In recent years, numerous approaches have been made to more accurately attempt energy load forecasting through the use of predictive modeling. Because energy use patterns are dependent on a wide variety of factors, determining an appropriate forecasting model for consumption behaviors is a highly specialized task. Therefore, models that are based on network specifications rather than generalization are preferred including: long-short-term memory (LSTM), the autoregressive integrated moving average (ARIMA), and other complementary models, such as vector autoregression (VAR), Bayesian vector autoregression (BVAR), and seasonal ARIMA (SARIMA).

Here, we investigated the combination of LSTM and ARIMA with the goal of creating a machine learning tool that better detects outliers in energy consumption. We investigated the combination of LSTM and ARIMA to better detect outliers in energy consumption. Each model analysis was performed using kWh power data, provided by TVA, for power meters from the years 2015-2018. The machine learning models were trained over two years of data and tested for comparison over data from 2018. The tested regions were Memphis (0138), Huntsville (0099), and Nashville (0158).

LSTM flagged residual values of the time series data that went above or below 3 standard deviations from the mean. ARIMA found instances where the difference between actual and predicted values was greater than 2000kwh.

Based on the work by UTC in Phase 1 of the project in 2018, we were able to validate the use of our model for the Phase 2 work. Each model was evaluated as a comparison between the individual meters the model detected as anomalies and the power company-provided meter report of anomalies. Providing the minimal amount of false anomalies detected, there are fewer meters that need to be reviewed manually. Our combination method had the highest accuracy and specificity compared to just using ARIMA and LSTM on their own. Without the implementation of our machine learning model, over 1440 data points have to be

reviewed with reliance on manual detection – using our model, only select meters would need to be reviewed.

To further implement this model on a wide scale, for Phase 2, we have created a series of functions that will result in a set of potential anomalies for any meter in a desired date range. The steps to implementing our model is as follows:

1. **‘IEE\_meter\_data.sql’ file**
   1. This is the SQL command that will retrieve the IEE data from the PowerBilling Database.
   2. You can specify the date range you wish to retrieve for training and building the model. For example, we used 2015-2018.
   3. The CustomerID of the region you wish to evaluate is required
2. Run this file and save the output as the “last\_reading” which should be the most updated meter data that is then sent to Oracle
3. Run this file again, but including “DESCENDING” at the end of the file to retrieve the “first\_reading” of the meters

B. **‘LSTM\_Prediction\_Build.py’**

1. This file contains the function for creating the prediction for each meter
2. The output of this function will be ‘.csv’ files with ‘meter\_number/Prediction’ as the path
3. Import the function ‘lstm\_prediction\_build()’ from this file and then run it on the desired region and with the last\_reading collected by step 1 as your input (The code for this can be seen in the **Test.py** file provided).

C. **‘Anomaly\_Detection\_Single\_day.py’**

1. This file contains the function for testing the prediction against actual values to detect potential anomalies
2. It requires the output from step 3 and the first\_reading collected by step 2.
3. Import the function ‘anomaly\_detection\_single\_day’ from this file and run it on the collected predictions from step 3 (again, the implementation of this function is explained in the provided **Test.py** file).
4. The output will be the potential anomalies that will then be evaluated. These will automatically be saved as .csv files by each meter and then by day.