression

lr = LinearRegression()

lr.fit(X_train, y_train)
```

Out[43]:
```
▾ LinearRegression
LinearRegression()
```

In [44]:
```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = lr, X = X_train, y = y_train, cv = 2)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 47.33 %
Standard Deviation: 0.06 %
```

In [45]:
```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
```

Out[45]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [46]:
```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = rf, X = X_train, y = y_train, cv = 2)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 95.59 %
Standard Deviation: 0.03 %
```

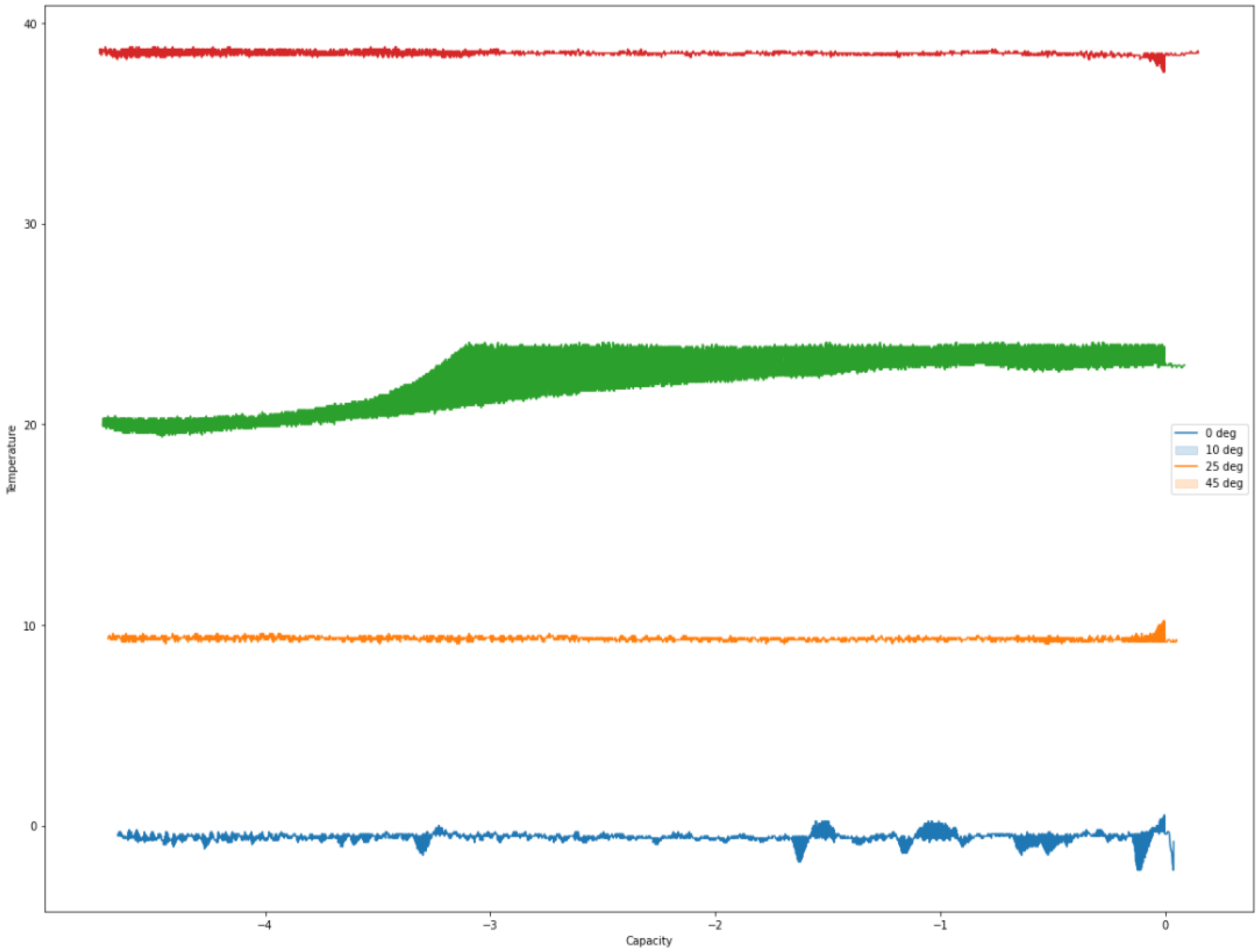
In [47]:
```
from sklearn.linear_model import LogisticRegression
log = LogisticRegression()
log.fit(X_train, y_train)
```

Out[47]:
```
▾ LogisticRegression
LogisticRegression()
```

In [48]:
```
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = log, X = X_train, y = y_train, cv = 2)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 42.07 %
Standard Deviation: 0.32 %
```

In [56]:
```
plt.figure(figsize=(20,15))
sns.lineplot(x=data__820_C20DisCh_0_deg['Capacity'], y=data__820_C20DisCh_0_deg['Temperature'])
sns.lineplot(x=data__800_C20DisCh_10_deg['Capacity'], y=data__800_C20DisCh_10_deg['Temperature'])
sns.lineplot(x=data__773_C20DisCh_25_deg['Capacity'], y=data__773_C20DisCh_25_deg['Temperature'])
sns.lineplot(x=data__761_C20DisCh_45_deg['Capacity'], y=data__761_C20DisCh_45_deg['Temperature'])
plt.legend(labels=['0 deg','10 deg','25 deg','45 deg'])
plt.xlabel('Capacity')
plt.ylabel('Temperature')
plt.show()
```

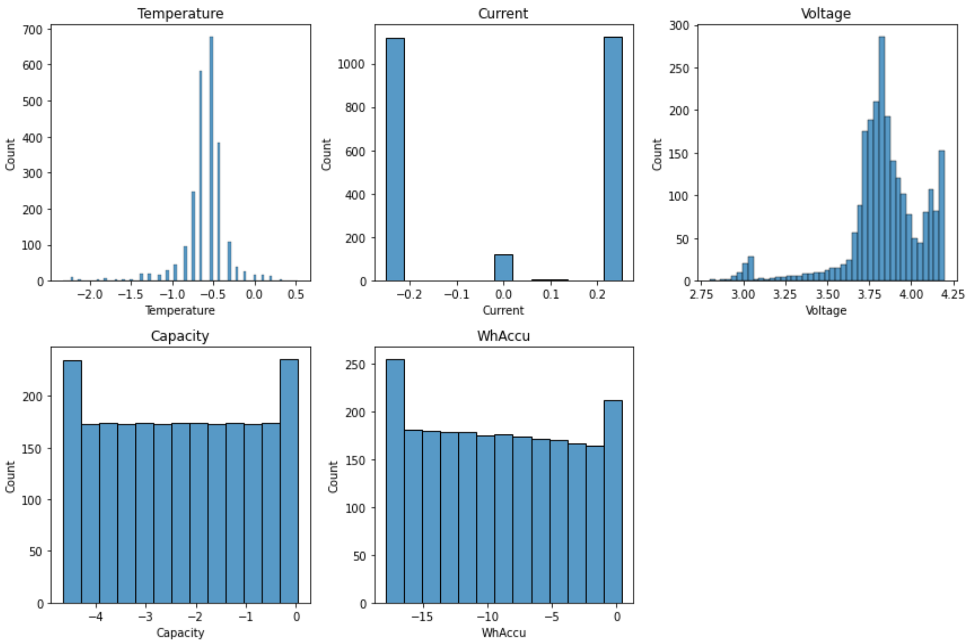


In [64]:
```
df = pd.concat([data__820_C20DisCh_0_deg, data__800_C20DisCh_10_deg, data__773_C20DisCh_25_deg, data__761_C20DisCh_45_deg], axis = 0 )
df = df.iloc[:,1:]

df[['Step','Cycle', 'Cycle Level', 'Voltage','Current','Temperature','Capacity','WhAccu','Cnt']] = df[['Step','Cycle', 'Cycle Level', 'Voltage','Curren

from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()

df['Status'] = encoder.fit_transform(df['Status'])
df['Procedure'] = encoder.fit_transform(df['Procedure'])

df
```

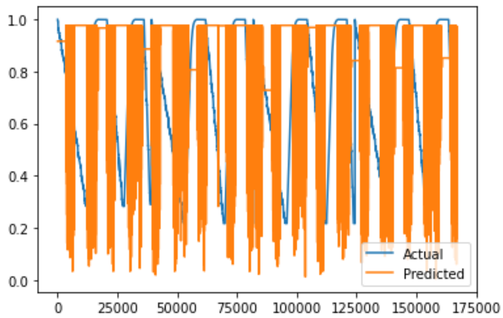
Out[64]:

	Step	Status	Prog Time	Step Time	Cycle	Cycle Level	Procedure	Voltage	Current	Temperature	Capacity	WhAccu	Cnt
1	24.0	1	1900-01-01 00:01:27.800	1900-01-01 00:01:00	0.0	0.0	0	4.194359	-0.251014	0.529946	-0.004167	-0.017482	1.0
2	24.0	1	1900-01-01 00:02:27.800	1900-01-01 00:02:00	0.0	0.0	0	4.192844	-0.248479	0.423957	-0.008337	-0.034969	1.0
3	24.0	1	1900-01-01 00:03:27.800	1900-01-01 00:03:00	0.0	0.0	0	4.191497	-0.251014	0.317968	-0.012508	-0.052454	1.0

In [61]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# create a figure and axes for subplots
fig, axs = plt.subplots(2, 3, figsize=(12, 8))

# plot the histograms on the axes
sns.histplot(data=data__820_C20DisCh_0_deg, x='Temperature', ax=axs[0,0])
sns.histplot(data=data__820_C20DisCh_0_deg, x='Current', ax=axs[0,1])
sns.histplot(data=data__820_C20DisCh_0_deg, x='Voltage', ax=axs[0,2])
sns.histplot(data=data__820_C20DisCh_0_deg, x='Capacity', ax=axs[1,0])
sns.histplot(data=data__820_C20DisCh_0_deg, x='WhAccu', ax=axs[1,1])

# remove the unused subplot
fig.delaxes(axs[1,2])

# set the titles for each subplot
axs[0,0].set_title('Temperature')
axs[0,1].set_title('Current')
axs[0,2].set_title('Voltage')
axs[1,0].set_title('Capacity')
axs[1,1].set_title('WhAccu')

# adjust the spacing between subplots
fig.tight_layout()

# show the plot
plt.show()
```



In [89]:
```
# Predict capacity using the random forest regression model
y_pred = rfc.predict(X_test[:166883])

# Visualize the actual and predicted capacity values
import matplotlib.pyplot as plt
plt.plot(y_test[:166883], label="Actual")
plt.plot(y_pred[:166883], label="Predicted")
plt.legend()
plt.show()
```



RESULTS, CONCLUSIONS AND FUTURE SCOPE

In our results we find that, while attempting to calculate battery capacity (state of charge), the random forest model is best suited to this dataset, and gives us a high accuracy of 95.59%. This unfortunately also results in an overfitted model, which is one of random forest's weakness.

For future scope we would need to perform more testing and obtain cleaner data, that ideally doesn't need to be balanced or have outliers removed. We could also perform hyperparameter tuning or apply an artificial neural network(ANN) model to get better predictions without overfitting. We could try to predict other battery variables, such as internal resistance (IR) or knee onset (start of degradation).

REFERENCES

1. IPCC. Summary for Policymakers. In Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change (eds. Masson-Delmotte, V. et al.) 1–31. In Press; 2021.

2. Kollmeyer, Phillip; Skells, Michael (2020), "Turnigy Graphene 5000mAh 65C Li-ion Battery Data", Mendeley Data, V1, doi: 10.17632/4fx8cjprxm.1

3. 5 Ways Graphene Will Change Gadgets Forever, Laptop, 14 April 2014, Michael Andronico https://news.yahoo.com/5-ways-graphene-change-gadgets-155243022.html

4. https://hobbyking.com/en_us/graphene-1000mah-3s-45c-w-xt60.html?___store=en_us

5. Kollmeyer, Philip; Vidal, Carlos; Naguib, Mina; Skells, Michael (2020), "LG 18650HG2 Li-ion Battery Data and Example Deep Neural Network xEV SOC Estimator Script", Mendeley Data, V2, doi: 10.17632/cp3473x7xv.2